WALKOWIAK

ADVENTURES

UND WIE MAN SIE AUF DEM

ATARI®

600XL/800XL
PROGRAMMIERT

(P) by GoodByteXL

Bearbeitungsstand: 7. Juni 2Ø22

Hinweise
Mit Ablauf des Jahres 2Ø13 wurde der Zeitschriften- und Buchverlag DATA Becker geschlossen; zum 31. März 2Ø14 wurde der gesamte Geschäftsbetrieb aufgegeben. Es gibt keinen Rechtsnachfolger.

Also gilt es zur digitalen Bewahrung der Bücher die Autoren zu finden und um Freigabe zu bitten. Ein Kontakt zu *Jörg Walkowiak* konnte bisher nicht hergestellt werden; die Spur seiner Computerbücher endet in der Deutschen National-bibliothek.

Die Programme in diesem Buch funktionieren nur begrenzt, bieten also noch reichlich Optimierungspotential.

Dank für die Hilfen und Informationen geht an

Tigerduck
CharlieChaplin
Rockford
Mr. Bacardi
Bernhard S.

Bit Byter *Rockford* hat die Programme geprüft und bewertet dieses Buch folgendermaßen: "Was ich dem Buch von J. Walkowiak mehr als fast allen anderen A8-Büchern zu Gute halten kann, und weswegen ich dieses Buch trotz der vielen Fehler für wichtig halte: Hier wird der Atari als kreatives Gerät genutzt. Es regt an, Welten zu erschaffen und andere Nutzer in deren Bann zu ziehen. Anders als beim Erstellen von Jump and Run oder Shootern hat nämlich der Autor des Spiels selbst nie einen Spielspaß daran, er kennt Story, Verlauf und Lösung schon. Seine Rolle ist die des Lenkers und Geschichtenerzählers. Und das ist das, was "creative computing" ausmacht. Dafür hat er bei mir 2 Daumen nach oben, denn Kreativität macht nämlich letztlich die sinnvollste Schnittstelle zwischen Mensch und Maschine."

Funktionierende Versionen in einer Bearbeitung durch einen A8-Fan findet ihr z. B. auf der Webseite von Mr. Bacardi für "Atari XL Basic Listings".

Anmerkung
Cover und Layout des Buches entsprechen weitestgehend dem des Originaldrucks von 1984. Durch notwendige Fehlerkorrekturen konnte das Werk nicht immer zeilengleich digitalisiert werden, aber die Seiteninhalte wurden gewahrt und erlauben so einen Abgleich zum originalen Buch aus 1984.

Das Buch erschien ursprünglich bei DATA Becker:

ISBN 3-89Ø11-Ø59-2

Copyright (C) 1984 DATA BECKER GmbH
Merowingerstr. 3Ø
4ØØØ Düsseldorf

Wichtiger Hinweis

Die in diesem Buch wiedergegebenen Schaltungen, Verfahren und Programme werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind ausschließlich für Amateur- und Lehrzwecke bestimmt und dürfen nicht gewerblich genutzt werden.

Alle Schaltungen, technische Angaben und Programme in diesem Buch wurden von den Autoren mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung wirksamer Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. DATA BECKER sieht sich deshalb gezwungen, darauf hinzuweisen, daß weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernommen werden kann. Für die Mitteilung eventueller Fehler ist der Autor jederzeit dankbar.

INHALTSVERZEICHNIS

VORWORT

1 EINFÜHRUNG 7
Abenteuer Heute. Geschichte und Entwicklung.

2 DAS KONZEPT 23
Aufmachung. Funktionen. Vorüberlegungen.

3 DIE VERWIRKLICHUNG 35
Gestaltung der Welt. Exkurs: Variablen und Felder. Exkurs: READ DATA. Formatierung der Ausgabe. Objekte. Wortschatz. Exkurs: Stringbehandlung. Analyse der Eingabe. Schematischer Überblick. Wann geht Was ? Die Bedingungen. Die Aktionen. Programmierung der Befehlsausführung. Letzte Schritte. Listing.

4 PERFEKTIONIERUNG 1Ø7
Save Game. Load Game. Exkurs: Externe Datenspeicherung. Benutzerfreundlichkeit. Motivierung des Spielers. Orientierung und Desorientierung. Ortswechsel. Versteckte Zugänge. Limitierungen. Verfügbares Licht. Zufälle. Vom Text- zum Grafikadventure.

5 ADVENTUREPRAXIS 163
Spielanleitung. Tipps zur Lösung. Listing Verzaubertes Schloß. Listing Goldrausch. Listing Space Mission. Listing Adventuregenerator. Listing Grafik-Editor.

6 ANHANG 275
Speicherplatzoptimierung. Ablaufoptimierung.

VORWORT

Dieses Buch richtet sich an all diejenigen unter den ATARI 6ØØ XL und ATARI 8ØØ XL Besitzern, die der zahllosen Schieß- und Actionspiele überdrüssig geworden sind und die sich nun mit intelligenteren Spielen befassen wollen.

Mit diesem Buch halten Sie den Schlüssel zur Welt der Adventurespiele in der Hand, den Schlüssel zu einer Welt, die Sie als Atari Freund bislang vermutlich nur am Rande kennengelernt haben.
Denn während Sie die verschiedensten Fachzeitschriften durchgeblättert haben, werden Sie immer wieder die Feststellung gemacht haben, daß die dort veröffentlichten Programme aus Ihrem Computer immer nur eine bessere Spielkonsole machten.

Daß, und wie das ATARI - BASIC neben seiner Verwendung für grafische Darstellungen auch für die Bearbeitung von Texten geeignet ist, werden Sie auf spielerische Weise bei der Beschäftigung mit ©*Adventures - und wie man sie auf dem Atari programmiert*©, feststellen.

Schritt für Schritt werden wir gemeinsam eine Konzeption zur Realisierung von Abenteuerspielen entwickeln, deren optische wie auch spieltechnische Erscheinungsform sich durchaus mit professionellen Adventures messen kann.

Selbst wenn Sie sich nicht zu den Programmierprofis zählen, kurze Exkurse, zur Kennzeichnung *kursiv* gedruckt, werden Sie in die zum Verständnis der Programme wichtigen Grundlagen einführen, so daß Sie auch als Anfänger mit dem Inhalt dieses Buches ohne Schwierigkeiten zurechtkommen werden.

Dabei hilft Ihnen auch der Aufbau der Programme, denn nach jedem Schritt werden Sie die Richtigkeit der Überlegungen sogleich in der Praxis erproben können.

So programmieren wir zunächst die Handlungswelt eines Adventures, dann werden wir uns überlegen, wie dem Spieler die gezielte Fortbewegung möglich gemacht werden kann, und wie wir ihm Gegenstände zur Hand geben können, mit denen er arbeiten kann.

Ein weiteres Kapitel wird Sie in die besonderen Geheimnisse der Adventureprogrammierung einweihen; es wird Ihnen zusätzliche Vorschläge machen, wie Sie einem Spieler Ihrer Adventures das Leben erschweren können.

Darüber hinaus finden Sie in diesem Kapitel genaue Anweisungen, wie Sie jedes erstellte Textadventure zu einem Grafikadventure ausbauen können.

Und damit die Gestaltung der Grafiken nicht allzu arbeitsaufwendig wird, erhalten Sie auch das Listing eines Grafik Programmers, der die erforderlichen Unterprogramme für Sie erzeugen wird.

Aber es bleibt nicht allein bei diesem Hilfsprogramm, sondern mit dem Listing ©Venturefix© stellen wir Ihnen einen Adventuregenerator zur Verfügung, der es Ihnen, selbst wenn Sie Ihren ATARI erst heute erstanden haben sollten und somit keinerlei Programmierkenntnisse haben, erlaubt, voll funktionsfähige Adventureprogramme zu erzeugen.

Und sollten Sie es vorziehen, zunächst erst einige Adventures zu spielen, so schlagen Sie bitte das letzte Kapitel auf, denn hier finden Sie die Textadventures *Goldrausch* und *Das verzauberte Schloß*, wie auch ein Listing für das Grafikadventure *Space Mission*.

Ich nehme an, daß Sie sich nun lieber mit dem eigentlichen Thema dieses Buches statt mit einer langen Vorrede befassen wollen, und so bleibt mir im Moment nur noch übrig, mich für den Kauf dieses Buches recht herzlich bei Ihnen zu bedanken.

Darüber hinaus gilt mein besonderer Dank Herrn Dr. Achim Becker, der die Herausgabe dieses Buches möglich gemacht hat, wie auch Frau Alicia Clees und Herrn Claus Wagner, die mich bei der redaktionellen Arbeit auf unermüdliche Weise unterstützt haben.

Recklinghausen,
Oktober 1984 Jörg Walkowiak

Wichtige Anmerkungen zur Eingabe der BASIC-Programme:

BASIC-Programmzeilen
In diesem Buch sind überlange Programmzeilen in ATARI-BASIC ausgedruckt, die sich nicht in dieser Form eingeben lassen.

Der Autor setzt einerseits voraus, dass die Standardvorgabe des ATARI mit Schreibbeginn auf Position 3 einer Zeile eingehalten wird, erzeugt andererseits aber überlange Zeilen, die sich nur mit Tricks erzeugen lassen und darüber hinaus den Schreibbeginn auf Position 1 einer Zeile bedingen.

Der ATARI kann maximal 3 Bildschirmzeilen zu je 4Ø Zeichen, insgesamt also 12Ø Zeichen in eine BASIC-Zeile übernehmen. Mit der Standardvorgabe lassen sich nur 3 Bildschirmzeilen zu je 38 Zeichen erzeugen, also insgesamt 114 Zeichen.

Die überlangen Programmzeilen werden unter Ausnutzung aller Befehlsabkürzungen und anderer Eigenarten des ATARI-BASIC erzeugt. Sie erscheinen dann beim Ausdrucken im Listing als überlange Zeilen von mehr als 114 bzw. 12Ø Zeichen. Der Programmierer muss sich also bestens mit ATARI-BASIC auskennen, bevor er sich an eine erfolgreiche Eingabe wagt.

Die Aufteilung der überlangen Programmzeilen in 2 Zeilen hilft, wobei auf Abhängigkeiten innerhalb des Programms, insbesondere bei Sprungbefehlen, geachtet werden muss.

ATARI 6ØØXL
Der 6ØØXL in der Basisvariante mit 16 KiB Speicher erlaubt nur kurze Programme und ist hier nicht geeignet. Wird dazu noch ein DOS für die Kommunikation mit einer Diskettenstation geladen, vermindert sich der freie Speicher noch weiter.

Dezember 2Ø19, GoodByteXL

1. KAPITEL
- EINFÜHRUNG -

ABENTEUER sind außergewöhnliche Erlebnisse eines Menschen, die durch extreme Situationen und Gefahren gekennzeichnet Sind und sich unauslöschlich der Erinnerung des *Abenteurers* einprägen.
Sie üben einen großen Einfluss auf die Psyche des Menschen aus und faszinieren bereits seit undenklichen Zeiten auch an den Geschehnissen unbeteiligte Personen.
Diese Eigenschaft machte sich die Dichtung zunutze, und der Lauf der Zeit führte zur Entwicklung diverser literarischer Spielarten.
Waren es zunächst Minnesänger, die dem Volke oder auch gekrönten Häuptern von den Taten ferner Helden berichteten, so erwuchsen diese Spielmannsdichtungen im 12. Jahrhundert zu großangelegten Epen wie 'Lanzelot' oder Wolfram von Eschenbachs 'Parzival'.
'Don Quijote' und 'Münchhausen' begeisterten mit ihren Erlebnissen ebenfalls Tausende.
Nach dieser Zeit des Schelmenromans machten sich die Reiseerzählungen (Defoe, Cooper, Karl May) das Thema Abenteuer zunutze, bis die Entwicklung mit Kriminal- und Science-Fiction-Romanen ihren Abschluss fand.

ABENTEUER HEUTE

Wer kennt sie nicht, die auf der vorangegangenen Seite erwähnten Abenteuerromane, welche einem Lexikon unserer Tage entstammen könnten; wer hat sich in seiner Kindheit nicht von den Erlebnissen Old Shatterhands oder Robinson Crusoes mitreißen lassen ? - Wer hat sich nicht mit den Hauptpersonen identifiziert und sich vorgestellt, wie man sich wohl selbst in der gleichen Situation verhalten hätte ?

Doch vielleicht gehören Sie tatsächlich zu den wenigen, die aus Gründen der Bequemlichkeit niemals ein solches Buch gelesen haben. Warum auch, werden Sie dann fragen, ich kann es doch viel bequemer haben !
Don Quijote habe ich im Fernsehen gesehen, Lederstrumpf und Karl May dank der Programmdirektoren unserer Sender sogar öfter, warum sollte ich also meine Freizeit in irgendeiner Weise aktiv gestalten, wenn es auch anders geht, und außerdem: Lesen ist doch schon längst nicht mehr zeitgemäß.

Sicher, mit dem letzten Argument hätten Sie vermutlich recht, denn waren die Kinderstuben früher mit Puppen, Baukästen, Eisenbahnen oder Kinderbüchern gefüllt, so sind es heute ferngesteuerte Modelle, Experimentierkästen, Videorekorder, Telespiele und Heimcomputer. Unaufhaltsam ist die Technik auch hier auf dem Vormarsch, was bei richtiger Anwendung jedoch durchaus zu begrüßen ist.
Es mag wohl Computerbenutzer geben, die als einzige Anwendung realistische Raumschlachten und diverse andere Schieß- und Actionspiele kennen, doch irgendwann werden auch diese Anwender bemerken, daß der Spielablauf immer der gleiche ist. Denken ist nicht gefragt - nur der Bewegungsrhythmus der Steuertasten ändert sich von Spiel zu Spiel ein wenig.
Also unterzieht man den Softwaremarkt einer neuerlichen Prüfung - und stellt fest, daß die angloamerikanischen Computerbesitzer viel besser bedient werden; denn neben den

bekannten Arcade - Games haben sich in Amerika und England längst die Adventurespiele etabliert.
Adventures, zu deutsch Abenteuer, stellen neben Strategiespielen wie Schach oder Othello sicherlich die intelligenteste Anwendung eines Computers zu Spielzwecken dar.

Diese Programme ermöglichen es jedem Computerbesitzer, ohne große Risiken und Kosten die interessantesten Abenteuer zu erleben, wobei selbst das Unmögliche zum Normalzustand werden kann.
Stellen Sie sich einen Androiden vor, den Sie mit knappen Anweisungen durch alle Gefahren steuern, dem Sie Anweisungen geben, die er dann an Ihrer Stelle ausführt und die über Sieg oder Niederlage entscheiden.
Befehlen Sie ihm, daß er in eine bestimmte Richtung gehen soll, und er wird Ihnen daraufhin mitteilen, was er an seinem neuen Aufenthaltsort sieht, oder er wird Sie darüber aufklären, daß leider kein Weg in der gewünschten Richtung existiert.
Lassen Sie ihn Dinge, die er findet, näher untersuchen und befehlen Sie ihm, wenn Sie es für sinnvoll halten, diese Gegenstände zu nehmen.
Schließlich glauben Sie, eine im Hinblick auf das Spielziel sinnvolle Einsatzmöglichkeit für ein jedes dieser Objekte gefunden zu haben und lassen den Androiden entsprechend handeln; genießen Sie Ihren Triumph oder stellen Sie an Ihrem sich langsam entwickelnden Groll fest, wie sehr dieses Spiel Sie gepackt und der alltäglichen Welt entrissen hat.

Haben Sie ein Adventure in den Speicher Ihres Computers geladen, stehen Ihnen für Ihr Spiel nicht nur einige farbenfrohe Screens zur Verfügung, sondern Sie agieren in einer komplett ausstaffierten Welt, einer Welt, deren Geheimnisse und Rätsel nur darauf warten, von Ihnen entdeckt und gelöst zu werden.

Dabei wird ein gutes Adventure Sie wie im richtigen Leben agieren lassen, das heißt, Aktion und Reaktion stehen in gewohntem Zusammenhang, Lebenserfahrungen erweisen sich auch im Computerspiel als nützlich.

Sie werden entdecken, daß selbst eine alltägliche Handlung wie das Öffnen einer Tür Schwierigkeiten bereiten kann, die sich mit Ausgabe einer Meldung wie *Ich habe den passenden Schlüssel nicht !* ankündigen.
Selbst Magie kann erforderlich sein, um als Antwort *O.K. - Die Tür ist auf* zu erhalten, denn nur die Phantasie des Programmierers setzt die Grenzen zwischen Möglichem und Unmöglichem.

Andererseits kann das Adventurespiel aber auch als Lehrmittel eingesetzt werden. So ist beispielsweise ein Adventure, ausgestattet mit den Daten unseres Sonnensystems denkbar, welches es uns gestattet, den Weltraum ganz bequem von unserem Lieblingssessel aus zu erforschen.

Welche Methode könnte besser geeignet sein, um Kindern oder Wißbegierigen gleich welchen Alters Allgemeinwissen oder auch spezielle Informationen nahezubringen ?

Wie auch immer, ob ein Adventure nun als Lehrprogramm oder nur zur Unterhaltung geschrieben wurde, ein gutes Reaktionsvermögen ist bei weitem nicht mehr ausreichend, um ein Spiel zu bewältigen.
Sie werden schon einen scharfen Verstand und logisches Denkvermögen brauchen, um die zahlreichen Gefahren im Leben eines Abenteurers meistern zu können:

Wie wollen Sie einen Fisch fangen, damit Sie nicht verhungern müssen, wenn Ihnen nur die bloßen Hände zur Verfügung stehen?
Wie können sie den hungrigen Bären davon abhalten, Sie als Zwischenmahlzeit zu betrachten?
Wie wollen sie einen reißenden Fluss überqueren, der zudem noch von urweltlichen Raubtieren bewohnt wird?

Mit einigem Nachdenken lassen sich all diese Probleme lösen, Sie müssen nur die richtigen Ideen und entsprechend viel Phantasie haben:

Ihre Aufgabe besteht darin, in ein bestimmtes Haus, der Eingang ist durch ein Gittertor gesichert, zu gelangen. Bei genauem Hinsehen sehen Sie einige Zentimeter hinter dem Tor einen Schlüssel liegen, doch leider sind Ihre Arme zu kurz, um durch das Tor hindurch den Schlüssel greifen zu können.

Wie wollen Sie in das Haus hineingelangen?

Nun, überklettern des Tores ist nicht möglich, aber einige Schritte vor dem Haus waren Ihnen einige Bäume aufgefallen. Irgendwann im Laufe des Spieles haben Sie außerdem ein Kaugummi gefunden.

Ist der Groschen gefallen?

Wir haben auch einige Zeit für die Lösung dieses Problems gebraucht, vor allem, weil uns niemand so deutlich auf verwendbare Gegenstände hingewiesen hatte:

Man brauchte einfach nur zu dem Baum zu gehen und einen Ast abzubrechen. Dann ging man zum Eingang

> des Hauses zurück, kaute das Kaugummi gut durch, befestigte es an einem Ende des Stockes, schob dieses Ende durch das Gitter des Tores und berührte den Schlüssel, worauf dieser am Kaugummi kleben blieb; durch vorsichtiges Zurückziehen des Stockes gelangte man schließlich und endlich in den Besitz des Schlüssels.
> Welch ein Glück, dass dieser dann auch zu dem Tor paßte.

Eine elegante Lösung, nicht ganz einfach, aber vom Programmierer schon während der Entwicklungsarbeit genau so geplant. Ich sollte allerdings ergänzen, daß in der dem Spiel beigefügten Anleitung einige Spielerfahrung vorausgesetzt wurde.

Vielleicht hat obiges Beispiel Sie bereits in ©Abenteuerlaune© versetzt. Sollten sie die Kraft des geschriebenen Wortes zur Faszination jedoch noch nicht bemerkt haben, dann führen Sie sich bei nächster Gelegenheit doch einmal die *Unendliche Geschichte* zu Gemüte.

Welche Steigerungsmöglichkeiten werden sich nun durch die Implementierung guter Romane in Computersysteme ergeben ?

Vielleicht wird es sich eines Tages als notwendig erweisen, die Erläuterung des Begriffes ©Abenteuer© in unseren Lexika zu ergänzen:

> Im 2Ø. Jahrhundert ermöglichte die fortgeschrittene Computertechnik den Abenteuerroman zum Mitspielen. Als Meilenstein gilt Scott Adams "Adventureland"; es folgten zahlreiche weitere sogenannte Adventures, die bis zu perfekten Weltsimulationen entwickelt wurden. Zahlreiche Buchautoren mit bekannten Namen (z. B. Michael Crighton) erkannten rechtzeitig die neuen Möglichkeiten und schufen mit ihren Werken für viele Menschen einen Ausgleich zur Alltagswelt.

GESCHICHTE UND ENTWICKLUNG DER ADVENTURES

Nachdem aus der ersten Rechenmaschine ein programmgesteuertes Rechengehirn geworden war, setzten in Programmiererkreisen die unterschiedlichsten Bemühungen ein, den Computern menschliches Denken und menschliche Verhaltensweisen beizubringen. Im Jahre 1966 gelang es schließlich Joseph Weizenbaum vom Massachusetts Institute of Technology mit seinem Programm ELIZA, nicht nur die Fachleute, sondern auch die Öffentlichkeit zu interessieren.
Eliza, in abgemagerten Versionen heute auch für jeden Microcomputer erhältlich, simuliert einen Psychiater und ahmt dessen typische Gesprächstechnik nach. Bei Versuchen mit Testpersonen zeigten sich diese nach der jeweiligen Sitzung sehr überrascht, einer Maschine ihre persönlichsten Probleme mitgeteilt zu haben.

Die weitere Entwicklung der Künstlichen Intelligenz brachte den Fachwissenschaftlern ein auf einer PDP-1Ø implementiertes Programm: ADVENTURE - Colossal Cave. Zur Abwechslung durften die gelehrten Herren auf einem anderen Gebiet forschen, nämlich nach Schätzen in einer düsteren, von Magie beherrschten Welt. Colossal Cave erfreute sich in Fachkreisen einer recht großen Beliebtheit und wurde schließlich im Sommer 1979 dann auch einer größeren Öffentlichkeit zugänglich gemacht. Die heute wohl jedermann bekannte Firma MICROSOFT veröffentlichte *Microsoft©s Adventure*, eine Diskettenversion dieses Uradventures für den damals in Amerika populärsten Microcomputer, den TRS-8Ø.

Salonfähig gemacht wurden die Adventures jedoch schon im Jahre 1978 von einem jungen Amerikaner namens Scott Adams, welcher mit seinem ADVENTURELAND den Grundstock für den Newcomer ADVENTURE INTERNATIONAL legte.

Auch in diesem Spiel geht es darum, in einer mystischen Welt diverse Schätze zu finden, was natürlich durch Drachen, Labyrinthe, Lavaströme und sonstige Eigen- und Unarten erschwert wird. Dabei fängt alles ganz harmlos mitten im Walde an. Doch wenn man sich erst einmal orientiert hat und weiß, in welche Richtung man sich wenden muß, und dann auch noch den Eingang in das unterirdische Reich findet, dann ...

Die Nachfrage nach ADVENTURELAND war riesig, wohl auch deshalb, weil es noch keine Spiele in Spielhallenqualität gab, und so wurde wegen des großen Erfolges aus diesem ADVENTURELAND rasch eine Reihe von 12 Adventures, wobei durch geschickte Wahl der Thematik das Interesse der Kunden wachgehalten wurde.
Wollten Sie sich nicht auch schon immer einmal mit dem Grafen Dracula anlegen - in *The Count* können Sie es. Oder möchten Sie lieber einen Saboteur in einem Atomkraftwerk finden - kein Problem, *Mission Impossible* macht©s möglich.

Zu dieser Zeit erschien auf dem amerikanischen Softwaremarkt als Mitreiter auf der gleichen Welle ein Programmtyp, der als ***©INTERACTIVE FICTION©*** bezeichnet wurde. Diese Programme sollten einen neuen Literaturtyp darstellen, sollten, wie der Name sagt, Bücher zum Mitwirken sein. Zunächst ist der Spieler nur Leser, er wird nach Programmstart sehr genau in die Handlung eingeführt. Zahlreiche Seiten flimmern über den Bildschirm, die dem Leser Auskunft über Vorgeschichte, augenblickliche Situation, Hintergründe usw. geben, und die sich eben wie ein Buch lesen lassen.

So beginnt *Local Call For Death* wie eine Kriminalgeschichte: Drei Männer treffen sich in ihrem Clubhaus, im Verlauf des Abends wird einer von ihnen ans Telefon gerufen. Es ist der Neffe, welcher den Onkel, sichtlich erregt, um Geld bittet. Der Onkel versagt ihm jedoch die Unterstützung, weiß er doch aus Erfahrung, daß

das Geld auf der Rennbahn verwettet werden wird. Am nächsten Morgen erfährt er vom Selbstmord seines Neffen.

An dieser Stelle übernimmt nun der Spieler. Er stellt ab jetzt eine der Hauptpersonen dar, und versucht im direkten Gespräch mit den Mitwirkenden den vermeintlichen Selbstmord aufzuklären.

Typisch für diese Programme ist die klare Fixierung auf ein einziges Ziel, wie hier die Aufklärung eines Mordes.
Um dieses Ziel zu erreichen, müssen wir als Spieler uns weniger irgendwelcher Gegenstände als geschickter Fragen bedienen. Was ist wohl interessanter, in einem Buch ein Gespräch zu lesen, in einem Film ein Verhör zu verfolgen, oder das Erfolgserlebnis zu haben, durch eigenes Kombinationsvermögen einen Verbrecher überführt zu haben ?

Gerade diese Fähigkeit zu Dialogen macht den Reiz von Interactive Fiction aus, ein Musterbeispiel ist das ebenfalls bei Adventure International erschienene *Encounter in the Park*.
In diesem Spiel begegnen wir während des Morgenspazierganges unserer Traumfrau, und es wird unser erklärtes Ziel, eben dieses Mädchen zu verführen und dazu zu bringen, uns ihr Ja - Wort zu geben. Der Programmierer hat dem Spieler zur Erfüllung dieser Aufgabe ein breites Spektrum an Überredungskünsten offengelassen, und es macht wirklich großen Spaß, zu erleben, wie diese Dame auf gewisse Vorschläge reagiert.
Das Schlimmste, was uns bei allzu unseriösen Vorschlägen passieren kann, ist, daß sie sich entrüstet von uns abwendet, und wir Mitteilung darüber erhalten, daß wir aus lauter Enttäuschung den Rest unseres Lebens im Kloster verbracht haben.

Wie bereits gesagt, war das Interesse an all diesen Textadventures sehr groß, immer mehr Firmen veröffentlichten demzufolge mehr oder weniger gute Kopien

dieser Spiele. Befleißigt durch den zunehmenden Druck der Konkurrenz, ließen sich daher einige Softwarehersteller neue Generationen von Adventurespielen einfallen.
Es erschienen die ersten ***LABYRINTH - ADVENTURES***, welche den Spieler ihrer Art entsprechend vor die Aufgabe stellten, lebend aus einem Gebäude zu entkommen. Fast immer endete ein Spiel damit, daß dieser sich hoffnungslos in dem perspektivisch auf dem Bildschirm dargestellten Labyrinth verirrte.

Nächste Stufe der Entwicklung waren die ***QUEST - ADVENTURES***, die dann meistens auch gleich als Real - Time Adventures implementiert wurden. Bei diesem Sproß der großen Adventurefamilie wird der Spieler einem zusätzlichen Streßeffekt unterworfen, denn sollte er beim Erscheinen irgendwelcher Räuber oder anderer Unholde nicht sofort F für Fight eingeben und danach laufend irgendwelche Tasten oder gar Tastenkombinationen zu dem Zwecke drücken, sein Schwert in genau definierte Richtungen zu schwingen, endete das Spiel bereits sehr frühzeitig wegen zu starker Beschädigungen der Hauptperson.
Leider ergibt sich aus der Konzeption dieser Spiele eine starke Einschränkung des Handlungsspielraumes des Spielers, denn es bleiben diesem nur einige wenige Aktionen wie Kämpfen, Verhandeln, Kaufen oder Flüchten, die ihn per Menü, daher QUESTion, angeboten werden.

Die nächste Entwicklungsstufe der Adventurespiele wurde dank des Preisverfalls für Grafikprozessoren und RAM-Bausteine möglich, denn nun war es für die Computerhersteller nicht mehr schwierig, dem Heimanwender bei preisgünstigen Computern eine grafische Leistung zur Verfügung zu stellen, die früher nur auf vielfach größeren und somit auch teureren Anlagen möglich war.

So begann im Jahre 1982 (für Apple - Benutzer etwas früher) mit dem Vertrieb des ersten ***GRAFIKADVENTURES*** ein neuer Aufschwung für diese Spiele. Denn waren sie von vielen

Computerbesitzern als reine Textspiele bislang verpönt worden, so waren sie nun wegen der vielen bunten Bilder auf einmal interessant; darüber hinaus erwiesen sie sich als hervorragend dazu geeignet, der computerunkundigen Verwandtschaft auf die Frage ©*Was kann denn dein Computer ?*© zumindest ein erstauntes ©*Oh, ist ja schön !*© zu entlocken.

An Spielwert und Ziel hatte sich natürlich überhaupt nichts geändert.
Plötzlich waren jedoch weitere Käuferschichten erreichbar; ein Grund, viele jahrelang bekannte Adventures im neuen Gewand auf den Markt zu werfen: Die ausführlichen Beschreibungen waren verschwunden, statt dessen waren die augenblicklichen Gegebenheiten auf dem Fernsehschirm zu sehen.

Ob die Programme dadurch besser oder einfacher spielbar geworden waren, sei dahingestellt, ich persönlich ziehe es jedoch vor, durch eine Mitteilung wie

Ich bin im Wald. Um mich herum stehen lauter Bäume. Zwischen zwei Eichen sehe ich ein Erdloch.

auf wichtige Dinge hingewiesen zu werden, als im Unterschied dazu nur einige Bäume zu sehen; diese halte ich dann verfrüht für ein ganz normales Stück Wald, weshalb ich auch schnell weitergehe, die Bäume nicht weiter untersuche und dadurch das Erdloch, einen kleinen schwarzen Flecken auf dem Monitor, natürlich übersehe. Selbstverständlich liegt ausgerechnet darin ein Teil des Schatzes oder aber ein Gegenstand, ohne den eine erfolgreiche Beendigung des Spieles nicht möglich ist.

2. KAPITEL
- DAS KONZEPT -

In den folgenden Kapiteln dieses Buches werden wir gemeinsam Adventurespiele entwickeln. Selbstverständlich wünschen wir für unser Werk ein möglichst professionelles Aussehen, daher werden wir nicht darauf verzichten können, Eigenschaften und spezielle Features auf dem Markt befindlicher Programme zu analysieren.
Anschließend können wir uns dann Gedanken darüber machen, wie wir die einzelnen Funktionen in BASIC programmieren.
Dabei wollen wir darauf achten, daß diese einzelnen Routinen dermaßen aufgebaut sind, daß sie nicht nur für ein bestimmtes Adventure brauchbar, sondern ohne Änderungen für den Einsatz in allen unseren Abenteuerprogrammen geeignet sind.

DIE AUFMACHUNG

Es wurde bereits herausgestellt, daß ein Adventure den Spieler in eine andere Welt versetzen will, in eine Welt, die das Unmögliche möglich macht. Er soll hier Abenteuer erleben, muß sich also bewegen und handeln können, muß über die Resultate seiner Bemühungen informiert werden, damit er weitere Handlungen vornehmen kann.
Sinnvolles und zielgerichtetes Handeln ist dem Menschen jedoch erst nach Gebrauch seiner Sinne möglich. Die Informationen, die der Abenteurer im wirklichen Leben mit seinen Augen, Ohren und seinem Tastsinn aufnimmt, müssen dem Spieler daher in geeigneter Form angeboten werden.
Da das menschliche Gehirn bis heute nicht direkt an einen Computer angeschlossen werden kann, müssen uns Antworten auf die Fragen *Wo bin ich ? Was sehe ich ? Wohin kann ich gehen ? Was fühle ich ?* und *Was sehe ich ?* auf andere Art und Weise zugänglich gemacht werden, was hier in Gestalt kurzer, aber vollständiger Sätze geschieht.

Die Erfassung der Umwelt wird allerdings in den seltensten Fällen bewußt ablaufen, Rauminhalte werden ohne besonderen

Wunsch innerhalb von Sekundenbruchteilen registriert, daher kann auch dem Adventurespieler nicht zugemutet werden, zunächst umfangreiche Abhandlungen über den gerade betretenen Raum zu studieren.
Selbst wenn es nur fünf oder zehn Sekunden wären, die er mit dem Lesen von mehreren Zeilen Beschreibungen zubrächte, würde die dazu erforderliche Konzentration den Spielfluß hemmen und es dem Spieler unmöglich machen, sich mit der Spielfigur zu identifizieren und sich der Illusion einer Scheinwelt hinzugeben.

Um dieses Problem zu umgehen, beschränkt man die Textausgabe auf ein akzeptables Minimum an Informationen und präsentiert diese, sofern es sich nicht um Grafikadventures handelt, in zwei übersichtlichen Zonen auf dem Monitor, von denen jede eine genau definierte Funktion hat.

So wird uns die obere Bildhälfte darüber Auskunft geben, wo wir uns gerade befinden und was wir sehen, die untere Hälfte ist hingegen für die Kommunikation mit der für uns handelnden Hauptperson des Adventures bestimmt. Hier geben wir unsere Anweisungen ein und hier erfahren wir, was auf unsere Eingaben hin geschieht.

Ein typisches Schirmbild könnte etwa so aussehen:

Ich bin in einem düsteren Wald.
Ich sehe viele alte Bäume.

Ich kann nach Norden, Osten, Westen.

Was soll ich tun ?

Für den Fall, daß wir beabsichtigen, in dem Spiel möglichst rasch weiterzukommen, wäre es falsch, ziellos in eine der drei Himmelsrichtungen weiterzumarschieren.
Das Nächstliegendste, was ein Adventurespieler in jeder Situation, die für ihn neu ist, tun sollte, ist, alles auf das genaueste zu untersuchen. So wären mögliche Antworten auf *Untersuche Baum* etwa: *Es handelt sich um alte Eichen.* oder *Auch hier greift das Baumsterben um sich.*
Diese Antworten helfen uns zwar nicht weiter, zeigen sie uns doch scheinbar nur, daß es doch nicht immer so wichtig ist, alles zu untersuchen, doch woher soll man die Antwort vorher wissen ? Denn auch eine Antwort wie *In einem Stamm sehe ich eine Öffnung.* wäre möglich gewesen. Vielleicht handelt es sich um einen alten Spechtbau, in welchem sich ein Teil des gesuchten Schatzes befindet; *Untersuche Öffnung* wird es an den Tag bringen.
Hätten wir keine Anhaltspunkte gefunden, auch auf *Untersuche Wald* hin - es könnte ja ein Zauberwald sein - nicht, müßten wir uns entscheiden, wohin wir uns nun sinnvollerweise begeben wollen.
Wie würden Sie sich in solch einer Situation verhalten, ohne Karte und Kompaß, hilflos und allein, weitab aller Wege ?
Vermutlich würden Sie auf einen Baum klettern, um zu sehen, was sich in der näheren und weiteren Umgebung befindet.

Ihre Eingabe sollte also lauten: *Besteige Baum*, und wenn Sie nicht eine Mitteilung in der Art *Dazu bin ich nicht sportlich genug !* erhalten, wird sich das Bild auf dem Monitor ändern:

Ich bin in der Krone einer alten Eiche.
Ich sehe im Norden ein Gebirge,
im Osten einen See,
im Westen steigt Rauch empor.

Ich kann nach unten.

BESTEIGE BAUM

Okay !

Was soll ich tun ?

Damit sieht die ganze Angelegenheit doch schon viel besser aus, denn nach Eingabe von *U* für Unten haben wir nur noch das Problem, zu entscheiden, welche Gegend wir zuerst besuchen wollen.

Die Mitteilung *Okay !* soll uns darüber informieren, daß die Eingabe verstanden und ausgeführt worden ist. Wäre dem nicht so, hätten wir beispielsweise *Ich verstehe das Verb nicht !* erhalten, womit uns das Programm sicherlich zu einer anderen Ausdrucksweise veranlassen würde. Gleiches gilt selbstverständlich für das Objekt unserer Eingabe.

Denkbar sind allerdings auch noch zwei weitere Antworten. Zum einen werden wir während eines Spieles oft die Mitteilung *I must be stupid, but ... - Ich verstehe nicht, was Du meinst !* erhalten, womit das Programm uns mitteilen will, daß es unsere Eingabe für recht sinnlos hält, was bei *Töte Baum* zweifellos der Fall wäre. Jedenfalls ist diese Aktion dann nicht im Spiel vorgesehen.
Ähnlich verhält es sich mit der zweiten Standardmeldung *You can©t do that ... yet. - Das geht im Moment nicht.* signalisiert uns, daß die benutzten Worte zwar zum programmierten Wortschatz gehören, der Befehl im Moment jedoch nicht sinnvoll ist: *Nimm Schlüssel* wird vom Adventureprogramm verstanden werden, wenn ein Schlüssel im Adventure auftritt, wird jedoch sinnlos, solange sich dieser Schlüssel nicht im gleichen Raum wie der Spieler befindet.

Dennoch dürfen wir aufatmen, denn sollten wir diese Antwort erhalten, sind wir auf dem richtigen Weg; es sind nur noch nicht alle erforderlichen Bedingungen erfüllt, um den Befehl auszuführen. So wird sich eine verschlossene Tür öffnen lassen, wenn wir uns ein für diesen Vorgang geeignetes Instrument beschafft haben. War die Antwort auf *Öffne Tür* allerdings *Ich verstehe nicht, was Du meinst.* dann wären alle weiteren Bemühungen zwecklos.

VORÜBERLEGUNGEN

Der generelle Aufbau der Adventurespiele ist uns nun bekannt. Wir haben eine Vorstellung davon, wie unser Adventure sich auf dem Monitor zu präsentieren hat und welche Informationen bereitgestellt werden müssen. Im folgenden wollen wir uns näher mit den einzelnen Elementen unseres Adventuresystems beschäftigen und eine Konzeption ausarbeiten, die im nächsten Kapitel realisiert wird.

Räume im Adventure:

Die Adventurewelt setzt sich aus den Räumen des jeweiligen Spieles zusammen, d. h. jeder durch den im Adventure herumwandernden Spieler erreichbare Ort wird als Raum bezeichnet. Dabei ist dessen Größe völlig unbedeutend, es kann, wie in unseren bisherigen Beispielen, ein Wald, bzw. eine Baumkrone gemeint sein, es kann sich jedoch auch um das Innere eines Autos oder Kleiderschranks handeln.
Die einzelnen Räume unterscheiden sich für den Spieler durch ihre Beschreibungen und ihre geografische Anordnung; programmtechnisch unterscheiden sie sich, wie wir noch sehen werden, durch ihre Raumnummer.

Alle diese Räume sind für den Spieler erreichbar, müssen also in geeigneter Weise miteinander verbunden werden. In jedem Adventure ist die Bewegung in die vier Himmelsrichtungen Norden, Süden, Westen und Osten möglich, einige erlauben zusätzlich Bewegungen in der Vertikalen.
Selbstverständlich darf es jedoch nicht geschehen, daß der Spieler Raum 5 in westlicher Richtung verläßt, Raum 6 dabei von Osten her betritt, und wenn er noch einmal in Raum 5 will, keinen Weg in Richtung Osten findet, oder, schlimmer noch, wenn er durch eine Bewegung nach unten wieder in Raum 5 gelangt.
Um der Kritik von erfahrenen Abenteurern vorzubeugen, Ausnahmen bestätigen auch hier die Regel, denn was wäre ein Adventure ohne ein magisches Labyrinth.

Objekte

Typisch für jeden Raum sind weiterhin die in ihm befindlichen Objekte. Wir müssen alle Gegenstände in den Räumen unterbringen, oder, korrekter ausgedrückt, jedem Objekt einen Raum zuordnen. Bei dieser Zuordnung vermeiden wir automatisch, daß irgendwelche Objekte doppelt existieren.

Ein weiteres Problem stellt sich uns mit der Unterbringung derjenigen Objekte, die bei Spielstart gar nicht im Adventure vorhanden sein dürfen. - Wo bleiben sie bis zu ihrem Auftreten ?

Angenommen, unser Held betritt einen Raum, und als sichtbares Objekt wird nur eine alte, verschlossene Holzkiste ausgegeben.
Auf die Eingabe des Spielers *Öffne Kiste* verlangen sämtliche Regeln der Logik, daß die alte, verschlossene Kiste vom Bildschirm verschwindet, dafür sollte jedoch eine

alte, geöffnete Kiste auftauchen, die womöglich mit Silbermünzen gefüllt ist.

Nun, die Lösung dieser Probleme wird uns keine allzu großen Schwierigkeiten machen. Wir werden in unserem Adventure einen zusätzlichen Raum, den Lagerraum, installieren und in diesem alle zur Zeit nicht benutzten Objekte lagern. Wenn wir keine Verbindungen zu diesem Raum hin programmieren, kann der Spieler ihn auch nie betreten und somit auf unerwünschte Weise mit den betreffenden Gegenständen in Kontakt kommen.

Im nächsten Kapitel werden wir sehen, wie einfach der Lagerraum und auch die gesamte Adventurewelt zu programmieren sind; weiterhin werden wir im Verlauf des Buches die Notwendigkeit weiterer besonderer Räume einsehen.

Eingaben

Wie bereits aus dem voranstehenden Text hervorgegangen ist, bestehen die Eingaben des Spielers meist aus einem Verb und einem Objekt. Unser Programm wird daher zunächst testen müssen, ob eine Eingabe erlaubt ist, wozu eine Trennung in Verb und Objekt erforderlich sein wird. Sollte der Spieler sich verschrieben haben, oder das benutzte Verb nicht vorgesehen sein, muß eine entsprechende Meldung gemacht werden. Findet unser Adventure das Eingabeverb innerhalb seines Wortschatzes wieder, muß das Objekt auf gleiche Weise überprüft werden.
Sind beide Worte definiert, kann das Programm in vorgeschriebener Weise reagieren.

Bei der Erstellung dieses Wortschatzes muß der Programmierer übrigens besondere Sorgfalt walten lassen, er

muß alle möglichen Eingaben, die irgendein Spieler machen könnte, vorhersehen.
Wie schwierig gerade dieses Unterfangen werden kann, bewies mir kürzlich einer meiner Bekannten, der nie zuvor ein Adventure gespielt hatte, und der, nachdem er das Titelbild zusammen mit den notwendigsten Instruktionen studiert hatte, alle Probleme auf einen Schlag mit *Finde Schatz* lösen wollte, einer Eingabe, mit der ich als Programmierer natürlich nicht gerechnet hatte.

Nebenbei wird auch die Einbringung von Synonymen erforderlich werden, denn sollte der Spieler allzu lange damit beschäftigt sein, dem Adventure seine Wünsche klarzumachen, wird er schnell das Interesse verlieren.

Gerade für uns Deutsche kann die Wahl der Worte jedoch zum Problem werden, denn wo der Amerikaner sich mit *Climb Tree, Shoot Gun, Drop Box, Look* leicht verständlich machen kann, sieht es für uns nicht ganz so eindeutig aus: *Kletter Baum, Schieß Gewehr, Leg Kiste, Sieh.*

Mitteilungen

Die erste Ausgabe auf dem Bildschirm, die nicht mehr zu der Beschreibung des Raumes gerechnet werden kann, informiert den Spieler über die Himmelsrichtungen, in deren Richtung er den Ort des Geschehens wieder verlassen kann.

Weiterhin müssen die Ausführung einer jeweiligen Aktion, wie auch eine eventuell darauf hin stattfindende Reaktion, dem Spieler deutlich gemacht werden. Im einfachsten Fall geschieht dies durch *O.K.*, meist werden jedoch zusätzliche Mitteilungen ausgegeben, wobei die unterschiedlichsten Eingaben oft die gleiche Antwort erfordern werden, Antworten, die aus eben diesem Grunde in einer Liste geführt, und standardisiert werden sollten. Beispielsweise

werden *So stark bin ich nicht.* oder *Das hat doch wohl keinen Sinn!* und *Ich verstehe nicht, was Du meinst !* Recht häufig auftauchen.

FUNKTIONEN DER ADVENTURES

In einem durchschnittlichen Adventure finden wir 3Ø - 5Ø verschiedene Räumlichkeiten, in jedem Raum einen oder mehrere Gegenstände. Diese Gegenstände können wiederum irgendwelche Sachen enthalten, und da man im Adventure nie genau weiß, was ein paar Züge weiter unbedingt gebraucht wird, neigen viele Adventurespieler dazu, alles mitzunehmen, was ihnen zwischen die Finger kommt und nicht zu schwer zum Tragen ist.
Den Überblick zu behalten, wird dabei recht schwierig, denn da wir alle bequem geworden sind, machen wir uns natürlich keine Notizen.
Diese Erkenntnis kam den Programmierern glücklicherweise recht früh, so daß heute jedes Adventure nach Eingabe von *INVENTUR* alle Gegenstände, die wir mit uns führen, auf dem Bildschirm ausdruckt.

Falls es sich um ein freundliches oder speziell für Anfänger geschriebenes Adventure handelt, können wir uns in verfahrenen Situationen Tipps geben lassen, denn dann ist der Befehl *HELP - HILFE* implementiert.

Im Gegensatz zur Standardeingabe aus Verb und Objekt handelt es sich hier, wie übrigens auch bei Inventur, um einen sogenannten Ein - Wort Befehl, welcher in diesem Fall jedoch nicht einmal eine Aktion einleitet.
Doch was kann dieses eine Wort nicht alles bewirken:

Ich würde alles untersuchen zeigt uns unsere Nachlässigkeit, die Mitteilung *Eine feenhafte Gestalt*

erscheint und schreibt den Satz ©Sesam öffne dich© in den Sand wird uns hingegen in einer verfahrenen Situation zu einem Freudensprung veranlassen.
Schlechter sieht es aus, wenn wir für die Hilfe bezahlen sollen: *Ein Troll erscheint und sagt, daß uns für eine Hilfe 1Ø Punkte abgezogen werden. Er will wissen, ob wir Hilfe benötigen ?*
Dunkle Wolken werden vermutlich in den Fällen am Horizont auftauchen, wenn wir auf die Eingabe von Hilfe folgende Nachricht erhalten: ©*Hilfe*© *ist in diesem Adventure nicht vorgesehen !* oder wenn als Antwort nur die allgemeine Spielanweisung wiederholt wird.

Sicherlich mag es ärgerlich sein, wenn man an einer gewissen Stelle trotz aller Mühen nicht weiterkommt, jedoch sollte niemand die letzten zwei Beispiele als einen Akt der Boshaftigkeit des Programmierers ansehen; wäre die Hilfsfunktion durchgehend gewährleistet, würden einige ganz Schlaue das Adventure in 3Ø Minuten lösen, dabei aber niemals die Faszination eines solchen Programmes kennenlernen.
Einige Programme achten daher auf den Gebrauch dieser Funktion und verweigern bei allzu häufiger Anwendung schließlich jede Hilfe oder lenken den Spieler gar auf falsche Fährten.

3. KAPITEL
- DIE VERWIRKLICHUNG -

HINWEISE ZU DEN LISTINGS

Bevor wir mit der Programmierung beginnen, möchte ich Ihnen noch einige Hinweise eher technischer Natur geben, die mit dazu beitragen sollen, daß Ihnen der Spaß an der Arbeit mit diesem Buch nicht durch irgendwelche auftretenden Fehler vergällt wird.

Sicherlich werden die folgenden Programmzeilen nicht die ersten sein, die Sie mühsam über die Tastatur eingeben, und Sie werden daher bereits die Erfahrung gemacht haben, daß sich immer wieder Tippfehler einschleichen.

Geradezu prädestiniert für Fehler dieser Art sind insbesondere DATA - Zeilen, besonders, wenn diese nur Ziffern enthalten wie es auch hier der Fall sein wird.
Weiterhin ist es in einigen Zeilen unerläßlich, diese *genau* so einzugeben, wie sie abgedruckt sind. Dies gilt insbesondere für die Anzahl der Blanks (Leerzeichen) in den Strings (Zeichenketten).

Um Ihnen eine wirksame *Kontrollmöglichkeit* zu geben, sind alle Listings in diesem Buch so abgedruckt, wie Sie sie *nach Eingabe von LIST* auf Ihrem Bildschirm sehen müßten.

Überprüfen Sie daher, wenn Sie den Eindruck gewinnen, daß irgendetwas wohl nicht wie vorgesehen funktioniert, zunächst nur jeweils die ganz rechts in einer Zeile stehenden Buchstaben, stimmen sie nicht überein, haben Sie mit allergrößter Wahrscheinlichkeit den Fehler bereits gefunden.

Darüber hinaus sollten Sie nur die vorgegebenen Zeilennummern verwenden, damit die Funktionsfähigkeit Ihres Programmes erhalten bleibt.
Denn mit der weiteren Entwicklung werden einige Programmzeilen ausgetauscht sowie zusätzliche Routinen eingefügt.

ATARI 6ØØ XL - Besitzer beachten bitte außerdem die Hinweise im Anhang !

DIE SPIELIDEE

Damit haben wir das Mindestmaß an grauer Theorie bewältigt, alles, was uns noch fehlt, ist eine Geschichte, die wir als Abenteuerspiel realisieren wollen. Dann können wir unseren Atari einschalten und mit der Praxis beginnen. Vielleicht haben Sie bereits eine feste Vorstellung von Ihrem ersten Adventure, wenn nicht, lassen Sie sich doch durch folgende Vorschläge inspirieren.

1. DAS SPUKHAUS

Sie erhalten den letzten Brief von Ihrer Tante, ein Schreiben, in dem sie Ihnen ihr altes hochherrschaftliches Haus vererbt. Stutzig macht Sie jedoch der Hinweis, daß Sie sich vorsichtig verhalten sollen, denn sonst könne es Ihnen wie Ihrem Onkel ergehen. Sie lassen sich durch solcherlei Gerede nicht davon abhalten, das Haus aufzusuchen. Sogleich stellen Sie allerlei Ungereimtheiten fest, finden eine geheime Bibliothek mit Büchern über Geisterbeschwörung, entdecken im Keller eine Opferstätte. Zahllose Geister machen Ihnen das Leben schwer, bis Sie schließlich entdecken, daß einer Ihrer Vorfahren den Sektenführer zum Ausbau der eigenen Macht ermordet hatte und zur Strafe lebendig eingemauert worden war. Sie sollen ihm nun zur ewigen Ruhe verhelfen.

2. DAS VERMÄCHTNIS DES ALTEN

Ihr Freund und Nachbar, ein berühmter Wissenschaftler, ist unter mysteriösen Umständen ums Leben gekommen. Sie erhalten eine Notiz, in der er Sie bittet, sein Lebenswerk zu vollenden. Nachdem Sie es geschafft haben, sein bislang geheimes Labor zu betreten, ist es Ihre Aufgabe,

seinen Supercomputer in Betrieb zu nehmen. Dieser gibt Ihnen dann weitere Hinweise.

3. STAR ODYSSEY

Ihr Raumschiff wird durch einen kosmischen Sturm so stark beschädigt, daß Sie auf einer unbemannten Station landen müssen, um Ersatzteile zu beschaffen, damit Sie zur Erde zurückkehren können.

4. GOLDRAUSCH

Sie treffen in der Wildnis auf einen tödlich verletzten alten Mann. Dieser macht Ihnen Mitteilung über seine Goldmine. Er zeigt Ihnen einige Goldstücke und berichtet Ihnen, daß er an sieben verschiedenen Stellen auf dem Minengelände weiteres Gold versteckt hätte; da er es nicht mehr benötige, könnten Sie es sich holen.

Diesen zuletzt gemachten Vorschlag, ein Adventure des Typs Schatzsuche, möchte ich als Grundlage für die nächsten Kapitel dieses Buches nehmen, und gemeinsam mit Ihnen ausarbeiten. Selbstverständlich steht es Ihnen frei, eine eigene Idee zu realisieren, jedoch werde ich bei allen noch vorzustellenden Erweiterungen immer wieder Bezug auf *Goldrausch* nehmen.

DIE GESTALTUNG DER WELT

Mit der Wahl des Themas steht der grobe Ablauf des Adventures bereits fest. Im folgenden werden wir einzelne Strukturen ausarbeiten und diese immer weiter verfeinern, bis eine bis ins Kleinste geplante Welt geschaffen worden ist.

Genauso wollen wir auch bei der Programmierung verfahren; Schritt für Schritt werden wir unser Adventure aufbauen und uns immer wieder von der korrekten Funktion der einzelnen Routinen überzeugen.

DIE KARTE

Am Anfang steht zunächst das Nichts. Doch werden wir die für unser Spiel so dringend benötigte Welt im folgenden nach unseren Vorstellungen schaffen. Was unsere Welt bieten muß, legt das Thema fest. Goldrausch wird vermutlich in und um ein Bergwerk stattfinden.

Stellen wir uns die Mine einmal vor: ein zerklüfteter Abhang voller Geröll; ein Werkzeugschuppen; Holzrinnen, die das für die Goldwaschung benötigte Wasser eines Gebirgsbaches heranführen; morsche, trockene Holzbalken, die den düsteren Eingang säumen; dunkle Gänge voller Gefahren.

Sicherlich werden uns ohne große Schwierigkeiten noch zahlreiche ähnliche Bilder einfallen, die uns eine für die vorgesehene Handlung ausreichende Menge Räume finden lassen werden.

Unser konkretes Beispiel möchte ich zunächst auf nur sechs Räume beschränken. Beginnen wollen wir mit einer Wanderung

durch einen Wald, bis wir zur Mine gelangen, wo unsere Aufgabe darin bestehen wird, die Schätze zu suchen.

Damit ist der vorläufige Beginn von Goldrausch festgelegt. Um jedoch über Sieg oder Niederlage des Spielers entscheiden zu können, müssen wir ebenfalls ein erfolgreiches Spielende festlegen, mit weiterem Fortschreiten unseres Projektes natürlich auch mehrere verlustbringende Situationen.

So können wir uns mit dem Finden aller Schätze begnügen, können dem Abenteurer aber auch die Aufgabe stellen, den gefundenen Schatz an einem sicheren Ort zu deponieren. Zunächst soll das Auffinden des Goldes jedoch ausreichend sein.

Aufgabe und Ziel stehen fest, ebenso zwei wesentliche Handlungsorte. Da ist zum einen der Wald, in dem wir starten und den wir aus zwei Räumen aufbauen werden, und zum anderen die Mine. Allerdings ist uns klar, daß wir in dieser kleinen Welt keine Handlung ablaufen lassen können. Denn wir brauchen Platz, um einige Gegenstände verstecken und dem Spieler einige Fallen stellen zu können.

Normalerweise wird ein Bergwerk wohl auch kaum zwischen den Wurzeln eines Baumes beginnen, wir müssen geeignete Übergänge der Landschaft schaffen, um das Spiel realistisch zu gestalten.

Drei weitere Räume stellen uns zur Ausarbeitung der ersten Version unseres Adventures genügend Platz zur Verfügung, wir können nun eine Liste aller auftretenden Orte formulieren:

```
im Wald
im Wald
durch einen Felshang begrenzter Waldrand
```

eine Waldlichtung
Waldlichtung am Berghang
Eingang ins Bergwerk

Der nächste Schritt zum fertigen Programm ist zweckmäßigerweise die Anfertigung einer Karte, in der alle Orte als Rechtecke dargestellt werden. Diese Rechtecke werden durchnummeriert und mit ihrer Raumbeschreibung versehen, wobei wir bereits auf eine passende Formulierung achten und auch nach Möglichkeit die Zeilenlänge von 38 Zeichen berücksichtigen, die wir auf unserem Atari verwenden wollen. So muß sich unsere Beschreibung reibungslos an den einleitenden Text *Ich bin* anfügen.

Welche Nummer der einzelne Raum erhält, spielt keine Rolle. Wir müssen nur mit eins beginnen und fortlaufend weiterzählen, wodurch wir auch doppelte Nummerierungen verhindern. Anschließend verbinden wir die Räume untereinander und notieren an jeder Begrenzung die entsprechende Himmelsrichtung.

Aus dieser Karte läßt sich nun ohne weiteres ablesen, daß, wenn der Spieler sich beispielsweise in Raum 3 befindet, folgende Mitteilungen auf dem Monitor ausgegeben werden müssen:

Ich bin im Wald. Vor mir steigt ein steiler Felshang auf.

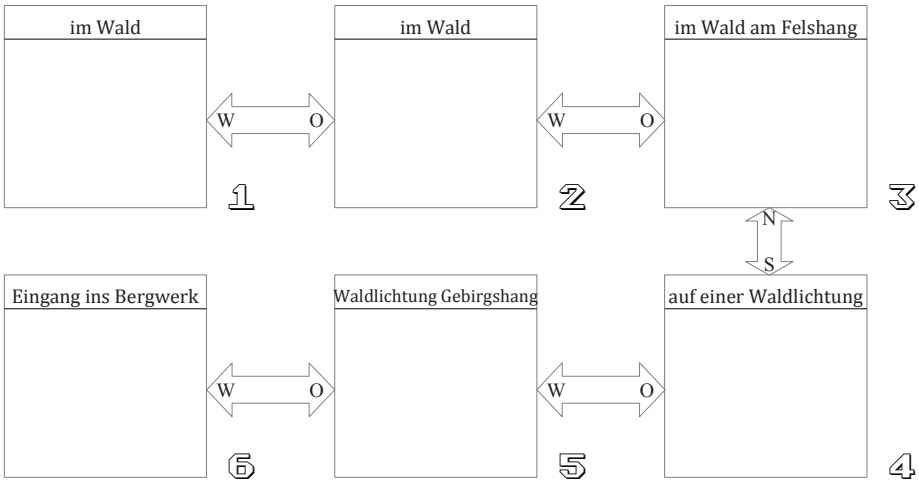
Weiterhin gibt uns unsere Karte Aufschluß über die möglichen Bewegungsrichtungen:

Ich kann nach Westen. Süden.

In unserem Programm könnten wir nun für jeden Raum eine Bedingung der Art *Wenn Spieler in Raum 3, dann drucke ©Ich*

bin im Wald. Vor mir steigt ein steiler Felshang auf© programmieren, würden damit unseren Speicher aber schon nach kurzer Zeit gefüllt haben.

Das Prinzip ist jedoch richtig. Wir werden alle Räume in einer Liste speichern, und zur Ausgabe der Beschreibung auf das jeweils benötigte Element zugreifen.



Geben wir zunächst die Raumbeschreibungen ein, die ausgedruckt werden sollen:

```
200 REM ********** RAUMBESCHREIBUNGEN
201 PRINT "IM WALD."
202 PRINT "IM WALD."
203 PRINT "IM WALD VOR EINEM FELSHANG.
"
204 PRINT "AUF EINER WALDLICHTUNG."
205 PRINT "AUF EINER LICHTUNG AM RANDE
 EINES BERGHANGES."
206 PRINT "AM EINGANGSSTOLLEN EINES AL
TEN BERGWERKES."
```

Wie ein Programmlauf zeigt, existieren die Räume damit bereits im Speicher unseres Computers, wir müssen nun einen Weg finden, sie gezielt anzusprechen.
Hierbei machen wir uns die Unterprogrammtechnik zunutze, die sich bei Atari - Computern mit deren Möglichkeit zu berechneten Sprüngen geradezu anbietet. Wir wollen für jeden einzelnen Raum eine Subroutine vorsehen, die nichts anderes zu tun hat, als dessen Beschreibung auszugeben.

Ein Unterprogramm wird üblicherweise mit GOSUB aufgerufen und mit RETURN beendet, ergänzen wir daher die für die sechs Räume zuständigen Zeilen 201 bis 206:

```
200 REM ********** RAUMBESCHREIBUNGEN
201 PRINT "IM WALD.":RETURN
202 PRINT "IM WALD.":RETURN
203 PRINT "IM WALD VOR EINEM FELSHANG.
":RETURN
204 PRINT "AUF EINER WALDLICHTUNG.":RE
TURN
205 PRINT "AUF EINER LICHTUNG AM RANDE
 EINES BERGHANGES.":RETURN
206 PRINT "AM EINGANGSSTOLLEN EINES AL
TEN BERGWERKES.":RETURN
```

Die Auswahl des Raumes hängt selbstverständlich einzig und allein vom Aufenthaltsort des Spielers ab, das Sprungziel entspricht demnach der Zeile 2ØØ plus der Nummer des aktuellen Raumes.
Nehmen wir an, der Spieler sei in Raum 3 (SPIELER=3), dann druckt der Atari nach Eingabe von GOSUB 2ØØ+SPIELER die korrekte Beschreibung aus.

Nach Eingabe der folgenden Zeilen sind wir bereits in der Lage, die einzelnen Räume zu inspizieren:

```
19Ø GOTO 139Ø

1Ø8Ø PRINT
111Ø PRINT "ICH BIN ";:GOSU8 2ØØ+SPIEL
ER
139Ø PRINT "SPIELER IN WELCHEN RAUM ";
:INPUT SPIELER
5ØØØ GOTO 1Ø8Ø
```

Erkundungen unserer Welt werden dadurch möglich, doch ist die Art der Weiterbewegung keineswegs befriedigend. Wir wollen nicht wahllos von einem Raum in einen anderen springen, sondern streben eine gezielte Orientierung anhand der Himmelsrichtungen an.
Normalerweise wird der gerade vom Spieler besuchte Raum mindestens einen, maximal sechs Ausgänge haben. An Raum 1 grenzt im Osten Raum 2 an, in die Richtungen Norden, Süden und Westen, sowie nach oben und unten, führt kein Weg. Wenn wir darin übereinkommen, daß die Himmelsrichtungen stets in der Reihenfolge N, S, W, O, Oben und Unten genannt werden, kann Raum 1 folgendermaßen spezifiziert werden:

```
          1  im Wald     -,-,-,X,-,-
```

Zweckmäßigerweise werden Ausgänge dann nicht durch ein X markiert, sondern es wird explizit der in dieser Richtung zu erreichende Raum aufgeführt:

```
1  im Wald      Ø,Ø,Ø,2,Ø,Ø
2  im Wald      Ø,Ø,1,3,Ø,Ø
```

Diese sechs Werte werden für jeden Raum in einer Variablen, nennen wir sie DURCHGANG, gespeichert. Da hier zwei verschiedene Werte (Raum und Richtung) den zu betretenden Raum kennzeichnen, ist die Speicherung in einem zweidimensionalen Feld angebracht.

EXKURS: VARIABLEN & FELDER

Um Zwischenergebnisse oder Daten eines Programmes zu speichern, werden in jeder Computersprache Variablen verwendet.
Diese Variablen verhalten sieh wie kleine Fächer, in denen etwas zur Aufbewahrung abgelegt werden kann. Diese Fächer haben Namen, Kofferraum oder Schreibtischschublade wären möglich. Dabei gibt es dann Fächer gleicher Art: am Schreibtisch untereinander Schublade 1, Schublade 2, 3, usw. Jedes Fach heißt Schublade und unterscheidet sich vom nächsten nur durch seine Nummer. So auch in BASIC: Wir können jeder Variablen einen eigenen Namen geben, oder auch gleichartige mit einem einzigen Namen und einer Indexnummer versehen, diese wird dann in Klammern hinter die Bezeichnung der Variablen geschrieben.
Dabei lassen sich zwei grundsätzliche Variablentypen unterscheiden. Die einen nehmen nur Zahlen auf, Zahlen, mit denen auch gerechnet werden kann; die anderen Variablen dienen der Aufnahme von Texten. Zusammensetzung und Länge der Texte spielen keine Rolle, genauer: Wir wollen jedes Zeichen auf unserer Tastatur und Längen bis zu 32767 Zeichen hintereinander erlauben. Wir dürfen nur nicht vergessen, dem Interpreter die gewünschte Länge mitzuteilen, was mit einer DIM - Anweisung in der Form DIM TEXT$(Länge der Zeichenkette) geschehen kann.

Einige Beispiele für korrekte Variablen wären: A, A1, A2, TEXT1 (nur für Zahlen). TEXT1$ (für Texte) als einfache Variablen. A(1), A(2), A(3) als indizierte Variablen.
Hervorzuheben ist an dieser Stelle, daß das Atari BASIC keine indizierten String - Variablen erlaubt !

Vielleicht haben Sie auch schon einige unserer Mitmenschen gesehen, welche versuchen, ihre Wohnung durch das Aufstellen von mit kleinen Miniaturen gefüllter Setzkästen - sie wurden für die Aufbewahrung der kleinen Lettern in einer Druckerei geschaffen - zu verschönern; wir Anfänger stellen uns jetzt einmal jedes einzelne Kästchen als eine Variable vor.
Die Lage eines jeden Kästchens wird durch die Koordinaten Reihe und Spalte festgelegt, d. h., wenn wir wissen wollen, was in dem Fach (2,3) liegt, sehen wir in dem Kästchen nach, welches an dritter Stelle in der zweiten Reihe zu finden ist.
Diese Anordnung wird in Computersprachen als ein zweidimensionales Feld bezeichnet, oder, viel schöner, als Array.

In solch einem Array halten wir nun die Informationen über die Auswege aus einem jeden Raum fest, wobei jeder Raum eine eigene Reihe erhält. Sechs Spalten sind dann erforderlich, um die sogenannte Travel - Table (Richtungstabelle) zu implementieren:

RAUM	***N***	***S***	***W***	***O***	***OB***	***U***
1	Ø	Ø	Ø	2	Ø	Ø
2	Ø	Ø	1	3	Ø	Ø
3	Ø	4	2	Ø	Ø	Ø
4	3	Ø	5	Ø	Ø	Ø
5	Ø	Ø	6	4	Ø	Ø
6	Ø	Ø	Ø	5	Ø	Ø

```
5Ø1 DURCHGANG(1,1)=Ø:DURCHGANG(1,2)=Ø
5Ø2 DURCHGANG(1,3)=Ø:DURCHGANG(1,4)=2
5Ø3 DURCHGANG(1,5)=Ø:DURCHGANG(1,6)=Ø
```

Obige Zeilen programmieren den Durchgang von Raum 1 nach Raum 2 in östlicher Richtung. Entsprechend könnten wir mit den anderen Räumen verfahren, würden dabei jedoch sehr verschwenderisch mit unserem Speicherplatz umgehen (wie oft schreiben wir ©Durchgang© ?), außerdem bereitet die immense Tipparbeit nicht allzu viel Vergnügen.

Aus diesem Grunde verzichten wir auf die Zeilen 5Ø1 bis 5Ø3, und rufen uns an dieser Stelle lieber die Befehle DATA und READ in die Erinnerung zurück.

EXKURS: READ DATA

Üblicherweise benutzen wir die INPUT - Anweisung, um Daten per Tastatur an ein laufendes Programm zu übergeben. Sie hat die Form INPUT Var, wobei Var eine beliebige Variable sein kann. Zur sequentiellen Eingabe mehrerer Daten kann eine ausreichende Anzahl Variablen, durch Kommata getrennt, angehängt werden: INPUT LÄNGE, BREITE, HÖHE kann somit drei Zahlen nacheinander zur weiteren Verarbeitung ins Programm übernehmen.
Sind die Daten bereits bei Programmstart bekannt, können sie direkt ins Programm eingebaut werden, was natürlich nur sinnvoll ist, wenn diese Daten für jeden Programmlauf stets gleich sind. Die dazu notwendigen Schlüsselworte lauten READ und DATA. INPUT wird durch READ ersetzt: READ LÄNGE, BREITE, HÖHE leistet daher die gleiche Funktion, die Eingabedaten werden mit DATA 1,2,3 in einer beliebigen Programmzeile bereitgestellt.
Wichtig zu wissen ist, daß die Daten in der auftretenden Reihenfolge eingelesen werden, die Zahl der Daten in DATA -

Zeilen muß daher gleich der Zahl der Variablen in READ - Anweisungen sein, andernfalls tritt eine Fehlermeldung auf bzw. werden nicht alle Daten berücksichtigt.
Sollen die Daten wieder mit dem ersten Element beginnend eingelesen werden, so ist der Befehl RESTORE zu verwenden.

Wir geben daher folgende Zeilen in unseren Atari ein:

```
15Ø DIM DURCHGANG(AR,6)

5ØØ REM ************* RICHTUNGSTABELLE
5Ø1 DATA Ø,Ø,Ø,2,Ø,Ø
5Ø2 DATA Ø,Ø,1,3,Ø,Ø
5Ø3 DATA Ø,4,2,Ø,Ø,Ø
5Ø4 DATA 3,Ø,5,Ø,Ø,Ø
5Ø5 DATA Ø,Ø,6,4,Ø,Ø
5Ø6 DATA Ø,Ø,Ø,5,Ø,Ø
```

Nun müssen die Daten an die Arbeitsvariablen überwiesen werden. Dazu lesen wir zunächst das Ziel aus den DATA - Zeilen aus und schreiben es dann in die entsprechende Variable des Feldes DURCHGANG. Um Fehler zu vermeiden, muß die Anzahl der Räume bekannt sein; wir führen daher zuvor noch die Variable AR = Anzahl Räume ein. Zweckmäßigerweise sollte sie gleich zu Beginn des Programmtextes initialisiert werden, damit spätere Erweiterungen des Adventures leicht durchgeführt werden können.

```
11Ø AR=6
19Ø GOTO 94Ø
```

Wir gehen nun innerhalb einer Schleife alle Räume durch

```
94Ø FOR RAUM=1 TO AR
```

und initialisieren die Richtungstabelle. Da für jeden Raum sechs Richtungsdaten gelesen werden müssen, installieren wir eine zweite Schleife innerhalb der ersten:

```
95Ø FOR RICHTUNG=1 TO 6
96Ø READ ZIEL:DURCHGANG(RAUM,RICHTUNG)
=ZIEL
97Ø NEXT RICHTUNG
98Ø NEXT RAUM
```

Jede Reihe unseres Arrays ist nun mit einem Raum identisch; in den sechs Spalten einer jeden Reihe ist der Raum gespeichert, den der Spieler durch Eingabe der betreffenden Richtung betreten kann.
Durch diese Anordnung wird uns die im folgenden zu programmierende Fortbewegung des Spielers auf sehr einfache Art und Weise möglich sein.

Bevor dieser sich aber überhaupt irgendwohin wenden kann, muß er sich über die ihm zur Verfügung stehenden Auswege informieren können. Wir müssen also einige Programazeilen entwickeln, die die freien Himmelsrichtungen im Klartext auf dem Bildschirm ausdrucken.
Nehmen wir einmal an, unsere Hauptperson befindet sich augenblicklich in Raum 3. Dann wird die Reihe 3 unserer Richtungstabelle angesprochen:

RAUM	***N***	***S***	***W***	***O***	***OB***	***U***
1	Ø	Ø	Ø	2	Ø	Ø
2	Ø	Ø	1	3	Ø	Ø
3	***Ø***	***4***	***2***	***Ø***	***Ø***	***Ø***
4	3	Ø	5	Ø	Ø	Ø
5	Ø	Ø	6	4	Ø	Ø
6	Ø	Ø	Ø	5	Ø	Ø

Alles was wir tun müssen, ist, die Bezeichnungen derjenigen Spalten auszugeben, die keine Ø enthalten; in unserem speziellen Beispiel die Spalten zwei und drei, Süden und Westen.

Wie wir festgestellt haben, ist die Reihe des Feldes mit dem zu untersuchenden Raum identisch; zur Bereithaltung dieser aktuellen Raumnummer hatten wir zuvor die Variable SPIELER eingeführt. Zur Ausgabe der nicht versperrten Richtungen muß unser Programm somit die sechs Richtungen des Raumes SPIELER testen, und die Namen aller Spalten ausdrucken, welche einen Wert ungleich null haben:

Sechs Richtungen müssen überprüft werden:

```
123Ø FOR RICHTUNG=1 TO 6
124Ø IF DURCHGANG(SPIELER,RICHTUNG)<>O
 THEN PRINT "*** AUSGANG *** "
128Ø NEXT RICHTUNG
```

Bei einem Probelauf wird unser Programm jetzt jeden Raum beschreiben sowie jeden möglichen Ausgang anzeigen; leider aber noch nicht dessen Richtung.
Glücklicherweise steht bei der Überprüfung, ob ein Ausweg existiert, immer die Nummer der gerade untersuchten Richtung in der Zählvariablen RICHTUNG zur Verfügung, weshalb wir genau so verfahren können, wie bei der Ausgabe der Raumbeschreibungen:

```
99Ø GOTO 1Ø8Ø

1Ø1Ø REM ******** KLARTEXT: RICHTUNGEN
1Ø11 PRINT "NORDEN":RETURN
1Ø12 PRINT "SUEDEN":RETURN
1Ø13 PRINT "WESTEN":RETURN
1Ø14 PRINT "OSTEN":RETURN
```

```
1Ø15 PRINT "OBEN":RETURN
1Ø16 PRINT "UNTEN":RETURN

124Ø IF DURCHGANG(SPIELER,RICHTUNG)<>Ø
 THEN GOSUB 1Ø1Ø+RICHTUNG
```

Zur Kontrolle starten wir unser Programm und geben einige Raumnummern ein - ein Blick auf unsere Karte wird uns davon überzeugen, daß wir bisher alles richtig gemacht haben.

Nach diesen Vorbereitungen werden wir nun die gezielte Bewegung in unserem Adventure möglich machen.
Jeder, der bereits einmal ein Abenteuerspiel geladen hatte, wird wissen, daß die Eingabe einer Himmelsrichtung zum Zwecke der Fortbewegung die wohl häufigste Anweisung an die Spielfigur ist. Daher sollten wir als Programmierer dem Spieler soweit entgegenkommen, daß wir die Eingabe des Anfangsbuchstabens als ausreichend betrachten, womit wir dem Abenteurer das Ausschreiben der Himmelsrichtungen, und somit mehrere hundert Tastendrücke, ersparen.

Leider lassen sich auf dem Markt zahlreiche Abenteuerspiele finden, die für die Fortbewegung keine von der Befehlsdecodierung und -ausführung unabhängige Routine aufweisen, und explizit ein GEH WESTLICH erfordern. Wir wollen unsere Richtungsanweisungen jedoch auch deshalb in abgekürzter Form eingeben, weil wir sie dann bevorzugt behandeln können, was auch einer schnelleren Reaktion des Programmes auf die Eingabe hin zugutekommt.

Ersetzen wir zunächst die Zeile 139Ø, um adventuregemäße Eingaben zu ermöglichen:

```
15Ø DIM DURCHGANG(AR,6),EINGABE$(2Ø)
```

```
139Ø PRINT "WAS SOLL ICH TUN";:INPUT E
INGABE$
```

Bevor unser Programm nun irgendwelche Manipulationen des Spieles durchführt, sollte es zunächst die Länge der Spielereingabe überprüfen. Ist die Eingabe länger als zwei Buchstaben (oben=OB), wird es sich um irgendeine Handlung handeln, andernfalls muß die Bewegungsroutine durchlaufen werden. Diese testet zunächst, ob der Weg in der gewünschten Richtung überhaupt frei ist, wenn ja, wird in der Richtungstabelle unter der entsprechenden Richtung (Spalte) des Raumes, in welchem der Spieler sich aufhält (Reihe), der neue Raum abgelesen und der entsprechenden Variablen (SPIELER) zugewiesen. Abschließend wird der Spieler über die Durchführung der Aktion informiert:

```
1Ø8Ø PRINT
14ØØ IF LEN(EINGABE$)>2 THEN 148Ø
141Ø IF EINGABE$="N" AND DURCHGANG(SPI
ELER,1)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,1):PRINT "O.K.":GOTO 1Ø8Ø
142Ø IF EINGABE$="S" AND DURCHGANG(SPI
ELER,2)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,2):PRINT "O.K.":GOTO 1Ø8Ø
143Ø IF EINGABE$="W" AND DURCHGANG(SPI
ELER,3)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,3):PRINT "O.K.":GOTO 1Ø8Ø
144Ø IF EINGABE$="O" AND DURCHGANG(SPI
ELER,4)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,4):PRINT "O.K.":GOTO 1Ø8Ø
145Ø IF EINGABE$="OB" AND DURCHGANG(SP
IELER,5)<>Ø THEN SPIELER=DURCHGANG(SPI
ELER,5):PRINT "O.K.":GOTO 1Ø8Ø
146Ø IF EINGABE$="U" AND DURCHGANG(SPI
ELER,6)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,6):PRINT "O.K.":GOTO 1Ø8Ø
147Ø PRINT "DAHIN FUEHRT KEIN WEG !":G
OTO 1Ø8Ø
```

```
148Ø REM AB HIER SPAETER HANDLUNGEN
2ØØØ GOTO 1Ø8Ø
```

Mit Beginn eines jeden Spieles müssen wir dem Programm natürlich den Startraum mitteilen:

```
12Ø SPIELER=1
```

Ein kurzer Spaziergang durch unsere Adventurewelt wird uns nun schnell davon überzeugen, daß wir als nächstes unbedingt die Gestaltung des Monitorbildes in Angriff nehmen müssen, oder unseren Augen wird sich nach einigen Spielzügen ein wenig informatives Durcheinander bieten.

FORMATIERUNG DER AUSGABE

Um eine saubere Bildschirmgestaltung zu ermöglichen, muß mit Spielstart der Bildschirm natürlich gelöscht werden:

```
1Ø GRAPHICS Ø
1Ø8Ø PRINT
```

Diese Leerzeile ist auf einem leeren Schirm selbstverständlich sinnlos. Erforderlich wird sie immer nach Ausführung eines Befehls, denn unsere Eingabe, wie auch eventuell erfolgte Mitteilungen, müssen um eine Zeile nach oben rutschen. Dieses Scrollen erzielen wir durch ein PRINT in die letzte Bildschirmzeile und ist als Abschluß eines jeden Spielzuges vonnöten.

Für unsere Eingaben benötigen wir die unterste Zeile, d. h. wir müssen nach Ausgabe der Informationen an den Spieler den Cursor dorthin bewegen.

Von den Fähigkeiten her bietet uns das Atari BASIC zwei Möglichkeiten. Zunächst wären da die Cursor - Steuerbefehle, welche die Schreibmarke um eine entsprechende Anzahl Zeilen nach unten bewegen können, doch leider ist dieser Wert von der Menge der ausgegebenen Informationen, sprich Anzahl von Gegenständen, abhängig, und steht während des Spieles nicht so ohne weiteres zur Verfügung.
Da wir einen der Steuercodes im folgenden jedoch verwenden werden, soll auch auf diese Möglichkeit eingegangen werden.

Vereinfacht kann gesagt werden, daß für unseren BASIC - Interpreter jeder Befehl als eine Codezahl, als sogenanntes Token, und nicht als Buchstabenfolge existiert.
Alle Routinen des Interpreters, die für die Bildschirmverwaltung vorgesehen sind, können unter Zuhilfenahme der Funktion CHR$() über diese Zahl angesprochen werden, wobei für die einfachen Cursorsteuerbefehle folgende Zuordnung gilt:

```
CHR$(28)   Cursor hoch
CHR$(29)   Cursor runter
CHR$(3Ø)   Cursor links
CHR$(31)   Cursor rechts
```

So würde beispielsweise der Befehl PRINT CHR$(28) den Cursor um eine Zeile nach oben bewegen.

Für die Lösung wesentlich geeigneter ist der ©POSITION© Befehl, ein Befehl, welcher den Cursor auf eine genau definierte Position setzt. Er wird verwendet in der Form *POSITION X,Y*, wobei Y der Zeile und X der Schreibposition innerhalb einer Zeile entspricht:

```
139Ø POSITION 2,23:PRINT "WAS SOLL ICH
 TUN";:INPUT EINGABE$
```

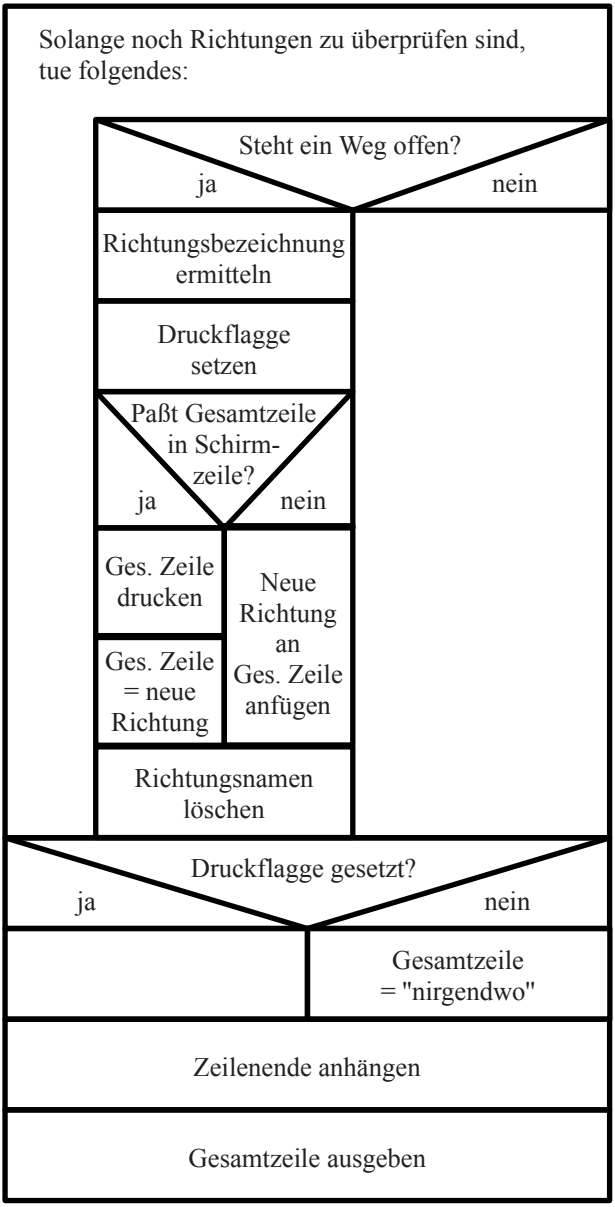
Unsere Eingabe erfolgt somit in Zeile 24 (oberste Zeile gilt als Zeile Ø), und das abschließende RETURN wird den gesamten Bildschirminhalt um eine Zeile nach oben schieben, weshalb für die Mitteilung der Reaktionen an den Spieler keine weiteren Maßnahmen getroffen werden müssen. Mit Beginn des neuen Zuges macht Zeile 1Ø8Ø die unterste Reihe für den nächsten Eingabezyklus wieder frei.

Diese Änderung hat zur Folge, daß momentan alle Ausgaben in der untersten Bildschirmzeile gemacht werden, die Raumbeschreibungen müssen jedoch in der obersten Zeile beginnen:

```
11ØØ POSITION 2,Ø:PRINT "ICH BIN ";:GO
SUB 2ØØ+SPIELER
```

Den Abenteurer über die freien Ausgänge zu informieren, wird für uns allerdings nicht ganz so einfach sein, denn selbst wenn wir mittels eines Semikolons den ©Wagenrücklauf© unterdrücken würden, wäre es unmöglich, mehr als drei Wege einwandfrei auszugeben.
Denn kein Wort sollte über das Ende der Zeile hinausgehen, ebenso sollten die verschiedenen Himmelsrichtungen durch ein Komma getrennt wie auch der Satz mit einem Punkt abgeschlossen werden. Entwickeln wir daher eine Routine, welche diese Aufgabenstellung nach folgendem Schema bewältigt:

Zeilenende vorbereiten: Backspace + "."
Druckflagge löschen
Gesamtzei1e beginnen: "Ich kann nach"


Solange noch Richtungen zu überprüfen sind,
tue folgendes:
Steht ein Weg offen?
ja
nein
Richtungsbezeichnung ermitteln
Druckflagge setzen
Paßt Gesamtzeile in Schirm-zeile?
ja
nein
Ges. Zeile drucken
Ges. Zeile = neue Richtung
Neue Richtung an Ges. Zeile anfügen
Richtungsnamen löschen
Druckflagge gesetzt?
ja
nein
Gesamtzeile = "nirgendwo"
Zeilenende anhängen
Gesamtzeile ausgeben

Bei der Erstellung dieses Struktogramms wurde von folgenden Überlegungen ausgegangen:

ICH KANN NACH NORDEN, SUEDEN, OSTEN,
OBEN, UNTEN.

Fest steht zunächst, daß bevor überhaupt irgend etwas gedruckt werden muß, die Druckzeile mit *Ich kann nach* zu beginnen hat (122Ø).

Nun könnten innerhalb einer Schleife auf gewohnte Weise die Richtungen im Klartext ausgedruckt werden, leider bestünde dabei keine Möglichkeit, die Länge der Ausgabe zu kontrollieren. Als Lösung bot es sich an, zunächst eine komplette Druckzeile zu erstellen, die gerade noch in eine Bildschirmzeile paßte (126Ø, 127Ø).

Daher wird, unter Verwendung einer Hilfsvariablen T$ (T für TEILSTRING), der jeweilige Text ermittelt, und dieser dann, solange die maximale Zeilenlänge nicht überschritten wird, zu der Druckzeile ©addiert©. So lassen sich bis zu drei Richtungsangaben ausdrucken, eine vierte Himmelsrichtung würde allerdings den Grenzwert von 38 Zeichen überschreiten.
Also muß zuvor die bisher ermittelte Druckzeile ausgegeben und die vierte Richtung als neuer Inhalt in DZ$ registriert werden.
Dieser Vorgang wiederholt sich mit der Bearbeitung der zweiten Druckzeile, und zwar solange, bis alle sechs Richtungen überprüft worden sind.
Die nächste Aufgabe besteht nun darin, einen Punkt am Ende des Satzes zu platzieren. Zeile 1Ø4Ø erzeugt für diesen Zweck den String ZENDE$ (Zeilenende), welcher den Cursor um zwei Positionen zurückbewegt und dann einen Punkt druckt. Dieser String wird an den zuletzt zu druckenden Text in DZ$ angefügt; anschließend wird der Inhalt dieser Variablen ausgegeben.

Ein letztes Problem bereitet die immerhin mögliche Situation, daß der Spieler sich in einem Raum befinden kann, welcher scheinbar keine Ausgänge hat. Dann würde die bislang erzeugte Routine ein recht unschönes Bild auf dem Schirm erzeugen, weshalb eine weitere Hilfsvariable als Signal dafür, ob mindestens eine Richtung gedruckt wurde, vorgesehen wird.
Dabei wird die Tatsache genutzt, daß der Rechner einen Variableninhalt von -1 immer als wahr und jeden anderen Wert als falsch betrachtet. So kann im Falle, daß nichts gedruckt wurde (GEDRUCKT<>-1), der Text ©nirgendwo© an die Druckzeile angefügt werden.

Somit werden folgende Änderungen und Ergänzungen unseres Programmes notwendig:

```
15Ø DIM DURCHGANG(AR,6),EINGABE$(2Ø),
DZ$(42),T$(4Ø),ZENDE$(3)
1Ø11 T$="NORDEN, ":RETURN
1Ø12 T$="SUEDEN, ":RETURN
1Ø13 T$="WESTEN, ":RETURN
1Ø14 T$="OSTEN, ":RETURN
1Ø15 T$="OBEN, ":RETURN
1Ø16 T$="UNTEN, ":RETURN

1Ø4Ø ZENDE$=CHR$(3Ø):ZENDE$(LEN(ZENDE$
)+1)=CHR$(3Ø):ZENDE$(LEN(ZENDE$)+1)=".
"
122Ø DZ$="ICH KANN NACH ":GEDRUCKT=Ø
124Ø IF DURCHGANG(SPIELER,RICHTUNG)=Ø
 THEN GOTO 128Ø
125Ø GOSUB 1Ø1Ø+RICHTUNG:GEDRUCKT=-1
126Ø IF LEN(DZ$)+LEN(T$)>=38 THEN PRIN
T DZ$:DZ$=T$(1,LEN(T$)):T$="":GOTO 128
Ø
127Ø DZ$(LEN(DZ$)+1)=T$
128Ø T$="":NEXT RICHTUNG
```

```
129Ø IF  NOT GEDRUCKT THEN DZ$(LEN(DZ$)+
1)="NIRGENDWO. "
13ØØ DZ$(LEN(DZ$)+1)=ZENDE$
131Ø PRINT DZ$
```

GOLD, SILBER UND ANDERE NÜTZLICHE DINGE

Eine Wanderung durch unsere Adventurewelt verschafft dem Abenteurer bislang einen recht trostlosen und uninteressanten Eindruck, stehen ihm doch keinerlei Handlungsgegenstände zur Verfügung. Darüber hinaus werden reine Ortsbeschreibungen beim besten Willen kein Spiel ermöglichen. Nehmen wir also wieder einmal unsere Karte zur Hand und bereichern wir unsere Welt um die gewünschten Gegenstände, wobei wir uns in der Regel auf für den Fortlauf des Spieles wichtige Objekte beschränken wollen. Denn für reine Verschönerungen wäre der spätere Aufwand zu groß, schließlich müssen wir gewährleisten, daß der Spieler alles das, was er findet, auch in die Hand nehmen und untersuchen kann. Dennoch kommen wir nicht umhin, in jedem Raum mindestens ein Objekt zu platzieren, denn daß der Abenteurer so ohne weiteres gar nichts sieht, ist doch sehr zweifelhaft. Falls die Phantasie nicht ausreicht, ist das betreffende Objekt eben *nichts besonderes*, eine Formulierung, die wir auch deshalb so häufig in Adventurespielen finden, weil sie gerne zur Verminderung des Arbeitsaufwandes gewählt wird, denn in einem normalen Wald beispielsweise sind Bäume, Sträucher, Gras, Insekten usw. eben wirklich nichts besonderes.
Oftmals kann jedoch auch auf unnötige Dinge nicht verzichtet werden, besonders dann nicht, wenn sie zum Milieu gehören und das Adventure einen realistischen Eindruck hinterlassen soll.

So werden auch wir in unserem Wald Bäume aufstellen müssen, obwohl die Handlung sie nicht erfordert. In Raum zwei, der

bereits nahe dem Gebirge liegt, vertiefen einige Felsbrocken den Eindruck der Wirklichkeit. Weiterhin gehört zu einem Bergwerk eine Baracke, egal ob es nun die Wohnstätte des Minenbesitzers oder ein Geräteschuppen ist. Irgendwo müssen wir schließlich auch die zu suchenden Schätze verstecken, zusätzlich sind einige Gegenstände, die dem Spieler eher schaden als nützen, angebracht.

Lassen Sie mich daher bitte im folgenden den mir vorschwebenden Handlungsablauf näher konkretisieren und die benötigten Gegenstände auswählen:

HANDLUNGSABLAUF GOLDRAUSCH

Version 1 soll dem Spieler die Aufgabe stellen, zwei Schätze, Goldstücke und Silbermünzen, im Umfeld der Mine zu finden. Bei Spielbeginn soll sich der Spieler im Wald befinden und sich zunächst an das Bergwerk herantasten müssen. Auf dem Weg zur Mine entdeckt er im Wald ein Erdloch, auf dessen Grund sich eine schwere Eisentruhe befindet. In dieser befindet sich das Silber, doch leider ist die Truhe mittels Vorhängeschloß und einer schweren Eisenkette verschlossen; ein passender Schlüssel läßt sich nirgends finden. Doch am Eingangsstollen des Bergwerkes liegen einige Eisenstangen, die stabil genug sind, um damit die Kette zu zerreißen oder das Schloß zu sprengen. Wir können es den Spieler auch mit Sprengstoff versuchen lassen, soll er in einer Holzhütte ruhig Dynamit finden - für uns eine gute Gelegenheit, sein vorzeitiges Ende zu programmieren. Die Goldstücke deponieren wir in einer Felsenhöhle, die zum Leidwesen des Spielers von einem wilden Bären bewohnt wird. Das wird der Spieler jedoch erst

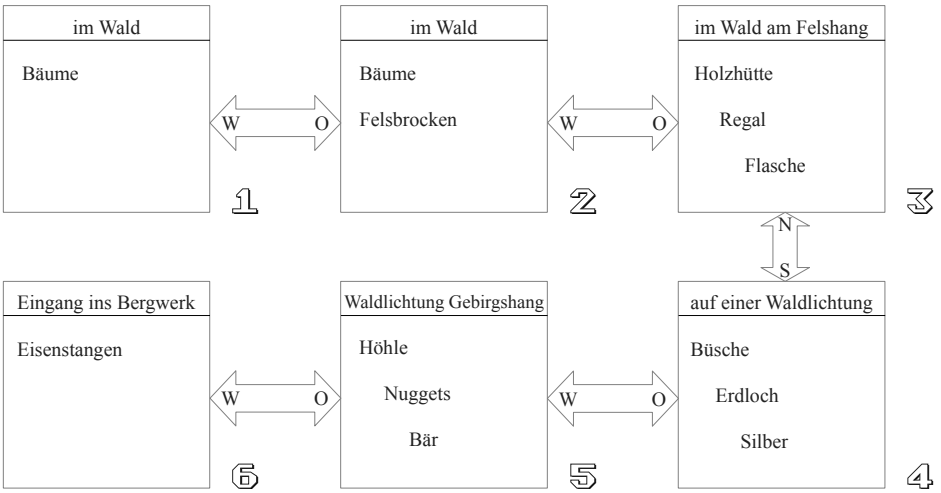
erfahren, wenn er die Höhle untersucht, sollte die Gier nach dem gelben Metall siegen und ihn dazu verleiten, sofort zuzugreifen, wird ebenso der Hunger des Bären selbigen seine Scheu vor dem Menschen verlieren lassen - nach dem Dynamit die zweite Falle für den Abenteurer.
Andererseits wissen wir alle dank Wilhelm Busch, daß man Bären mit Honig eine große Freude machen kann, stellen wir auf einem Regal in der Hütte daher eine Flasche bereit, die den begehrten Stoff enthält. Betritt der Spieler damit gewappnet die Höhle, wird der Bär den Honig wittern und mit der Flasche in den Tiefen der Höhle verschwinden, womit einem erfolgreichen Ende dieses Miniadventures nichts mehr im Wege steht.

Die Karte zu unserem Adventure *Goldrausch* dürfte nun ähnlich der folgenden Abbildung aussehen.

DIE OBJEKTE

Bevor wir mit der Programmierung der Handlungsgegenstände unser Adventure vervollständigen, werden einige weitere Vorüberlegungen notwendig.

Um dem Spieler das Hineinleben in unsere Computerwelt zu ermöglichen, werden wir diese Welt nicht mit trockenen Substantiven beschreiben können, sondern wir werden uns bemühen, das Interesse des Spielers durch plastische Schilderungen wachzuhalten. Die lapidare Mitteilung ©Ich sehe einen Bären.© läßt den Spieler unser Adventure zwar spielen, die Nachricht ©Ich sehe einen äußerst grimmig dreinblickenden Bären© wird ihn unser Adventure dagegen erleben lassen.


im Wald
Bäume
W
O
1
im Wald
Bäume
Felsbrocken
W
O
2
im Wald am Felshang
Holzhütte
Regal
Flasche
3
N
S
Eingang ins Bergwerk
Eisenstangen
W
O
6
Waldlichtung Gebirgshang
Höhle
Nuggets
Bär
W
O
5
auf einer Waldlichtung
Büsche
Erdloch
Silber
4

Sollte in dem Abenteurer dann das Bestreben erwachen, einen neuen Freund zu finden, dann wäre eine Eingabe wie ©Streichel äußerst grimmig dreinblickenden Bären© für uns als Programmierer zwar einfach zu realisieren, für den Spieler hingegen unzumutbar.
Daher werden wir jedem Objekt neben der eigentlichen Objektbeschreibung einen Rufnamen zuordnen, womit wir auch die Möglichkeit erhalten, die Wortlänge unseres Adventures nach Bedarf frei festzulegen. Vermutlich haben Sie bemerkt, daß professionelle Adventures aus dem englischsprachigen Raum meist nur die ersten drei oder vier Buchstaben zur Identifikation eines jeden Wortes erfordern. Drei Buchstaben sind auch für deutsche Adventures in der Regel ausreichend, nur bei der Realisation eines größeren Projektes sollten Sie zuvor eine Liste der vorgesehenen, Gegenstände erstellen und diese aufmerksam durchsehen, ansonsten werden Sie bei der Programmierung der Bedingungen und Aktionen überrascht feststellen, wie viele für ein Adventure wichtige Substantive mit Sch beginnen. Wird Ihr Spiel weiterhin mit Groß- und Kleinschrift ausgestaltet, achten Sie bitte darauf, bei den Rufnamen nur Großbuchstaben zu verwenden, denn diese Kürzel werden später mit den Eingaben des Spielers auf Gleichheit überprüft.

Als dritte Information neben dem ausführlichen und dem Kurznamen werden wir zu jedem Objekt die Nummer des Raumes speichern müssen, in dem der Gegenstand sich gerade befindet. Da sich diese Zahlen während des Spieles laufend ändern, können sie nicht aus einer fest programmierten Programmzeile abgelesen werden, sondern müssen in entsprechenden Variablen bereitstehen.
Erzeugen wir daher eine Liste, in der jeder Eintrag dem Aufenthaltsort eines Objektes entspricht.
Zweckmäßigerweise verwenden wir die indizierte Variable OB() und geben uns damit ein einfaches Mittel zur Hand, um die Position eines jeden Gegenstandes kontrollieren und manipulieren zu können.

Denn solange ein Gegenstand nicht aktiv am Spiel beteiligt ist, soll OB = Ø sein, wir sagen, der Gegenstand befindet sich im *Lagerraum*. Andernfalls entspricht der Inhalt von OB entweder dem Wert der Variablen Spieler und ist somit im Raum sichtbar, oder aber er ist gleich -1 in dem Falle, daß der Abenteurer ihn mit sich führt (*Inventory*).

Die folgenden Zeilen stellen die Beschreibungen aller Gegenstände, die für eine Realisation des bisherigen Spielplanes (Goldrausch, Miniversion) erforderlich sind, für die Ausgabe auf dem Bildschirm bereit.

Jede Zeile entspricht dabei einem kleinen Unterprogramm, da wir die bereits anhand der Himmelsrichtungen erarbeitete Technik verwenden wollen:

```
3ØØ REM ***************** GEGENSTAENDE
3Ø1 T$="VIELE GROSSE BAEUME":RETURN
3Ø2 T$="VIELE GROSSE BAEUME":RETURN
3Ø3 T$="EINIGE FELSBROCKEN":RETURN
3Ø4 T$="EINE VERFALLENE HOLZHUETTE":RE
TURN
3Ø5 T$="EINE VERSCHMUTZTE KORBFLASCHE"
:RETURN
3Ø6 T$="HONIG":RETURN
3Ø7 T$="EINE HOLZKISTE":RETURN
3Ø8 T$="EIN KLAPPRIGES REGAL":RETURN
3Ø9 T$="ETWAS SPRENGSTOFF":RETURN
31Ø T$="EIN DUESTERES ERDLOCH":RETURN
311 T$="EINE ROSTIGE EISENTRUHE":RETUR
N
312 T$="*SILBERMUENZEN*":RETURN
313 T$="EINE DUESTERE FELSENHOEHLE":RE
TURN
314 T$="EINEN GRIMMIG DREINBLICKENDEN
BAEREN":RETURN
```

```
315 T$="NIEDRIGE BUESCHE":RETURN
316 T$="MEHRERE EISENSTANGEN":RETURN
317 T$="* NUGGETS *":RETURN
```

Jedem dieser siebzehn Gegenstände ordnen wir nun eine Position innerhalb unserer Adventurewelt zu:

```
4ØØ REM ********** POSITIONEN OBJEKTE
41Ø DATA 1,2,2,3,Ø,Ø,3,Ø,Ø,Ø,Ø,Ø,5,Ø,4
,6,5

9ØØ FOR OBJEKT=1 TO AO
91Ø READ LAGEORT:OB(OBJEKT)=LAGEORT
92Ø NEXT OBJEKT
```

Machen wir die Einrichtung eines Raumes nun dem Spieler sichtbar:
In der Variablen SPIELER finden wir die Nummer des betretenen Raumes, OB() enthält die Raumnummern aller Objekte. Überprüfen wir daher innerhalb einer Schleife, ob die Positionsnummer eines Gegenstandes gleich der aktuellen Raumnummer ist:

```
111Ø DZ$="ICH SEHE ":GEDRUCKT=Ø
112Ø FOR I=1 TO AO
113Ø IF OB(I)<>SPIELER THEN GOTO 117Ø
114Ø GEDRUCKT=-1:GOSUB 3ØØ+I:T$(LEN(T$
)+1)=", "
115Ø IF LEN(DZ$)+LEN(T$)>=38 THEN PRIN
T DZ$:DZ$=T$(1,LEN(T$)):T$="":GOTO 117
Ø
116Ø DZ$(LEN(DZ$)+1)=T$:T$=""
117Ø NEXT I
118Ø IF  NOT GEDRUCKT THEN DZ$(LEN(DZ$)
+1)="NICHTS BESONDERES. "
119Ø DZ$(LEN(DZ$)+1)=ZENDE$
```

```
12ØØ PRINT DZ$
```

Programmzeile 111Ø setzt die Signalflagge, die wir später benötigen, um feststellen zu können, ob etwas ausgedruckt wurde, zurück und legt den Beginn der Druckzeile fest.
113Ø testet für jeden Gegenstand, ob dieser sich in einem anderen Raum als der Spieler befindet. Fällt dieser Test positiv aus, wird sofort das nächste Objekt überprüft, andernfalls wird die Druckflagge gesetzt, die Objektbeschreibung geholt und diese durch ein Komma ergänzt. Dieser Vorgang wiederholt sich, bis eine Gesamtlänge der Druckzeile von 38 Zeichen erreicht worden ist und der Text ausgegeben wird, oder bis die Schleife vollständig abgearbeitet wurde und somit alle sichtbaren Gegenstände ausgegeben worden sind.
119Ø schließlich ersetzt das letzte Zeichen und schließt die Druckroutine mit einem Punkt ab.

Um den Bildschirmaufbau der amerikanischen Originaladventures zu erreichen, fehlen uns nun nur noch eine Trennungslinie sowie eine weitere Leerzeile:

```
15Ø DIM .....,LEERZEILE$(38)

1Ø2Ø LEERZEILE$="
                "
121Ø PRINT LEERZEILE$
132Ø PRINT"---------------------------
---------":REM 37 X CONTROL R
```

Ein letztes Problem werden wir wiederum erst nach mehreren Eingaben feststellen. Die Mitteilungen innerhalb der unteren Schirmhälfte rücken immer weiter nach oben und vermischen sich nach einigen Zügen mit den Beschreibungen der Umgebung. Es ist daher Aufgabe des Programmes, vor deren Ausgabe die obersten Zeilen zu löschen, was durch

Drucken einer ausreichenden Menge von Leerzeilen geschehen kann:

```
1Ø9Ø FOR ZEILE=Ø TO 1Ø:POSITION 2,ZEIL
E:PRINT LEERZEILE$:NEXT ZEILE
```

DER WORTSCHATZ

Bevor wir nun das eigentliche Spiel programmieren und uns mit den für einen Spieler nicht sichtbaren Aktionen des Programmes beschäftigen, müssen wir schnell noch die dazu notwendigen Verben eingeben. Diese werden, zusammen mit den zuvor erwähnten Rufnamen der Objekte, den kompletten Wortschatz unseres Adventures darstellen und somit Grundlage zur Verständigung mit dem Spieler sein.
Diese Worte sind es, die wir mit den Eingaben des Abenteurers auf Identität prüfen, das heißt, es werden ständig Vergleiche durchgeführt werden. Nun wäre es aber bei weitem zu aufwendig, ähnlich wie bei der Behandlung der Objekte, für jedes einzelne Wort eine eigene Zeile zu programmieren. Was uns hier jetzt zugute kommt, ist unsere Vereinbarung, eine genau definierte Wortlänge zu wählen, womit uns die Anzahl der signifikanten Stellen einer Eingabe genau benannt ist. Deshalb bietet es sich an, jeweils gleich lange Teile eines großen Strings zu analysieren. Von diesen Strings werden wir zwei benötigen, den einen für die Verben und den anderen für die Kurzbeschreibungen der Gegenstände:

```
16Ø DIM VERBEN$(4Ø),OBJEKTE$(8Ø)
17Ø VERBEN$="UNTENIMMLEG OEFFBENUZERS"
18Ø OBJEKTE$="BAEUBAEUFELSHUETFLASHONI
KISTREGASPREERDLTRUHMUENHOEHBAERBUESEI
SNUGG"
```

ANALYSE DER EINGABEN

Vom Spieler gewünschte Richtungsänderungen, eine Befehlsgruppe mit Wortlängen von ein bis zwei Buchstaben, werden bereits erkannt und korrekt ausgeführt. Die Standardeingabe besteht jedoch immer aus Verb und Objekt, wobei die Wortlängen, abgesehen von der zur Erkennung der Begriffe notwendigen Mindestwortlänge, keinen Restriktionen unterworfen werden. Um wie vom Spieler gewünscht reagieren zu können, muß unser Programm dessen Eingabe überprüfen:

Ist das benutzte Verb programmiert ?
Ist das Objekt vorgesehen ?

Können beide Fragen mit Ja beantwortet werden, wird die Eingabe in Verb und Objekt zerlegt, Verb- und Objektnummer werden ermittelt; sind die Bestandteile der Eingabe nicht definiert, wird eine entsprechende Mitteilung an den Abenteurer ausgegeben.

EXKURS: STRINGBEHANDLUNG

Der wesentliche Unterschied zwischen einem Computer und einem programmierbaren Taschenrechner ist die Fähigkeit des Computers, mit Texten umzugehen. Natürlich hat ein Rechner bislang kein Verständnis für diese Texte, sondern er faßt sie als eine Ansammlung bestimmter Zeichen, als sogenannte Zeichenkette oder String auf. Bei diesen Strings handelt es sich im Allgemeinen um Kombinationen aller per Tastatur eingebbarer Zeichen. Das Betriebssystem des Computers stellt nun eine Reihe von Funktionen zur Behandlung dieser Zeichenketten zur Verfügung.
Unser Atari bietet neben einer Reihe von Konvertierungsbefehlen wie VAL() und STR$(), die eine Zahl

in einen String bzw. einen String in eine Zahl verwandeln, die Möglichkeit, Strings zusammenzusetzen und zu zerlegen. So können wir mit B$=A$(1,3) die ersten drei Buchstaben der Zeichenkette A$ in BS zur weiteren Bearbeitung durch das Programm bereithalten, im Falle von A$ gleich "ABCDEF" wäre der Inhalt von BS beispielsweise "ABC".
Um nun gezielt einen solchen Substring erzeugen zu können, müssen bei Aufruf dieser Funktion der Start- und der Endwert als Parameter bekannt sein. Dies kann einerseits durch eine Vereinbarung geschehen (so beginnt in unserem String OBJEKTE$ nach jedem dritten Buchstaben ein neues Wort), wie auch durch die gezielte Suche nach einem Kennzeichen innerhalb eines Strings.
So wird unser Adventureprogramm beispielsweise die Position des Leerzeichens zwischen Verb und Objekt der Eingabe eines Spielers ermitteln und unter Verwendung dieser Kennzahl die zwei Substrings EVERB$ und EOBJEKT$ ermitteln.
Um die Anzahl erforderlicher Buchstaben jedoch berechnen zu können, muß zuvor mit LEN(EINGABE$) die Gesamtlänge der Eingabe ermittelt werden.

Wird die Ausführung einer Eingabe nicht in den Zeilen 141Ø bis 147Ø durchgeführt, so wird, falls die Länge der Eingabe weniger als drei Buchstaben beträgt, angenommen, daß es sich um einen Eingabefehler handelt. Nachdem *Dahin führt kein Weg !* ausgegeben worden ist, beginnt ein neuer Spielerzug.
Handelt es sich um eine längere Eingabe, wird der Programmlauf mit Zeile 2ØØØ fortgesetzt:

```
16Ø DIM ...,EVERB$(2Ø),EOBJEKT$(2Ø)
147Ø IF LEN(EINGABE$)<3 THEN PRINT "DA
HIN FUEHRT KEIN WEG !":GOTO 1Ø8Ø

2ØØØ LN=LEN(EINGABE$)
2Ø1Ø FOR 1=1 TO LN
2Ø2Ø IF EINGABE$(I,I)<>" " THEN NEXT I
```

```
2Ø3Ø EVERB$=EINGABE$(1,I)
2Ø4Ø IF LEN(EVERB$)=LN THEN GOTO 2Ø9Ø
2Ø5Ø EOBJEKT$=EINGABE$(I+1,LN)
```

Nach Ermittlung der Länge der gesamten Eingabe wird innerhalb einer Schleife jeder Buchstabe, beginnend von links, darauf überprüft, ob er mit einem Leerzeichen identisch ist. Ist das Leerzeichen gefunden, steht die Länge des Verbes fest (vom ersten bis zum gerade geprüften Zeichen) und das Verb kann der Variablen EVERB$ zugewiesen werden. Die Zeichenkette ab der nächsten Position (da das Leerzeichen übersprungen werden muß) entspricht somit dem Objekt der Eingabe.
Wurde ein ©Ein Wort Befehl© verwendet (HELP), so wird die Verblänge gleich der Eingabelänge sein. Da kein Objekt zur weiteren Bearbeitung bereitgestellt werden muß, wird direkt (Zeile 2Ø4Ø) zur Verbanalyse (Zeile 2Ø9Ø) verzweigt.

```
2Ø9Ø FOR I=1 TO LEN(VERBEN$) STEP 4
21ØØ VN=VN+1
211Ø IF VERBEN$(I,I+2)=EVERB$ THEN 214
Ø
212Ø NEXT I
```

Innerhalb einer weiteren Schleife werden die jeweils nächsten vier Buchstaben unseres Strings VERBEN$, der die Bezeichnungen aller im Spiel möglichen Verben enthält, mit dem zuvor ermittelten Eingabeverb verglichen. Ein Zähler hält die Nummer des gerade überprüften Verbes bereit, so daß die Schleife verlassen werden kann, wenn eine Identität festgestellt worden ist. Wurde die Schleife vollständig abgearbeitet und kein passender Substring festgestellt, so hatte der Programmierer das betreffende Verb nicht vorgesehen, und dem Spieler wird darüber Mitteilung gemacht:

```
213Ø PRINT "ICH VERSTEHE DAS VERB NICH
T !":GOTO 1Ø8Ø
```

Anschließend wird nach gleichem Prinzip die Objektnummer ermittelt:

```
214Ø FOR I=1 TO LEN(OBJEKTE$) STEP 4
215Ø N=N+1
216Ø IF OBJEKTE$(I,I+2)=EOBJEKT$ THEN
 GOTO 22ØØ
217Ø NEXT I
218Ø PRINT "ICH VERSTEHE DAS OBJEKT NI
CHT !":GOTO 1Ø8Ø
```

Mit Eingabe dieser Zeilen haben wir einen wesentlichen Teil unserer Entwicklungsarbeit hinter uns gebracht. Die entwickelten Routinen sorgen für einen ansprechenden Aufbau des Bildschirmes, wie sie auch das ©Verständnis© für die Eingabe des Spielers erzeugen.
Sie ermitteln, welches der möglichen Verben der Abenteurer benutzt hat und mit welchem Gegenstand er etwas tun möchte, womit eine Verzweigung des Programmablaufes an eine zur Ausführung der betreffenden Aktion vorgesehene Programmzeile möglich wird. Wegen dieser zentralen Steuerungsaufgaben wird dieser Programmteil auch als TREIBER bezeichnet.

ÜBERBLICK: AUFBAU DER PROGRAMME

Somit wird deutlich, daß unsere Abenteuerprogramme im wesentlichen aus drei Teilen bestehen:

1. *Daten des Adventures*
2. *Adventuretreiber*
3. *Ausführung der Spielzüge*

1. Die Daten stellen die Grundlage eines jeden Spieles dar. Sie werden zum Teil an Arbeitsvariablen übergeben oder innerhalb von kurzen Unterprogrammen für die Übergabe an das Hauptprogramm bereitgestellt.

2. Dem Treiber fällt die Steuerung des gesamten Programmablaufs zu. Er gestaltet die Bildschirmausgabe, übernimmt die Eingaben des Spielers, wertet sie aus und leitet die Befehlsausführung ein.
Der Treiber ist vom jeweiligen Adventure unabhängig und kann unverändert für alle Adventures übernommen werden.

3. Der dritte Programmteil stellt das eigentliche Adventure dar. Treffen alle für eine Handlung notwendigen Bedingungen zu, so wird diese Aktion ausgeführt.

Damit Ihnen der Überblick erhalten bleibt, fassen wir an dieser Stelle den Aufbau unserer Adventures zusammen, wobei bereits auch die noch im folgenden zu erstellenden Routinen mit angegeben werden.
Dabei deutet es sich bereits an, wie universell dieses Konzept eingesetzt werden kann.
So werden einige kleine Änderungen am Treiber ausreichen, um aus einem Textadventure ein Grafikadventure machen zu können, ebenso wird es Dank des einheitlichen, systematischen Aufbaus mit genau definierten Programmzeilen für die diversen Funktionsblöcke weiterhin möglich sein, einen Adventuregenerator zu entwickeln.

KONZEPTIONELLER AUFBAU: TEXTADVENTURE

DATEN

Ø	*1ØØ*	*TITELBILD*
1ØØ	*15Ø*	*KONTROLLDATEN*
15Ø	*16Ø*	*SPEICHERPLATZ RESERVIEREN*
17Ø		*GESAMTSTRING VERBEN*
18Ø		*KURZNAMEN DER OBJEKTE*
2ØØ	*3ØØ*	*RAUMBESCHREIBUNGEN*
3ØØ	*4ØØ*	*OBJEKTBESCHREIBUNGEN*
4ØØ	*5ØØ*	*LAGE DER OBJEKTE*
5ØØ	*6ØØ*	*VERBINDUNGEN DER RÄUME*
6ØØ	*7ØØ*	*STANDARDMITTEILUNGEN*
7ØØ	*8ØØ*	*EV. 2. TITEL*
8ØØ	*9ØØ*	*ALLGEMEINE SPIELANWEISUNG*
9ØØ	*1ØØØ*	*SPIELDATEN INITIALISIEREN*

ADVENTURETREIBER

1ØØØ	*1Ø2Ø*	*RICHTUNGSTABELLE*
1Ø3Ø	*1Ø4Ø*	*STEUERVARIABLEN*
1Ø8Ø	*132Ø*	*BILDSCHIRMVERWALTUNG*
1321	*1389*	*BES. EREIGNISSE*
139Ø		*EINGABE DES SPIELZUGES*
1391	*1399*	*FALLEN & HINDERNISSE*
14ØØ	*15ØØ*	*HAUPTPERSON BEWEGEN*
15ØØ	*16ØØ*	*INVENTUR*
16ØØ	*17ØØ*	*SAVE GAME*
17ØØ	*18ØØ*	*LOAD GAME*
18ØØ	*195Ø*	*HELP, VOKABULAR, INSTRUKT.*
195Ø	*2ØØØ*	*SPIELABBRUCH*
2ØØØ	*218Ø*	*ANALYSE DER EINGABE*
22ØØ		*SPRUNGTABELLE AUSFÜHRUNG*

AUSFÜHRUNG

5ØØØ	*3ØØØØ*	*SPIELZÜGE*

WANN GEHT WAS ?

Stellen wir uns nun den Spieler vor, der unsere zwei Schätze finden will. Mit einem Blick stellt er fest, daß er sich mitten in einem Wald befindet, umgeben von Bäumen und Felsbrocken. Als geübter Adventurespieler oder Leser der vorangegangenen Kapitel weiß er, daß des Pudels Kern in den unscheinbarsten, alltäglichsten Dingen liegen kann.

Wie wird er reagieren ?
Welche Eingaben müssen wir erwarten ?

Mit an Sicherheit grenzender Wahrscheinlichkeit probiert er mit Anweisungen wie *UNTERSUCHE WALD*, *UNTERSUCHE BAUM*, oder *UNTERSUCHE FELSBROCKEN* weiterzukommen.
NIMM BAUM ist weniger wahrscheinlich, dennoch müssen wir, um allen Kritikern unserer Programme den Wind aus den Segeln zu nehmen, eine Mitteilung der Art *So stark bin ich nicht.* programmieren, außerdem animieren vielfältige Mitteilungen, die zeigen, daß eine Eingabe auch wirklich verstanden wurde, den Abenteurer zum Weiterspielen.
Verständlich - wäre das Ergebnis jeder zweiten oder dritten Eingabe ein *Ich verstehe nicht, was Du meinst !* dann wird der Reiz zum Abschalten des Computers doch sehr groß.

In der Praxis werden wir für die Räume 1 und 2 daher, weil wir hier noch keine Handlung vorgesehen haben, sondern nur unsere Welt größer wirken lassen wollen, nur die Ausgabe diverser Meldungen an den Spieler vorsehen, eine Aufgabe, die ein Print - Befehl schnell erledigt. Doch nachdem der Spieler Raum 3 betreten und die Holzhütte entdeckt hat, wird er diese ebenfalls in Augenschein nehmen und dabei das Regal mit der Korbflasche entdecken. Diese muß natürlich ab diesem Zeitpunkt, zusätzlich zum Regal und zur Hütte, in der Beschreibung der Szene erscheinen (©Ich sehe ...©), das heißt, wir müssen ihren Aufenthaltsort verändern. Weiteres

Kopfzerbrechen konnte uns die Eingabe *NIMM FLASCHE* bereiten: Nun muß die soeben erschienene Korbflasche wieder verschwinden. Zurück an ihren alten Platz kann sie nicht, denn wenn unser Spieler *INVENTUR* macht, muß sie selbstverständlich aufgelistet werden.

Ebenso selbstverständlich kann der Spieler die Flasche nicht schon in Raum 1 untersuchen, genau so wenig in Raum 2, überhaupt sollten alle Eingaben betreffend der Flasche abschlägig beantwortet werden, solange diese sich nicht in unmittelbarer Nähe des Spielers befindet.

Dies ist jedoch eine Geschmackssache, die im Ermessen des Programmierers liegt. Denn selbstverständlich darf er die Meinung vertreten, daß die Flasche in Wirklichkeit bereits seit undenklichen 2eiten unberührt in dem Regal steht, und daß der Spieler sie nur noch nicht gesehen hat.
Wird dieser Spieler durch die zahlreichen Überraschungen eines Abenteuers jedoch gezwungen, noch einmal von vorne zu beginnen, kennt er die Startpositionen diverser Gegenstände, und man kann es ihm ohne weiteres ermöglichen, diese zu nehmen, auch ohne daß sie in der Zeile ©Ich sehe ...© gelistet worden waren.

In der Tat wird der Vorgang des Nehmens dadurch sogar einfacher zu programmieren, schließlich muß eine Bedingung weniger überprüft werden, doch wollen wir es wie die Mehrheit der Adventureproduzenten halten und nur sichtbare Gegenstände auch greifbar sein lassen.

So wird denn auch die Holzhütte von unserer Hauptperson nur untersucht werden können, wenn sie sich in greifbarer Nähe befindet. Den Honig werden wir erst entdecken, nachdem wir die Korbflasche in unsere Hände genommen und geöffnet haben. Der Bär wird sich nur friedlich verhalten, wenn er durch den Verzehr des Honigs anderweitig beschäftigt ist.

Alle diese Bedingungen müssen vor Ausführung eines Befehls zunächst auf ihre Richtigkeit überprüft werden. Bereits unser kurzes Adventure mit nur sechs Räumen wird somit eine Vielzahl an Programmzeilen erfordern. Berücksichtigen wir nur die zu diesem Zeitpunkt der Entwicklung eingebrachten Verben untersuche, nimm, lege weg, öffne, benutze, zerstöre und unsere siebzehn Gegenstände, so müssen wir bereits Aktionen für 1Ø2 unterschiedliche Spielereingaben programmieren. Würden wir die Programmzeilen alle ohne besonderes Konzept hintereinander schreiben, wäre es leicht, für die auftretenden, recht langen Ausführungszeiten des Programmes eine Erklärung zu finden. Um sie jedoch möglichst gering zu halten, werden wir den nun zu erstellenden Programmteil in mehrere Blöcke aufteilen.

Prinzipiell stehen uns zwei voneinander verschiedene Möglichkeiten zur Verfügung. Zum einen könnten wir für jeden Raum eine ausreichende Anzahl von Programmzeilen reservieren, und dort für jede mögliche Aktion des Spielers ein paar Statements reservieren. Diese Technik wird recht häufig gewählt, hat auch den Vorteil, daß sie wegen der direkten Ausführung eines Befehls, der ja zuvor nicht irgendwie aufbereitet werden muß, recht schnell ist, und daß vor allem ein recht übersichtliches Programm entsteht, welches bequem und einfach, ohne Nachsehen in irgendwelchen Listen, erweitert oder editiert werden kann. Sollten Sie sich einmal mit dieser Ausführung eines Adventureprogrammes beschäftigen wollen, dann soll die folgende Realisation unseres Raumes 1 als Anregung genügen. Beachten Sie, daß es sich um ein Beispiel handelt, welches mit unserem Konzept nichts zu tun hat, überschreiben Sie daher bitte keine Zeilen unseres bisher erstellten Programmes.

```
1ØØ PRINT"WAS SOLL ICH TUN":INPUT EING
ABE$
2ØØ REM EINGABE$ IN VERB$ UND OBJEKT$
ZERLEGEN (WIE BESPROCHEN)
3ØØ ON RAUM GOTO 1ØØØ,2ØØØ,3ØØØ,4ØØ
```

```
Ø,5ØØØ,6ØØØ
3Ø1 REM ABHAENGIG VON DER RAUMNUMMER
IN DEN BETREFFENDEN RAUM SPRINGEN

1ØØØ GRAPHICS Ø: REM RAUM 1
1Ø1Ø PRINT "ICH BIN IM WALD."
1Ø2Ø PRINT "ICH SEHE VIELE GROSSE BAEU
ME."
1Ø3Ø REM
1Ø4Ø REM WEITERE GEGENSTAENDE DRUCKEN
11ØØ PRINT "ICH KANN NACH SUEDEN, OSTE
N."
12ØØ IF EINGABE$="S" THEN RAUM=6:GOTO
1ØØ
121Ø IF EINGABE$="O" THEN RAUM=2:GOTO
1ØØ
126Ø IF LEN(EINGABE$)<3 THEN PRINT "DA
HIN FUEHRT KEIN WEG !":GOTO 1ØØ
13ØØ IF EINGABE$="INV" OR "INVENTUR" T
HEN GOTO 2ØØ
14ØØ IF VE$="UNT" AND OB$="BAE" THEN P
RINT "ICH SEHE NICHTS BESONDERES":GOTO
 1ØØ
141Ø IF VE$="NIM" AND OB$="BAE" THEN P
RINT "SO STARK BIN ICH NICHT.":GOTO 1Ø
Ø
1999 PRINT "ICH VERSTEHE NICHT, WAS DU
 MEINST.":GOTO 1ØØ
2ØØØ ab hier Behandlung Raum 2
3ØØØ ab hier Behandlung Raum 3
```

Wenn Sie sich obige Zeilen angesehen oder sogar in Ihren Atari eingegeben haben, werden Sie feststellen, daß auf eine noch einfachere Art ebenfalls ein funktionsfähiges Adventure entstehen kann, und Sie werden sich vielleicht fragen, warum wir nicht dieses Konzept verwendet haben, wo es zudem auch noch ohne lange Kommentare zu verstehen ist ?

Nun, einerseits werden besonders umfangreiche Adventures weitaus mehr Speicherplatz benötigen, denn wir würden es nicht vermeiden können, identische Routinen mehrmals aufzuführen, da wir, um bei einem bekannten Beispiel zu bleiben, die Flasche in jedem Raum ablegen (sechs identische Programmzeilen) und, nicht zu vergessen, auch wieder nehmen können müssen. Diesen zwölf Zeilen stehen in unserem Adventuresystem ganze zwei Programmzeilen gegenüber. Allein das rechtfertigt unsere Wahl, doch auch das Argument der Übersichtlichkeit und Editierfreundlichkeit ist schnell zu entkräften.
Stellen sie sich vor, bei der Erweiterung des Adventures werden zur Fortführung der Handlung weitere Flaschen benötigt, so daß wir aus diesem oder irgendwelchen anderen Gründen aus der Flasche lieber ein kleines Holzfaß machen wollen. Hatte Ihr Adventure zu diesem Zeitpunkt bereits vierzig Räume, dürfen Sie nun in entsprechend vielen Zeilen aus ©Nimm Flasche© ein ©Nimm Faß© machen. Entsprechendes gilt für ©Leg Flasche, Oeffne Flasche© usw.
Haben Sie sich hingegen an unser Adventuresystem gehalten, ändern Sie drei Buchstaben innerhalb unseres Strings in Zeile 18Ø, fertig !

Vergessen wir daher diese Methode und wenden wir uns wieder unserem Adventure zu. Wir lassen die Handlung ebenfalls in genau definierten Programmzeilen ablaufen, doch enthält nun jeder Block die Behandlung jeweils eines Verbes.

Unser Treiber hat bereits Verb- und Objektnummer ermittelt, daher können wir diese Blöcke gezielt anspringen. So ist es sinnvoll, allen Zeilen, die mit der Ausführung des Befehls ©untersuche© beschäftigt sind, Programmzeilen ab 5ØØØ zuzuteilen, die Aktion ©nim© wird entsprechend ab Zeile 6ØØØ realisiert.

Die Steuerung des Programmablaufes nimmt der Adventuretreiber anhand der Verbnummer in Zeile 22ØØ vor:

```
219Ø REM         UNT    NIM   LEG   OEF
BEN   ZER
22ØØ ON VN GOTO 5ØØØ,6ØØØ,7ØØØ,8ØØØ,9Ø
ØØ,1ØØØØ
```

Mit Programmzeile 5ØØØ beginnt nun der Block ©UNTERSUCHE©. Es ist leicht einzusehen, daß uns hier der größte Arbeitsaufwand abverlangt wird.
Nicht nur, daß auf *UNTERSUCHE HÜTTE* die Nachricht *IN DER HUETTE STEHT EIN ALTES REGAL.* ausgegeben und das Regal als sichtbares Objekt erscheinen muß. Nein, wir müssen auch sicherstellen, daß die Hütte an dem gerade besuchten Ort zur Verfügung steht, und gegebenenfalls eine Mitteilung wie *SO ETWAS SEHE ICH HIER NICHT.* an den Spieler übermitteln.

Diese Überlegungen zeigen bereits den Aufwand der Adventureprogrammierung, glücklicherweise lassen sich aber auch zahlreiche unterschiedliche Spielereingaben von ein und derselben Programmzeile beantworten.

So wird unser Programmteil, der die untersuchenden Handlungen des Spielers bearbeitet, zunächst prüfen, ob der Gegenstand, über den nähere Informationen gewünscht werden, entweder in Reichweite des Spielers, somit im gleichen Raum, oder gar in dessen Besitz ist, ansonsten kann sofort mit der Eingabe des nächsten Zuges begonnen werden:

```
5ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN GOTO 59ØØ
59ØØ REM GEGENSTAND NICHT VORHANDEN
599Ø PRINT "SO ETWAS SEHE ICH HIER NIC
HT !":GOTO 1Ø8Ø
```

DIE BEDINGUNGEN

Abgesehen von diesen zwei Bedingungen, lassen sich eine Reihe weiterer Voraussetzungen formulieren, die als Kriterium für die Ausführung einer Aktion herangezogen werden können, und ohne die ein Adventure nicht zu realisieren sein wird:

Objekt ist im Raum
Objekt wird vom Spieler mitgeführt
Objekt ist nicht im Raum
Flag ist gesetzt
Flag ist nicht gesetzt
Spieler ist in einem bestimmten Raum

Wie wir später noch sehen werden, erhebt diese Liste keinen Anspruch auf Vollständigkeit, denkbar ist beispielsweise eine Invertierung der letzten Bedingung, bestimmte Aktionen können in einem gewissen Raum nicht ausgeführt werden, wohl aber in jedem anderen Raum.

Einer Erläuterung bedürfen die an dieser stelle aufgetauchten Flags. Es handelt sich hierbei um weitere Signalschalter, die hier jedoch unmittelbar der Kontrolle des Spielverlaufes dienen werden, und die wir mittels eines weiteren Feldes realisieren wollen. Auch diese Feldvariablen sollen nur zwei Zustände annehmen können, entweder ist ein Flag gesetzt, dann soll der Inhalt der Variablen -1 sein, oder es ist nicht gesetzt, was einer Null entsprechen soll.
Praktisch nutzen werden wir sie zur Identifizierung bestimmter Zustände:

ist eine Tür geöffnet oder verschlossen ?
wurde ein Hindernis beseitigt oder nicht ?
soll ein Ungeheuer dem Treiben des Spielers ein Ende setzen, oder darf er weiterziehen ?

In der Programmpraxis wird die Formulierung einer einzigen Bedingung meist jedoch nicht zur eindeutigen Klassifizierung der Situation ausreichen, so daß logische Verknüpfungen mehrerer Bedingungen erforderlich werden, um zu gewährleisten, daß einige Handlungen nicht völlig zusammenhanglos und zu einem falschen Zeitpunkt durchgeführt werden.
Bei diesen logischen Operationen wird es sich um UND- beziehungsweise ODER- Verknüpfungen handeln, für unsere Adventurepraxis heißt das, entweder müssen beide (AND) oder nur die eine oder die andere (OR) von zwei oder mehreren Verknüpfungen erfüllt sein. Können wir nicht auf mehr als zwei Bedingungen verzichten, wird meist auch eine Klammerung unumgänglich sein. Ein Beispiel soll dies deutlich machen:

Denken wir an den Sprengstoff in unserer Kiste. Er soll dazu benutzt werden, allzu voreiligen Spielern die Freude am schnellen Weiterkommen zu vergällen.
Eine Warnung ist vor der Explosion aus Fairnessgründen jedoch durchaus angebracht, weshalb auf *Untersuche Sprengstoff* die Meldung *Er sieht sehr explosiv aus* ausgegeben werden soll. Wenn der Spieler dann noch versucht, ihn zu nehmen, trägt er selbst die Schuld an seinem Ende.

Hat der Spieler tatsächlich obige Eingabe gemacht, wird der Treiber die Nummern 1 (untersuche) und 9 (Sprengstoff) zur Verfügung stellen. Unser Problem liegt nun darin, daß wir uns nicht mit der Überprüfung eines Raumes zufrieden geben können, denn wir dürfen nicht vergessen, daß der Spieler zwar den Sprengstoff nicht greifen und mitnehmen kann, wohl aber die Holzkiste, und wenn er die Kiste mit sich führt, hat er auch den Sprengstoff bei sich. Prüfen wir daher, ob der Spieler den Sprengstoff gemeint hat, ***UND***

gleichzeitig die Holzkiste (Objekt Nummer 7) im gerade besuchten Raum ***ODER*** im Inventory des Abenteurers ist. Der eigentliche Handlungsgegenstand (Nummer 9) steht nicht zur Verfügung, nach Zeile 5ØØØ wird der Programmablauf daher mit Zeile 59ØØ fortgesetzt werden.

So muß die Ausführung des Befehls *Untersuche Sprengstoff* folgendermaßen formuliert werden:

```
59Ø4 IF N=9 AND (OB(7)=SP OR OB(7)=-1)
 THEN PRINT "ER SIEHT SEHR EXPLOSIV AU
S !":GOTO 1Ø8Ø
```

DIE AKTIONEN

Das vorangegangene Beispiel hat uns die Ausführung der wohl einfachsten Aktion, die Ausgabe einer Mitteilung an den Spieler, deutlich gemacht.
Meist ist dieser Print - Befehl jedoch nur Beiwerk zu anderen Handlungen, wie Manipulationen von OB(), um dem Abenteurer in einer Rückmeldung die Ausführung des Befehles zu bestätigen.

Im Gegensatz dazu bedeuten Befehle wie Nimm, Leg, Zerstöre und auch Öffne für einen Gegenstand immer eine Ortsveränderung. Entweder wird er neuerdings vom Spieler mitgeführt, oder er gehört nun zum Inventar eines bestimmten Raumes, vielleicht verschwindet dieser Gegenstand auch ganz aus dem Spiel. Stellen Sie sich nur die Verblüffung des Spielers von Goldrausch vor, wenn der Bär den Honig offensichtlich genossen hat, die Flasche aber immer noch irgendwo zu finden ist !

Eine andere, ebenfalls häufig benötigte Aktion ist das Setzen oder Löschen der zuvor erwähnten Flags, um

stufenweise die Voraussetzungen für spätere Handlungen zu schaffen.

Vergessen dürfen wir auch auf keinen Fall bedingte Ortswechsel des Spielers, welche durch Zauberei oder harmlose Aktionen ausgelöst werden. Eine spätere Erweiterung von Goldrausch könnte aus der Höhle des Bären ein riesiges Labyrinth von Gängen machen, und nach *Untersuche Höhle* befindet sich der Spieler mitten in der Höhle, obwohl er diese nicht durch eine Bewegung in die entsprechende Himmelsrichtung betreten hat. Aber durch den erhöhten Programmieraufwand wird unser Adventure nur realistischer werden, denn wie soll man eine Höhle gründlich untersuchen, ohne sie zu betreten ?

Ähnlich wie bei den Bedingungen, können wir auch hier eine Liste möglicher Aktionen zusammenstellen, wobei allerdings wiederum kein Anspruch auf Vollständigkeit erhoben wird. Jedoch handelt es sich dabei um einen ausreichenden Grundstock, denn es ist möglich, allein mit den vorgestellten Bedingungen und Aktionen gute Adventures zu schreiben.

Mitteilung an den Spieler ausgeben
Objekt verschwindet
Objekt kommt ins Inventory
Objekt erscheint neu im Raum
Flag wird gesetzt
Flag wird gelöscht
Spieler wird in anderen Raum versetzt

PROGRAMMIERUNG DER BEFEHLSAUSFÜHRUNG

Im folgenden finden Sie Hinweise zu Besonderheiten, die bei einzelnen Befehlen beachtet werden müssen. Dabei werde ich nur einige typische Programmzeilen erläutern, entnehmen Sie die restlichen Zeilen bitte dem Listing der Miniversion von Goldrausch am Ende dieses Kapitels.

Aktion: Untersuche

Zunächst steht fest, daß das Ergebnis der Untersuchung eines Gegenstandes immer in Form eines Satzes mitgeteilt wird.
Häufig wird es sich dabei um ein und denselben Satz handeln (Ich sehe nichts besonderes), weshalb wir die Mitteilungen an den Spieler in solchen Fällen nicht direkt ausgeben, sondern wir weisen den Text bei Programmstart einer Variablen zu und drucken deren Inhalt zu gegebener Zeit aus.
Da unser Editor bei der Eingabe von BASIC - Zeilen nur ungefähr 12Ø Zeichen problemlos annimmt, diese Zeilen neben den Informationen aber auch noch die Bedingungen enthalten müssen, wird es sich manchmal nicht vermeiden lassen, auch einige Meldungen vorzubereiten, an denen nur ein einziges Mal Bedarf besteht.

```
6ØØ REM ***************** MITTEILUNGEN
6Ø1 M1$="ICH SEHE NICHTS BESONDERES."
6Ø2 M2$="SO STARK BIN ICH NICHT."
6Ø3 M3$="WIE STELLST DU DIR DAS VOR ?"
6Ø4 M4$="DER BAER GREIFT SICH DEN HONI
G, UND"
6Ø5 M5$="VERSCHWINDET IN DER TIEFE DER
 HOEHLE."
```

Bereits erläutert wurde Zeile 5ØØØ, sie stellt für den gerade aktuellen Gegenstand durch Überprüfung des Wertes in OB() sicher, daß das Objekt sich in der Nähe des Spielers befindet.
Trifft dies zu, wird entsprechend der Objektnummer N die für diesen Gegenstand zuständige Programmzeile ermittelt. Sind alle dort programmierten Bedingungen erfüllt, wird der Rest der Zeile abgearbeitet und der Programmlauf anschließend mit dem Neuaufbau des Bildschirmbildes fortgesetzt:

```
5ØØ2 IF N=1 THEN PRINT M1$:GOTO 1Ø8Ø
5ØØ3 IF N=3 THEN PRINT M1$:GOTO 1Ø8Ø
5ØØ4 IF N=4 THEN PRINT "IN EINER ECKE
STEHT EIN REGAL.":OB(8)=SPIELER:GOTO 1
Ø8Ø
```

Wird die Holzhütte untersucht (N=4), entdeckt der Spieler das Regal. Dieses war bislang in Raum Ø, dem Lagerraum, und wird nun im Raum des Spielers positioniert, weshalb der Adventuretreiber beim anschließenden Durchlaufen der Zeilen 111Ø bis 12ØØ unter anderem auch *ein klappriges Regal* ausgibt.

Nun ist es aber, wie im Fall des Regales, denkbar, daß ein Objekt auftaucht, welches der Spieler anschließend an sich nimmt.
Dann wäre eine Formulierung wie in 5ØØ4 nicht mehr ausreichend, denn die Flasche würde bei jeder weiteren Untersuchung des Regals dem Inventory automatisch entnommen und wieder in den Raum gestellt werden.

Diesen Fehler können wir nur vermeiden, wenn wir uns zuvor vergewissern, daß die Flasche noch nicht im Spiel war, somit also noch immer im Lagerraum ist.

```
5ØØ8 IF N=8 AND OB(5)=Ø THEN PRINT "AU
F DEM REGAL STEHT EINE KORBFLASCHE.":O
```

```
B(5)=SPIELER:GOTO 1Ø8Ø
5ØØ9 IF N=8 AND OB(5)<>Ø THEN PRINT M1
$:GOTO 1Ø8Ø
```

Bei allen weiteren Untersuchungen des Regals wird der Spieler nichts besonderes mehr feststellen (Zeile 5ØØ9).

Ein weiteres Beispiel soll den Gebrauch der Flags zeigen. Hat der Spieler die Schatztruhe gefunden, ist diese zunächst mittels einer Eisenkette verschlossen. Nachdem er dieses Hindernis beseitigt hat, muß die Truhe selbstverständlich erst geöffnet werden, ehe der wertvolle Inhalt sichtbar wird. Um diese drei Zustände dem Spieler deutlich zu machen, wäre es möglich, weitere Gegenstände, wie eine mit einer Eisenkette verschlossene Truhe, eine unverschlossene Truhe, und eine geöffnete Truhe, einzuführen. Dieser recht hohe Programmieraufwand, der nebenbei natürlich wiederum den zur Verfügung stehenden Speicherplatz einschränkt, lässt sich durch den Einsatz von Signalschaltern umgehen. Sinnvollerweise fertigen Sie während der Konstruktion Ihres Adventures eine Tabelle an, damit Sie auch wirklich die richtigen Flags umstellen:

Flag	Ø	-1
1	Kette heil	Kette zerstört
2	Truhe zu	Truhe geöffnet

Warum als Inhalt nur -1 und Ø, und warum nicht nur ein einziges Flag mit verschiedenen Ziffern für jeden möglichen Zustand vorgesehen worden ist, können Sie sich inzwischen selbst erklären.
Denn wir machen uns die Tatsache zunutze, daß für den BASIC - Interpreter ein wahrer Ausdruck immer durch eine -1 repräsentiert wird, und können somit unsere Programmzeilen wieder etwas kürzer halten, denn in den IF - Statements müssen wir keine Vergleichswerte angeben:

```
IF FL(1)=-1 entspricht IF FL(1)
IF FL(1)= Ø entspricht IF  NOT FL(1)
```

Für unser Beispiel gelten demnach folgende Zeilen:

```
5Ø11 IF N=11 AND  NOT FL(1)THEN PRINT
"SIE IST MIT EINER EISENKETTE VER-":PR
INT "SCHLOSSEN.":GOTO 1Ø8Ø
5Ø12 IF N=11 AND FL(1) AND  NOT FL(2)
THEN PRINT "AUSSEN SEHE ICH NICHTS BES
ONDERES.":GOTO 1Ø8Ø
5Ø13 IF N=11 AND FL(1) AND FL(2) AND O
B(12)=Ø THEN PRINT "SIE IST VOLLER SIL
BERMUENZEN.":OB(12)=SPIELER :GOTO 1Ø8Ø
```

Ein weiterer Punkt, dem wir besondere Beachtung schenken müssen, findet seinen Grund in der Auslegung unserer Wortanalyse.
Zur Ausstattung der ersten zwei Räume gehören die völlig identischen Objekte 1 und 2. Wird die Eingabe des Spieles überprüft, so wird als Objektnummer immer nur eine 1 ermittelt werden können, welche bei einem Aufenthalt des Spielers in Raum 2 nicht den Tatsachen entspricht.
Um Eingabefehler richtig quittieren zu können, hatten wir jedoch Programmzeile 5ØØØ eingesetzt, welche uns nun auch die Behandlung dieser Sonderfälle ermöglicht.
So lösen wir das Problem *Untersuche Baeume* für Raum 2 folgendermaßen:

```
59Ø1 1F N=1 AND SP1ELER=2 THEN PRINT M
1$:GOTO 1Ø8Ø
```

Zum Abschluss der Routine müssen wir eine letzte Zeile vorsehen, die, wenn in keiner der vorangegangenen Programmzeilen alle Bedingungen erfüllt waren, ohne jede Bedingung durchgeführt wird. Hiermit werden nicht

definierte Spieleingaben abgefangen, und wir verhindern, daß, wenn keine zutreffende Bedingung gefunden wurde, die Zeilen der anderen Verben abgearbeitet werden und somit der Programmlauf außer Kontrolle gerät. Selbst wenn wir eine Handlung, die für den Ablauf des Adventures nicht notwendig ist, vergessen, wird ein Spieler das niemals bemerken:

```
599Ø PRINT "SO ETWAS SEHE ICH HIER NIC
HT !":GOTO 1Ø8Ø
```

Aktion: NIMM

Genau so, wie der Spieler einen Gegenstand nur untersuchen kann, der sich in seiner Nähe befindet, so wird er ihn auch nur nehmen können, wenn der Gegenstand entweder im gleichen Raum wie der Spieler oder bereits in dessen Taschen ist.

```
6ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN GOTO 69ØØ
```

Danach wäre es an sich ganz einfach, den Befehl auszuführen, für eine Reihe von Objekten sollten wir dem Abenteurer seinen Wunsch jedoch versagen.
So wollen wir berücksichtigen, daß eine vollgefüllte Eisentruhe vermutlich zu gewichtig für einen einzelnen Menschen ist, ebenso können wir selbst dumme Versuche des Spielers wie *Nimm Baum* nicht unbeachtet lassen.

```
6ØØ1 IF N=1 THEN PRINT M2$:GOTO 1Ø8Ø
6ØØ5 IF N=11 THEN PRINT M2$:GOTO 1Ø8Ø
```

Natürlich kann es auch noch schlimmer kommen, denn wir müssen dem Spieler selbstverständlich eine ausreichende Anzahl von Fallen stellen, damit ihm der Sieg nicht zu einfach gemacht wird. So sollte laut unserem Drehbuch eine Berührung des Sprengstoffes zur Explosion und damit zum

Spielende führen. Gleiches gilt für den Fall, daß er versucht, sich der Nuggets zu bemächtigen, ohne dem Bären zuvor einen Leckerbissen angeboten zu haben. Ebenso fatal für den Spieler müsste sein Versuch sein, den Bären zu nehmen:

```
6Ø15 IF N=1 AND SPIELER=5 THEN MØ$="DE
R BAER HAT MICH ERSCHLAGEN.":GOTO 45ØØ
6Ø18 IF N=17 AND  NOT FL(3) THEN MØ$="
EIN BAER STUERZT SICH AUF MICH.":GOTO 4
5ØØ
69ØØ IF N=9 AND (OB(7)=SPIELER OR OB(7
)=-1) THEN MØ$="BEI DER BERUEHRUNG IST
 DER SPRENGSTOFF EXPLODIERT !":GOTO 45Ø
Ø
```

Anmerkung: In Zeile 45ØØ beginnt eine Routine, welche bei nicht gewonnenem Spiel aufgerufen wird. MØ$ ist in dieser Routine für eine erläuternde Nachricht vorgesehen.

Flag 3 wird gesetzt, sobald der Bär in den Besitz des Honigs gelangt.

Steht dem Versuch des Spielers, sich an irgendwelchen Gegenständen unseres Adventures zu bereichern, nichts im Wege, müssen wir dem betreffenden Objekt nur die neue Platznummer, eine -1, zuweisen:

```
6Ø1Ø IF N=5 THEN OB(5)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
```

Aktion: LEG

Irgendwann jedoch wird er sich dieser Objekte auch wieder entledigen wollen, entweder, weil wir als Programmierer seine Tragfähigkeit begrenzt haben, oder weil er sie irgendwie benutzen will.
In der Praxis wirft die Leg weg - Routine kaum Probleme auf, darüber hinaus überzeugt sie durch ihre Kürze.
Denn die einzige Voraussetzung ist, daß der wegzulegende Gegenstand im Besitz des Spielers ist, was für alle Objekte in Zeile 7ØØØ getestet wird:

```
7ØØØ IF OB(N)<>-1 THEN PRINT "SO ETWAS
 HABE ICH DOCH GAR NICHT !":GOTO 1Ø8Ø
79ØØ OB(N)=SPIELER:PRINT "O.K.":GOTO 1
Ø8Ø
```

Irrtümer des Spielenden werden sofort erkannt und entsprechend quittiert. War der Gegenstand im Inventory, wird ihm mit einer Änderung des Inhaltes von OB() ein neuer Aufenthaltsort zugewiesen.

Prinzipiell reichen diese beiden Zeilen aus, es kann jedoch erforderlich sein, neben dem Positionswechsel weitere Aktionen durchzuführen. So hatten wir für Goldrausch vorgesehen, daß der Bär sich die Flasche greift und damit in den Tiefen der Höhle verschwindet.
Diese Aktion darf natürlich nur durchgeführt werden, wenn der Spieler sich gerade in Raum 5 befindet, an allen anderen Orten wird der Vorgang ebenfalls durch Zeile 79ØØ ausgeführt.

```
7Ø2Ø IF N=5 AND SPIELER=5 THEN OB(5)=Ø
:FL(3)=-1:PRINT M4$:PRINT M5$:OB(14)=Ø
:GOTO 1Ø8Ø
```

Aktion: OEFFNE

Gleich zu Beginn dieses Programmteiles steht der nun schon bekannte Test, um technische Fehler zu vermeiden. Wiederum ist es nötig, sowohl den Raum als auch das Inventory zu überprüfen, schließlich wird der Spieler eine zu öffnende Tür nicht erst nehmen müssen, was aber für eine Flasche durchaus Bedingung sein kann:

```
8ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN PRINT "SO ETWAS IST HIER NICHT.":G
OTO 1Ø8Ø
```

Die Aktion selbst wird meist aus dem Setzen eines Flags und der Ausgabe einer Mitteilung bestehen:

```
8Ø25 IF N=11 AND FL(1) THEN PRINT "O.K
. - DER DECKEL KLAPPT NACH HINTEN.":FL
(2)=-1:GOTO 1Ø8Ø
```

Nicht vergessen dürfen wir unwichtige Aktionen, die aber von der Lage der Dinge her möglich sind. Als Beispiel mag noch einmal unsere Flasche dienen. Ein logisch denkender Abenteurer wird sie vor einer näheren Untersuchung erst öffnen wollen und wäre sicher überrascht, wenn ihm dies nicht gelingt, bloß weil wir es für unrelevant in Bezug auf den weiteren Spielverlauf gehalten haben:

```
8Ø1Ø IF N=5 THEN PRINT "O.K.":GOTO 1Ø8
Ø
```

Der Abschluss dieses Blockes gewährleistet, daß neben den technischen Falscheingaben auch inhaltliche Unmöglichkeiten wie *Öffne Honig* erkannt werden.

```
8999 PRINT "ICH VERSTEHE NICHT, WAS DU
 MEINST.":GOTO 1Ø8Ø
```

ZUSAMMENFASSUNG

Ich hoffe, daß Ihnen mit den Erläuterungen zu diesen wohl häufigsten Handlungen die Ausführung der eigentlichen Spielprogrammierung klargeworden ist. Prinzipiell lässt sich zusammenfassend sagen, daß die einzelnen Aktionen durch zwei Programmzeilen umklammert werden.
So wird zu Beginn eines Blockes getestet, ob von der technischen Seite her die Durchführung der Aktion möglich wird, abschließend wird sichergestellt, das der Programmlauf selbst bei Auftreten irgendeines Fehlers korrekt fortgesetzt wird.
Das Erkennen der richtigen Voraussetzungen für die Durchführung eines Befehls ist dabei, ebenso wie die Aktion selbst, meist eine recht einfach zu programmierende Angelegenheit, die bei unserem Adventuresystem zudem noch standardisiert wurde. Folgende Zusammenstellung soll Ihnen bei der Realisation Ihrer ersten eigenen Adventures eine Hilfe sein:

BEDINGUNG	***IM BASICPROGRAMM***
Objekt ist im Raum	OB(objekt)=SP
Objekt wird vom Spieler mitgeführt	OB(objekt)=-1
Objekt ist nicht im Raum	OB(objekt)<>SP
Flag ist gesetzt	FL(x)=-1
Flag ist nicht gesetzt	FL(x)=O
Spieler ist in einem bestimmten Raum	SP=Raumnummer

AKTIONEN	***IM BASICPROGRAMM***
Mitteilung an den Spieler ausgeben	PRINT" oder M$
Objekt verschwindet	OB(Objekt)=O
Objekt kommt ins Inventory	OB(Objekt)=-1
Objekt erscheint neu im Raum	OB(Objekt)=SP
Flag wird gesetzt	FL(x)=-1
Flag wird gelöscht	FL(X)=O
Spieler wird in anderen Raum versetzt	SP=Raumnummer

DIE LETZTEN SCHRITTE

Bevor unser Adventure den in Kapitel 2 entwickelten Vorstellungen entspricht, müssen wir uns noch Gedanken zu drei weiteren Programmteilen machen.
So ist es für einen einwandfreien Spielablauf unbedingt erforderlich, dem Spieler die Möglichkeit zu einer schnellen Inventur in die Hand zu geben.

Es wäre durchaus möglich, diese Aktion auf gleiche Art und Weise wie die Behandlung der übrigen Verben durchzuführen. Da es sich hierbei jedoch um eine grundsätzliche Funktion der Abenteuerprogramme handelt, werden wir unseren Treiber in geeigneter Weise ergänzen.

Inventur

Inventur gehört, wie auch die später zu erstellenden Routinen zum Abspeichern und Laden des Spielstandes, zur Befehlsgruppe der sogenannten Ein - Wort - Befehle. Für diesen Programmteil haben wir zwischen den Programmzeilen 148Ø (Ende der Bewegungsroutine) und 2ØØØ (Analyse der Eingabe) ausreichend Platz gelassen.

Um unseren Treiber nicht unnötig lange damit zu beschäftigen, einen jeden Spielerzug mit den Befehlen dieser Sondergruppe zu vergleichen, entscheiden wir zunächst, ob es sich um eine dieser speziellen Anweisungen oder um eine reguläre Eingabe handelt.
Die übliche Spielereingabe wird eine Länge von mindestens neun Buchstaben haben, gehen wir daher davon aus, daß es sich, wenn die Länge der Eingabe geringer ist, um einen Ein - Wort Befehl handelt:

```
148Ø IF LEN(EINGABE$)>8 THEN GOTO 2ØØØ
```

Handelte es sich bei der Eingabe des Spielers nicht um eine reguläre Verb/Objekt - Kombination, dann sollen alle Zeilen, in denen die Behandlung eines EW - Befehles beginnt, der Reihe nach durchlaufen werden, bis schließlich eine Übereinstimmung der ersten drei Buchstaben festgestellt wird.

In der Praxis besteht die Ausführung der Inventur aus einer einfachen Schleife, welche alle Gegenstände des Adventures, deren Aufenthaltsort mit -1 gekennzeichnet ist, auf dem Bildschirm ausdruckt:

```
1499 REM ************** START INVENTUR
15ØØ IF EINGABE$(1,3)(>"INV"·THEN GOTO
 2ØØØ
151Ø PRINT "ICH TRAGE FOLGENDES MIT MI
R:"
152Ø FOR I=1 TO AO
153Ø IF OB(I)=-1 THEN GOSUB 3ØØ+I:PRIN
T T$
154Ø NEXT I
155Ø GOTO 1Ø8Ø
156Ø REM *************** ENDE INVENTUR
```

Die Titel

Rein technisch gesehen haben wir unser Programm mit diesen Zeilen fertiggestellt. Alle gewünschten Funktionen stehen zur Verfügung und einem Spiel steht nichts mehr im Wege. Doch wie lange soll der Abenteurer in unserer Welt umherirren, wann hat er sein Ziel erreicht ?

Handlungsmäßig ist die Miniversion von Goldrausch beendet, wenn sowohl die Nuggets als auch die Silberstücke sich im Besitz des Spielers befinden.

Eine einzige BASIC - Zeile reicht aus, um in diesem Falle alle weiteren Eingaben zu unterbinden und einen Programmteil anzuspringen, welcher dem Spieler in angemessener Form seinen Erfolg mitteilt und das Spiel beendet.

```
134Ø IF OB(12)=-1 AND OB(17)=-1 THEN G
OTO 48ØØ
```

Für Adventurespiele typisch ist jedoch ein völlig anderes, weniger ruhmreiches Ende. Da auch Spieler unserer Programme meistens gezwungenermaßen ihr Spiel etwas früher beenden werden, programmieren wir ein weiteres Endbild ab Zeile 45ØØ.
Bei der Ausführung der spielbeendenden Handlungen hatten wir ja bereits eine entsprechende Nachricht vorbereitet, so daß der Spieler zumindest nicht über die Ursache seines Scheiterns im Unklaren gelassen werden muß.

```
45ØØ GRAPHICS Ø:REM ****** SPIELER TOT
46ØØ PRINT "AUCH DAS NOCH !":PRINT :PR
INT MØ$
461Ø PRINT :PRINT "ICH BIN TOT !":PRIN
T
462Ø PRINT "SOLL ICH ES NOCH EINMAL VE
RSUCHEN ";:INPUT EINGABE$
463Ø IF EINGABE$(1,1)="J" THEN CLR :GO
TO 1ØØ
464Ø GOTO 196Ø
```

Dabei erlaubt es uns die Einbeziehung der individuellen Nachricht in MØ$, dieses Titelbild für jede Situation und für jedes Adventure zu verwenden. Änderungen werden, sofern sie überhaupt erforderlich sind, nur wenig Aufwand verursachen. Aus diesem Grunde können obige Zeilen, ebenso wie die Siegesnachricht ab 48ØØ, als weitere Ergänzungen unseres Adventuretreibers betrachtet werden.

Bedenken wir weiterhin, daß es für Adventures durchaus nicht typisch ist, sie zu laden und in einigen wenigen Stunden durchzuspielen, und programmieren wir daher zu guter Letzt noch ein reguläres Programmende.

Auch wenn sich solche Software auf dem Markt befindet, es zeugt nicht von der Benutzerfreundlichkeit eines Programmes, wenn es nur durch Drücken der Break- oder Resettaste verlassen werden kann.

Fügen wir daher einen weiteren Ein - Wort Befehl in unser System ein:

```
15ØØ IF EINGABE$(1,3)<>"INV"·THEN GOTO
 195Ø
195Ø IF EINGABE$(1,3)<>"END" THEN GOTO
 2ØØØ
196Ø GRAPHICS Ø:PRINT "DER AUTOR WUENS
CHT IHNEN MEHR GLUECK":PRINT "FUERS NA
ECHSTE MAL !":PRINT :PRINT :PRINT :END
```

Sinngemäß waren das die letzten Zeilen unseres Adventures, alle denkbaren Versionen des Spielendes haben wir realisiert. Doch wie nimmt sich dagegen der Beginn unseres Werkes aus ?

Starten wir unser Programm so, wie es ist, besteht dessen erste Handlung darin, die Variablen vorzubereiten, eine Arbeit, die besonders bei größeren Adventures den Gedanken an einen Absturz des Programmes aufkommen lässt. Anschließend befindet man sich mitten im Programm, und ein unerfahrener Adventureneuling sieht dann möglicherweise den Sinn des Spieles darin, eine Sight Seeing Tour durch eine elektronisch simulierte Welt anzustellen.

Schenken wir daher einer alten Volksweisheit Glauben, die besagt, daß der erste Eindruck oft entscheidend ist.

Machen wir uns die Mühe, unser Programm, bevor wir es als fertig bezeichnen, mit einem, oder besser noch, mit mehreren Titeln zu versehen.

Dem ersten Titel wird dabei die Aufgabe zufallen, in knapper, aber ansprechender Form, Aufklärung über die im Speicher befindlichen Bytes zu geben, Programmtitel und -gattung zu nennen sowie Auskunft über den Autoren zu geben.

Das nächste Bild bringen wir auf den Schirm, bevor die Variablen initialisiert werden. Während dieses Zeitraumes kann dieser Titel Auskunft über die dem Spieler gestellte Aufgabe geben, und ihn in die Handlung einführen.

Gehört das Herausfinden der gestellten Aufgabe allerdings mit zur Spielidee, wird diese Tafel fehlen, und an den Programmtitel schließt sich gleich eine Erklärung der Adventurespiele an, die ansonsten als drittes Titelbild implementiert werden muß.

Denn wie war es, als Sie Ihr erstes Adventure in die Hand und in den Speicher bekamen, kannten Sie Sinn und Ziel dieser Programme, oder saßen Sie rat- und tatenlos vor Ihrem Computer ?

Vergessen wir doch nicht, daß es immer wieder Anfänger geben wird. Anfänger, die froh sind, wenn sie nach dem Einschalten des Rechners ein Programm ohne Fehlermeldungen laden und starten können. Eine grundsätzliche Spielanweisung bedeutet doch nur geringen zusätzlichen Aufwand, machen wir uns daher ans Werk und rechtfertigen ein mögliches Fehlen nicht etwa damit, daß das Herausfinden der Spielregeln bereits das erste beabsichtigte Hindernis wäre.

Für diese Aufgaben stehen uns die Programmzeilen 1Ø - 1ØØ, sowie 7ØØ - 9ØØ zur Verfügung.

Sinnvollerweise bauen wir die allgemeinen Erläuterungen dabei als Unterprogramm ein, was uns den weiteren Ausbau des Adventuretreibers um den Befehl INStruktionen erlaubt. Der Abenteurer kann sich nun jederzeit ohne einen Abbruch des Spieles die allgemeinen Regeln in die Erinnerung zurückrufen.

HINWEISE ZUM FOLGENDEN LISTING

Nachdem Sie die folgenden Zeilen in Ihren Atari eingegeben haben, steht Ihnen die lauffähige Miniversion von Goldrausch zur Verfügung. Es gilt jedoch zu beachten, daß es sich nur um zusätzliche Programmzeilen handelt, alle im vorangegangenen Text erläuterten Zeilen müssen sich bereits im Speicher befinden.

Sie werden sehen, daß mit den bisher vorgestellten Techniken bereits anspruchsvolle Adventures entwickelt werden können, die den Vergleich mit entsprechender käuflicher Software nicht zu scheuen brauchen.

Es liegt dann an Ihnen, ob Sie das Buch zunächst beiseite legen und ein eigenes Abenteuerspiel entwickeln oder lieber weitere Features kennenlernen wollen, die aus einem guten ein perfektes Adventure machen können.

```
1 REM GOLDRAUSCH, MINIVERSION; ATARI
2 REM (C) WALKOWIAK, OKTOBER 1984
3 REM --------------------------
1Ø GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
4,1,Ø:FOR I=1 TO 9:PRINT :NEXT I
2Ø PRINT "
  "
3Ø PRINT "         G O L D R A U S C H
  "
4Ø PRINT "
  "
5Ø PRINT "            MINIVERSION 1.Ø
  "
6Ø PRINT "         (C) 1984 BY WALKOWIA
K "
7Ø PRINT "
  ";
8Ø FOR I=1 TO 3ØØØ
9Ø NEXT I
1ØØ REM ********* DATEN DES ADVENTURES
11Ø AR=6:AO=18
15Ø DIM RAUM$(8Ø),DURCHGANG(AR,6),RICH
TUNG$(36),EINGABE$(2Ø),T$(4Ø),OB(AO),L
EERZEILE$(38),DZ$(5Ø),ZENDE$(3),FL(5)
16Ø DIM VERBEN$(4Ø),EVERB$(2Ø),EOBJEKT
$(2Ø),OBJEKTE$(12Ø),MØ$(4Ø),M1$(3Ø),M2
$(3Ø),M3$(3Ø),M4$(38),M5$(38)
19Ø GOTO 6ØØ
318 T$="EINE EISENSTANGE":RETURN
41Ø DATA 1,2,2,3,Ø,Ø,3,Ø,Ø,Ø,Ø,Ø,Ø,5,Ø,4
,6,5,Ø
699 REM ********* 2. TITEL: EINLEITUNG
7ØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
 4,1,Ø
71Ø PRINT "Herzlich Willkommen zur Min
iversion":PRINT "         von GOLDRAUS
CH !":PRINT
72Ø FOR I=1 TO 37:PRINT CHR$(13);:NEXT
 I:PRINT
73Ø PRINT "Auf der Suche nach dem Glue
ck in der":PRINT "neuen Welt begegnete
```

```
n Sie vor einigen"
74Ø PRINT "Tagen einem alten todkranke
n Mann,":PRINT "dem Sie in seinen letz
ten Stunden"
75Ø PRINT "Beistand leisteten.":PRINT
"Aus Dankbarkeit berichtete er Ihnen"
76Ø PRINT "von seiner Goldmine und den
 dort ver-"
77Ø PRINT "steckten Resten seines Verm
oegens.":PRINT "Zahllosen Gefahren wid
erstanden Sie"
78Ø PRINT "auf dem Weg dorthin; bald w
erden Sie":PRINT "Ihr Ziel erreicht ha
ben, und es wird"
785 PRINT "sich zeigen, ob der Alte di
e Wahrheit":PRINT "gesprochen oder im
Fiebertraum gere- det hatte.":PRINT
79Ø FOR I=1 TO 37:PRINT CHR$(13);:NEXT
 I:PRINT
795 PRINT "WUENSCHEN SIE RATSCHLAEGE F
UER IHR             WEITERES VORGEHEN
";:INPUT EINGABE$
798 IF EINGABE$(l,l)="J" THEN GOSUB 8Ø
Ø
799 GOTO 9ØØ
8ØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
 4,1,Ø
82Ø PRINT "           ATARI - VENTURES
          ":PRINT "     (C) 1984 BY JO
ERG WALKOWIAK     ":REM INVERS
83Ø PRINT :PRINT "Stellen Sie sich ein
en Roboter vor,   den Sie mit zahlreic
nen Kommandos"
84Ø PRINT "steuern koennen. Ich bin di
eser Ro-   boter, und ich werde mich f
uer Sie"
85Ø PRINT "den Gefahren der verwegenst
en Aben-   teuer   aussetzen."
855 PRINT "Damit Sie mich sinnvoll agi
eren las-  sen koennen, werde ich Ihne
n die":PRINT "Situation in der ich mic
```

```
h gerade"
86Ø PRINT "befinde, jeweils genau besc
hreiben.":PRINT "Anschliessend sagen S
ie mir mit zwei"
863 PRINT "Worten wie zum Beispiel UNT
ERSUCHE "
87Ø PRINT "TUER, NIMM MESSER, was ich
tun soll.":PRINT :PRINT "Darueber hina
us verstehe ich die Be-  fehle:";
88Ø PRINT"    INVENTUR und ENDE.":PRIN
T
89Ø PRINT "BITTE DRUECKEN SIE <RETURN>
 UND ...";:INPUT EINGABE$:GRAPHICS Ø:R
ETURN :REM INVERS
1ØØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:GOTO 1Ø2Ø
1Ø3Ø T$=""
15ØØ IF EINGABE$(1,3)<>"INV" THEN GOTO
·19ØØ
19ØØ IF EINGABE$(1,3)<>"INS" THEN GOTO
195Ø
191Ø GOSUB 8ØØ
192Ø GOTO 1Ø8Ø
1999 REM ************* EINGABE TRENNEN
2Ø6Ø VN=Ø:N=Ø
2Ø7Ø EVERB$=EVERB$(1,3)
2Ø8Ø EOBJEKT$=EOBJEKT$(1,3)
48ØØ GRAPHICS Ø
481Ø PRINT "HERZLICHEN GLUECKWUNSCH !"
482Ø PRINT :PRINT :PRINT "SIE HABEN DI
E IHNEN GESTELLTE AUFGABE":PRINT :PRIN
T "GELOEST, UND DUERFEN SICH AN EINEM"
483Ø PRINT "ANDEREN ADVENTURE VERSUCHE
N."
484Ø PRINT :PRINT :PRINT :END
4999 REM ***** SPIELERZUEGE AUSFUEHREN
5ØØ5 IF N=5 THEN PRINT "DIE FLASCHE IS
T GEFUELLT MIT HONIG.":GOTO 1Ø8Ø
5ØØ6 IF N=6 THEN PRINT M1$:GOTO 1Ø8Ø
5ØØ7 IF N=7 AND OB(9)=Ø THEN PRINT "IN
 DER KISTE LIEGT SPRENGSTOFF.":GOTO 1Ø
```

```
8Ø
5Ø1Ø IF N=1Ø THEN PRINT "IN DEM ERDLOC
H LIEGT EINE EISENTRUHE.":OB(11)=SPIEL
ER:GOTO 1Ø8Ø
5Ø14 IF N=12 THEN PRINT "GENAU DAS SUC
HE ICH !!":GOTO 1Ø8Ø
5Ø15 IF N=13 THEN PRINT "ICH HABE EINE
N BAEREN AUFGESCHRECKT.":OB(14)=SPIELE
R:GOTO 1Ø8Ø
5Ø17 IF N=15 THEN PRINT "DAZWISCHEN IS
T EIN ERDLOCH.":OB(1Ø)=SPIELER:GOTO 1Ø
8Ø
5Ø18 IF N=16 THEN PRINT "SIE SEHEN SEH
R STABIL AUS.":GOTO 1Ø8Ø
5Ø19 IF N=17 THEN PRINT "ES HANDELT SI
CH UM PURES GOLD !":GOTO 1Ø8Ø
59Ø2 IF N=6 AND OB(5)=-1 THEN PRINT "E
R IST SUESS UND GUT.":GOTO 1Ø8Ø
59Ø3 IF N=6 AND OB(5)<>-1 THEN PRINT "
ICH HABE KEINEN HONIG !":GOTO 1Ø8Ø
591Ø IF N=1 AND SPIELER=5 AND OB(14)=5
 THEN PRINT "ICH SCHEINE SEINEN APPETI
T ANZUREGEN.":GOTO 1Ø8Ø
6ØØ2 IF N=3 THEN PRINT M2$:GOTO 1Ø8Ø
6ØØ3 IF N=4 THEN PRINT M3$:GOTO 1Ø8Ø
6ØØ4 IF N=8 THEN PRINT M2$:GOTO 1Ø8Ø
6ØØ6 IF N=1Ø THEN PRINT M3$:GOTO 1Ø8Ø
6ØØ7 IF N=13 THEN PRINT M3$:GOTO 1Ø8Ø
6ØØ8 IF N=15 THEN PRINT M2$:GOTO 1Ø8Ø
6Ø11 IF N=6 THEN OB(5)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
6Ø12 IF N=7 THEN OB(7)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
6Ø14 IF N=12 THEN OB(12)=-1:PRINT "O.K
 ":GOTO 1Ø8Ø
6Ø16 IF N=16 THEN OB(18)=-1:PRINT "O.K
 ":GOTO 1Ø8Ø
6Ø17 IF N=17 AND FL(3) THEN PRINT "O.K
 ":OB(17)=-1:GOTO 1Ø8Ø
6999 PRINT "SO ETWAS SEHE ICH HIER NIC
HT !":GOTO 1Ø8Ø
```

```
7Ø1Ø IF N=6 AND SPIELER=5 THEN OB(6)=Ø
:FL(3)=-1:PRINT M4$:PRINT M5$:OB(14)=Ø
:GOTO 1Ø8Ø
8ØØ5 IF N=4 AND SPIELER=3 THEN PRINT "
DIE HUETTE STAND BEREITS OFFEN.":GOTO
1Ø8Ø
8Ø2Ø IF N=11 AND  NOT FL(1) THEN PRINT
 "DAS LAESST DIE KETTE NICHT ZU !":GOT
O 1Ø8Ø
9ØØØ REM *************** VERB=BENUTZE
9Ø1Ø IF N=16 AND SPIELER=4 THEN PRINT
"DIE KETTE ZERSPRINGT.":FL(1)=-1:GOTO
1Ø8Ø
9999 PRINT "ICH VERSTEHE NICHT, WAS DU
 MEINST !":GOTO 1Ø8Ø
1ØØØØ IF N=18 AND SPIELER=4 AND OB(18)
=-1 THEN PRINT "DIE KETTE ZERSPRINGT."
:FL(1)=-1:GOTO 1Ø8Ø
1ØØ1Ø IF N=18 AND SPIELER=4 AND OB(18)
<>-1 THEN PRINT "ICH HABE KEINE EISENS
TANGE.":GOTO 1Ø8Ø
1Ø999 PRINT "ICH VERSTEHE NICHT, WAS D
U MEINST !":GOTO 1Ø8Ø
```

4. KAPITEL
- PERFEKTIONIERUNG -

VOM GUTEN ZUM PERFEKTEN ADVENTURE

Wenn Sie mit der Miniversion von Goldrausch gespielt oder selber ein Adventure auf der Basis der bislang erarbeiteten Techniken erstellt haben, werden Sie mit mir übereinstimmen, daß der Standard gängiger Programme der Gattung Adventures durchaus erreicht wurde.

Dennoch lassen sich ab und an Abenteuerspiele finden, die mit weiteren Extras aufwarten. Um sich über die Konkurrenz erheben zu können, wurde entweder ihre Benutzerfreundlichkeit erhöht, oder es fanden bei der Realisierung des Spieles auch die kleinen Nebensächlichkeiten des täglichen Lebens Verwendung und erschweren dem Spieler das Leben nun zusätzlich.
Darüber hinaus existieren bereits einige Programme, die mit entsprechendem Werbeaufwand auf ihre noch nie dagewesenen Besonderheiten aufmerksam machen.
So wird auch dem Gehör des Spielers etwas geboten, von einzelnen Beeps und Pieps, die einen verstandenen oder aber einen undurchführbaren Befehl signalisieren über die akustische Untermalung sichtbarer Objekte bis hin zur Nachahmung der menschlichen Stimme. Leider ist der spielerische Nutzen all dieser Extras gleich null, und man sollte es sich mehr als zweimal überlegen, ob man sein Geld für speicherfüllende Effekthascherei opfert oder lieber eine um den gesparten Speicherplatz vergrößerte Adventurewelt vorzieht.

BENUTZERFREUNDLICHKEIT

wird für Geschäftsprogramme immer wieder gefordert, warum nicht auch für Spiele ?

Da gibt es doch tatsächlich Adventures, die dem Spieler nicht einmal mitteilen, auf welchen Wegen er einen gefährlichen Ort wieder verlassen kann. Sie warten nur darauf, ihn durch die nach Eingabe der verschiedensten Himmelsrichtungen auftauchende Mitteilung ©*In diese Richtung kannst du nicht. - You can©t go in that direction.*© zur Verzweiflung zu treiben.

Gut, daß der Spieler sich die Haare raufen darf und soll, damit bin ich einverstanden, doch scheinen mir diese Programme eher dazu geschrieben worden zu sein, um mit dem Werbespruch ©Wir garantieren Ihnen, daß Sie Monate brauchen, um alle Orte dieses Adventures kennenzulernen !© an den Adventurefan gebracht zu werden.

Diese Kritik kann unser Adventuresystem glücklicherweise nicht treffen, dennoch ist mit dem Fehlen jeder Möglichkeit zum Abspeichern des augenblicklichen Spielstandes eine Voraussetzung gegeben, die Freude an einem Spiel rasch zu vergällen.
Unglücklicherweise gilt das umso mehr für Adventures, bei deren Entwicklung wir einen besonderen Aufwand getrieben haben, und auf deren Komplexität und intelligent gestellte Fallen wir besonders stolz sein dürfen.

Helfen wir also dem Spieler, der sowieso mehr als nur sieben Leben haben muß, um alle möglichen adventuregemäßen Arten des Ablebens kennenlernen zu können, und schaffen wir die technischen Voraussetzungen dazu, daß er das Spiel, um eine Erfahrung reicher, ab dieser Stelle fortsetzen kann. Wenn er dann nicht rechtzeitig Gebrauch von dieser sinnvollen Erweiterung macht, kann er sich nur über sich selber ärgern.

Außerdem verhindern wir mit diesen Routinen einen Schnitt ins eigene Fleisch, denn während der Testphase eines Adventures wird sich immer wieder die Notwendigkeit kleiner

Programmänderungen zeigen, welche leider auch die Inhalte aller Variablen zunichtemachen.
Was bleibt, ist nur ein neuer Start mit Run und die Wiederholung aller notwendigen Eingaben, um wieder in den Raum zu gelangen, in dem der Fehler aufgetreten war, und um dann die Feststellung zu machen, daß die neue Version keineswegs eine Verbesserung bedeutet.

SAVE GAME / LOAD GAME

Überlegen wir zunächst, wodurch sich eine bestimmte Szene des Spieles von der Ausgangsstellung unterscheidet.

Sofort fällt uns der Spieler ein, der sich mehr oder weniger zielstrebig umherbewegt hat, ebenso denken wir an die Gegenstände, die er genommen oder abgelegt, zerstört, verbraucht oder gegessen hat, also an Gegenstände, die nicht mehr existieren ebenso wie an Dinge, die neu im Spiel aufgetaucht sind.

Neben diesen sichtbaren, und deshalb vielleicht weniger wichtigen, weil kontrollierbaren Variableninhalten, müssen wir ebenfalls zur Steuerung des Spielverlaufes gemachte Veränderungen retten, sonst kann es geschehen, daß eine Aktion trotz aller erforderlichen Gegenstände nicht den gewünschten Erfolg bringt.

Denken wir nur an die Flags, die nicht immer nur für den Verlauf einer zusammenhängenden Aktion, sondern auch für die Kontrolle scheinbar zusammenhangloser Ereignisse benutzt werden können.
Gleiches gilt für unsere Richtungstabelle, auch hier sind, wie wir später noch sehen werden, umfangreiche Veränderungen durchaus üblich.

Kurz gesagt: die Inhalte der Variablen SPIELER, OB(), FL() und DURCHGANG() müssen vor einem Spielabbruch als sequentielle Datei auf der Diskette bzw. Kassette gespeichert werden.

Exkurs : Externe Datenspeicherung

Neben der internen Datenspeicherung, bei der die Daten im eigentlichen Arbeitsspeicher der CPU abgelegt werden und ihr somit ständig zur Verfügung stehen, sind für den Umgang mit Daten, gleich welcher Art, auch externe Speicher notwendig, die den Speicher eines Computers fast beliebig erweitern und im wesentlichen der Archivierung und der Bereitstellung größerer Datenmengen dienen. Als Vertreter dieser Gattung sind beispielsweise Lochkarten und -streifen, Magnetbänder, Platten- und Walzenlaufwerke wie auch unsere 1Ø5Ø-Floppy oder der Recorder Atari 1Ø1Ø zu nennen.

Abgesehen von verschiedenen Kapazitäten und Zugriffszeiten, lassen sich auch Unterschiede in der Art der Datenspeicherung erkennen.

Speichergeräte, wie ein Lochstreifenstanzer/-Leser oder wie der Datenrecorder, arbeiten dabei mit sogenannten sequentiellen Dateien und sind am ehesten mit einer altertümlichen Papyrusrolle zu vergleichen, während ein Disketten- oder Plattenlaufwerk meist mit relativen Dateien arbeitet, was einem Karteikasten nahekommt.

Beide Verfahren haben ihre Vor- und Nachteile, die besonders im jeweiligen Platzbedarf und in der Zeit, die benötigt wird, um einen bestimmten Datensatz zu finden, zur Geltung kommen.
So wird jede Anschrift einer Kundendatei in einem Karteikasten eine eigene Karte beanspruchen, während auf

der Rolle zwecks Papierersparnis alle Adressen möglichst dicht notiert wurden. Wird aber die Adresse des Kunden 234 gesucht, nimmt man sich die 234. Karte, denn selbstverständlich sind alle nummeriert, bei der Rolle wird es sich hingegen nicht vermeiden lassen, alle Anschriften zu lesen oder sie zumindest von vorne an durchzuzählen.

Die Leser, die sich nun mehr mit Dateien befassen möchten, mögen dies mit entsprechender Literatur tun, ansonsten mag die Bemerkung genügen, daß die sequentielle Speicherung für unser Problem genau das richtige Verfahren ist, denn wir wollen unsere Variableninhalte der Reihe nach speichern und später wieder laden, das heißt, besondere Zugriffsmethoden sind für uns nicht erforderlich.

SAVE GAME

Die Übertragung der Daten wird zweckmäßigerweise innerhalb entsprechender Schleifen geschehen, wozu es jedoch nötig wird, die Kenndaten des Programmes um die Anzahl der verwendeten Flags zu erweitern, denn auch dieser Schleife muß ein Endwert zugrunde gelegt werden können.

Ebenso müssen wir eine Änderung der Programmzeile 15ØØ vornehmen, um die neuen Routinen in die Abfrage der Sonderbefehle einzubauen:

```
11Ø AR=6:AO=18:AF=3

15ØØ IF EINGABE$(1,3)<>"INV" THEN GOTO
 16ØØ
```

Handelte es sich bei der letzten Eingabe des Spielers nicht um den Befehl Inventur, werden einige Sprünge ausgeführt, bis die Identität mit einem der Kurzbefehle festgestellt worden ist:

```
1599 REM *****t************* SAVE GAME
16ØØ IF EINGABE$(1,3)<>"SAV" THEN GOTO
 17ØØ
1699 REM ****************** LOAD GAME
17ØØ IF EINGABE$(1,3)<>"LOA" THEN GOTO
 19ØØ
```

Bevor das Betriebssystem unseres Computers dann die Daten übertragen kann, müssen ihm zunächst einige Informationen, wie Übertragungsrichtung, Transportweg und Adressat genannt werden, eine Aufgabe, die der OPEN-Befehl übernimmt.

```
162Ø OPEN #1,8,Ø, "D:GAME"
```

Nach Ausführung dieser Zeile steht auf der Diskette (D:) ein Datenfile ©GAME© zur Verfügung, welches sequentiell beschrieben (8) werden kann.

Falls Sie einen Kassettenrekorder benutzen, so ersetzen Sie ©D:© bitte durch ©C:©.

Der Übertragung von Spieldaten steht nach dieser Vorbereitung nichts mehr im Wege. Die Ausgabe der Daten an die Diskette erfolgt dabei in ähnlicher Form wie eine Ausgabe auf den Bildschirm, der Print-Befehl muß nur durch die Angabe einer Filenummer (#1) umdirigiert werden.
Ein abschließender CLOSE-Befehl übernimmt es, uns vor der Rückkehr ins Hauptprogramm vor später auftretenden Fehlern zu schützen.

```
93Ø FOR I=1 TO AF:FL(1)=Ø:NEXT I
1625 PRINT #1,SPIELER
163Ø FOR I=1 TO AO
1631 PRINT #1,OB(I)
1632 NEXT I
```

```
1635 FOR RAUM=1 TO AR
1636 FOR RICHTUNG=1 TO 6
1637 PRINT #1,DURCHGANG(RAUM,RICHTUNG)
1638 NEXT RICHTUNG
1639 NEXT RAUM
1645 FOR I=1 TO AF
1646 PRINT #1,FL(I)
1647 NEXT I
165Ø CLOSE #1
167Ø GOTO 1Ø8Ø
```

LOAD GAME

Zur Rekonstruktion einer bestimmten Situation ist es erforderlich, diese Daten wieder an die entsprechenden Variablen zu überweisen, eine Aufgabe, die ein fast identischer Programmteil übernimmt:

```
172Ø OPEN 1,4,Ø,"D:GAME"
1725 INPUT #1,SPIELER
173Ø FOR I=1 TO AO
1731 INPUT #1,LAGEORT:OB(I)=LAGEORT
1732 NEXT I
1735 FOR RAUM=1 TO AR
1736 FOR RICHTUNG=1 TO 6
1737 INPUT #1,ZIEL:DURCHGANG(RAUM,RICH
TUNG)=ZIEL
1738 NEXT RICHTUNG
1739 NEXT RAUM
1745 FOR I=1 TO AF
1746 INPUT #1,FLAG:FL(I)=FLAG
1747 NEXT I
175Ø CLOSE #1
177Ø GOTO 1Ø8Ø
```

Natürlich müssen Sie als Rekorderbenutzer aus ©D:© ebenfalls wieder ein ©C:© machen.

Um unser Programm noch luxuriöser auszustatten, werden wir noch einige weitere Zeilen ergänzen. So ist es durchaus nicht notwendig, daß wir uns auf ein einziges File zur Abspeicherung beschränken, eine zweite Datei wird sogar unbedingt erforderlich werden, wenn weitere Mitglieder unserer Familie ihr Herz für Adventures entdecken und sich an dem gleichen Spiel versuchen wollen.

Lassen wir den Spielern daher die freie Wahl und erlauben wir ihnen, einen Namen nach Wunsch, bis zu einer maximalen Länge von acht Buchstaben, zu wählen. Das Adventure soll dieses File dann mit der Ergänzung DAT als Spielstandsdatei kennzeichnen.

Als zweckmäßig erweist es sich weiterhin, den Abbruch des Programmlaufes im Falle eines auftretenden Fehlers zu verhindern, was sehr leicht geschehen kann, wenn die falsche oder gar keine Diskette in das Laufwerk eingelegt wurde.
Solchermaßen komplettierte Unterprogramme werden dann ähnlich den folgenden Listings aussehen.

```
1599 REM ****************** SAVE GAME
16ØØ IF EINGABE$(1,3)<>"SAV" THEN GOTO
 17ØØ
16Ø5 PRINT "UNTER WELCHEM NAMEN ";:INP
UT EINGABE$:IF LEN(EINGABE$)>8 THEN PR
INT "BITTE ETWAS KUERZER !":GOTO 16Ø5
161Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
=".DAT"
1615 T$(LEN(T$)+1)=EINGABE$
162Ø OPEN #1,8,Ø,T$
1625 PRINT #1,SPIELER
163Ø FOR I=1 TO AO
1631 PRINT #1,OB(I)
```

```
1632 NEXT I
1635 FOR RAUM=1 TO AR
1636 FOR RICHTUNG=1 TO 6
1637 PRINT #1,DURCHGANG(RAUM,RICHTUNG)
1638 NEXT RICHTUNG
1639 NEXT RAUM
1645 FOR I=1 TO AF
1646 PRINT #1,FL(I)
1647 NEXT I
165Ø CLOSE #1
167Ø PRINT "O.K.": GOTO 1Ø8Ø
1699 REM ****************** LOAD GAME
17ØØ IF EINGABE$(1,3)<>"LOA" THEN GOTO
 18ØØ
17Ø5 PRINT "WELCHES SPIEL ";:INPUT EIN
GABE$:IF LEN(EINGABE$)>8 THEN PRINT "D
AS KANN ES NICHT GEBEN !":GOTO 17Ø5
171Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
=".DAT"
1715 T$(LEN(T$)+1)=EINGABE$
172Ø OPEN #1,4,Ø,T$
1725 INPUT #1,SPIELER
173Ø FOR I=1 TO AO
1731 INPUT #1,LAGEORT:OB(I)=LAGEORT
1732 NEXT I
1735 FOR RAUM=1 TO AR
1736 FOR RICHTUNG=1 TO 6
1737 INPUT #1,ZIEL:DURCHGANG(RAUM,RICH
TUNG)=ZIEL
1738 NEXT RICHTUNG
1739 NEXT RAUM
1745 FOR I=1 TO AF
1746 INPUT #1,FLAG:FL(I)=FLAG
1747 NEXT I
175Ø CLOSE #1
177Ø PRINT "O.K.":GOTO 1Ø8Ø
18ØØ IF EINGABE$(1,3)<>"HEL" THEN GOTO
 19ØØ
```

DER WORTSCHATZ

Häufig macht die richtige Wortwahl ein Adventurespiel zur Qual, denn schließlich wird man als Spieler gezwungen, sich der gleichen Ausdrucksweise wie der Programmierer zu bedienen, was nicht immer ganz einfach ist.
Importe, die für den fortgeschritteneren amerikanischen Markt produziert wurden, lassen sich mit dem Schulenglisch häufig gar nicht, fast immer aber nur schwer lösen. Ein deutsch-englisches Wörterbuch hat bei einigen Spielern daher seinen festen Platz in Reichweite des Computers, doch auch dieses Hilfsmittel versagt, wenn sich der Autor der Umgangssprache oder eines Slangs bedient hat.

Scheitern viele Spiele an der Sprachbarriere, freut man sich um so mehr, wenn aus irgendeiner Quelle plötzlich ein deutsches Adventure auftaucht.
Doch sofort trüben zwei Eigenarten den rosaroten Blick, denn unsere Muttersprache ist bei weitem nicht so gut dazu geeignet, ähnlich wie bei den englischsprachigen Originalen, mit nur zwei Worten die unterschiedlichsten Situationen zu artikulieren. Schlimmes Gastarbeiterdeutsch mag noch akzeptabel sein, und auch über die unmöglichsten Satzkonstruktionen kann man zumindest zu Beginn noch lachen, doch wie soll ein normal begabter und veranlagter Mensch ohne Hilfestellung auf diese Eingaben kommen ?

©Nimm Messer© ist dabei gar nicht einmal so übel, doch das Messer wird auch eine Aufgabe zu erfüllen haben. Es bleibt jedoch weiterhin ein erfreulicher Gegenstand, denn Eingaben wie ©Wirf Messer©, ©Schleife Messer' oder ©Schärfe Messer© erfordern nicht allzu viel Phantasie und sind ebenfalls recht eindeutig auszulegen.

Hat das Messer seine Funktion erfüllt, muß man sich seiner früher oder später entledigen, was bei einigen Spielen durch bewusstes Verlieren geschehen kann. Etwas besser als ©Verliere Messer© wirkt schon ©Leg Messer©, doch taucht

automatisch die Frage nach dem wohin auf oder es wird zumindest noch ein ©weg© erwartet. Diese Eingabe steht zwar im Widerspruch zu den üblichen Spielregeln, sollte aber dennoch, zumindest auf unserem System, versucht werden.

Betrachten wir abschließend noch einmal ein Beispiel der ersten Seiten dieses Buches. Wie gelangen wir in die Krone des Baumes, vielleicht mit ©Kletter Baum© ?

Leider nicht, aber wohin bringt uns ein besser klingendes· ©Erklettere Baum© ?

Wieder nur die Mitteilung, daß das Wort ©Erklettere© nicht verstanden wurde, also aufgeben oder weitergrübeln ?
Es wird sich nicht vermeiden lassen, Verben wie ersteige, besteige und erklimme zu erproben, und wenn die Aktion wirklich erforderlich ist, wird der Erfolg sich früher oder später einstellen.

Probleme dieser Art sind natürlich nicht sehr schön, und da dieses Kapitel von der Perfektionierung unseres Adventuresystemes handelt, werden wir uns nach geeigneten Lösungsmöglichkeiten umsehen.

SYNONYME

Gleichbedeutende Worte in reichlicher Menge einzuarbeiten, wäre ein erster Vorschlag zur Lösung des Sprachproblems. Wie jedoch das Beispiel des Baumes zeigt, kann die konsequente Ausstattung mit gleichbedeutenden Worten den Umfang eines Programmes sehr vergrößern, was ein immenses Maß zusätzlicher Routinearbeit bedeutet und den sowieso immer knapper werdenden freien Speicherplatz stark einschränkt.

Dennoch soll hier, wieder am Beispiel unserer Miniversion von Goldrausch, ein Lösungsvorschlag eingebracht werden, der durch seine Kürze und Einfachheit doch schon fast wieder überzeugt. Selbstverständlich handelt es sich um keine zwingend erforderlichen Programmzeilen, es liegt an Ihnen, ob Sie Wert auf einen großen Wortschatz legen oder ob Sie sich mit dem anschließend gemachten Vorschlag zur Verbesserung der Benutzerfreundlichkeit zufriedengeben wollen.

Die erste Gelegenheit, nicht sofort das richtige Wort zu treffen, hat ein Spieler von Goldrausch in Raum 3 bei dem Versuch, die Holzhütte näher in Augenschein zu nehmen.
Bislang wird er mit ©Untersuche Holzhütte© scheitern, der Erfolg wird sich erst bei ©Untersuche Hütte© einstellen.

Damit der zweite Begriff überhaupt verstanden werden kann, müssen wir unser Vokabular zunächst erweitern:

```
11Ø AR=6:AO=19
18Ø OBJEKTE$="BAEU... ...HOLZ"
```

Die ausführliche Beschreibung können wir uns sparen, desgleichen wird die Null als Aufbewahrungsort verbindlich, schließlich soll weder im Inventory noch in einem Raum eine weitere Hütte auftauchen.
Benötigt wird nur der Rufname, damit der Adventuretreiber die entsprechende Objektnummer ermitteln kann, was in Zeile 216Ø auch geschieht.

Normalerweise wird die Ausführung des Programmes in Zeile 22ØØ mit einem Sprung zur jeweiligen Befehlsausführung fortgesetzt. Dort findet sich natürlich keine einzige Bedingung, die erfüllt wäre, da nirgendwo für N ein Wert von 19 gefordert wird. Um diese Zeilen nicht ergänzen zu müssen, fügen wir folgenden Kunstgriff ein,

```
216Ø IF EO$=RN$(N) THEN 2185
```

```
2184 REM ******************** SYNONYME
2185 IF N=19 THEN N=4
```

und schon werden mit der Holzhuette genau die gleichen Aktionen durchgeführt wie mit der Huette.

Um unser Programm in die Lage zu versetzen, auch eine Vielzahl von Verben verstehen zu können, ist der zu treibende Aufwand sogar noch geringer.
Ein die gleiche Aktion einleitendes Wort wird ebenfalls dem Wortschatz des Adventures zur Verfügung gestellt, und die Sprungtabelle wird einfach um das bereits einmal benutzte Sprungziel ergänzt.

Goldrausch erweitern wir auf diese Weise um die zu ©Untersuche© identische Funktion ©Betrachte©:

```
17Ø VERBEN$="UNTE... ...BETR"
22ØØ ON VN GOTO 5ØØØ,6ØØØ,7ØØØ,8ØØØ,9Ø
ØØ,1ØØØØ,5ØØØ
```

Sie sehen, daß unser System sich um gleichbedeutende Verben besonders einfach erweitern lässt, weshalb wir zumindest auf diesen Bonus nicht verzichten sollten.

Synonyme der Objekte dürfen wir stiefmütterlich behandeln, da ihnen sowieso nicht die gleiche Bedeutung zukommt, denn die Kopfzeilen des Spieles werden fast immer, mit Ausnahme bei den Grafikadventures, Anhaltspunkte geben.

Wir können es dem Spieler aber auch noch einfacher machen, so daß er sich wirklich auf die Problemlösungen konzentrieren kann und nicht auf der Suche nach den richtigen Worten den roten Faden verliert.

Was würden Sie davon halten, wenn ein Adventure auf Wunsch sein gesamtes Vokabular preisgibt, wenn es eine Liste aller vorhandenen Objekte und möglichen Verben erstellt ?

VOKABULAR

Eine Auflistung des gesamten Vokabulars dürfte alle Probleme der Wortwahl aus der Welt schaffen, und der Spieler verliert nicht durch sinnlose Eingaben Zeit und Lust.

Leider lässt sich an dieser Stelle zum zweiten Mal der Volksmund zitieren, denn ©Wo Licht ist, ist auch Schatten©.

Stellen wir uns einen Spieler vor, dem Goldrausch noch nicht bekannt ist. Ohne viel gespielt zu haben, wird er, wenn er einigermaßen konsequent und logisch denken kann, sogleich eine Lösungsstrategie entwickeln, die ihn schnell ans Ziel bringt. Er kennt von Beginn an den gesamten Wortschatz des Adventures und weiß, daß sich irgendwo eine Flasche finden lässt, daß irgendwo eine Truhe liegt. Natürlich wird er ebenfalls sogleich ein starkes Verlangen nach dem Silber und den Nuggets entwickeln. Anschließend überlegt er, welche Gegenstände sich zerstören lassen, und seine Wahl wird auf die Flasche, die Hütte und die Kette fallen.
Er bemüht sich, die Schatztruhe zu finden, und nachdem er die Feststellung machte, daß eine Eisenkette ihn am Sichten des Inhaltes behindert, versucht er diese zu zerstören, was aber mit bloßen Händen kaum gelingen wird. Ein weiteres VOK wird ihm dann die Ahnung vermitteln, daß die Eisenstange vermutlich der einzige Gegenstand ist, der für diese Aufgabe in Frage kommt.

Aber auch die Gefahren werden entschärft, denn es bleibt ihm ebenfalls nicht verborgen, daß außer ihm noch ein Bär die Welt bevölkert !

Somit wird dem Programmierer auch noch die Aufgabe zufallen, genau abzuwägen, ob er den gesamten Wortschatz des Adventures preisgeben will oder nur eine Untermenge.

Dies könnte problemlos durch ein weiteres Titelbild geschehen, welches eine Anzahl der wichtigsten Worte ausgibt, allerdings würden wir damit unserem Ziel, ein universelles Adventuresystem zu entwickeln, entgegenhandeln, weil diese Tafel für jedes Programm neu erstellt werden müsste.

Für große Adventures mit einem entsprechenden Wortschatz bleibt die Auflistung des Vokabulars jedoch durchaus sinnvoll, weshalb wir bei der Entwicklung unseres Adventuresystemes ebenfalls nicht darauf verzichten wollen.

Die Behandlung des VOK - Befehles übernimmt aus eben genanntem Grunde der Treiber. Um eine wirklich universelle Verwendung dieses Programmteils zu gewährleisten, müssen die zur Ausgabe bestimmten Worte aus den Spieldaten des betreffenden Adventures ermittelt werden.

Zurzeit weist unser System allerdings nur die zur Identifikation eines Wortes notwendigen Kürzel auf, weshalb wir zunächst die betreffenden Strings ändern. Damit die Routine, welche aus diesen Zeichenketten VERBEN$ und OBJEKTE$ die einzelnen Worte ermittelt, funktionsfähig bleibt, muß die Länge der einzelnen Worte gleich sein, die Wortlänge wird sich somit nun nach der Anzahl der Buchstaben des längsten Wortes richten:

```
17Ø VERBEN$="UNTERSUCHENIMM      LEG
     OEFFNE    BENUTZE   ZERSTOERE BET
```

```
    RACHTE "
```

Weiterhin muß die Schrittweite der Schleifen von vier auf zehn geändert werden. Um nun jedoch für die Realisation der verschiedensten Spiele nicht immer wieder Änderungen vornehmen zu müssen, führen wir eine weitere Variable WL, gleichbedeutend mit Wortlänge, ein:

```
    13Ø WL=1Ø

    2Ø9Ø FOR 1=1 TO LEN(VERBEN$) STEP WL
    214Ø FOR 1=1 TO LEN(OBJEKTE$) STEP WL
```

Bauen wir nun das entsprechende Unterprogramm, unter Berücksichtigung der wahrscheinlichen Häufigkeit eines Aufrufs der entsprechenden Zeilen, im Anschluss an die INV, SAVE und LOAD Routine ein:

```
    18ØØ IF EINGABE$(1,3)<>"VOK" THEN GOTO
     19ØØ
    18Ø5 GOSUB 188Ø
```

Zunächst wird in Zeile 18Ø5 ein Unterprogramm aufgerufen, welches den Bildschirm löscht und eine Titelzeile ausgibt. Der Einsatz einer eigenen Routine wird gerechtfertigt durch den mehrmaligen Bedarf dieser Funktion bei längeren Adventures, wenn mehrere Seiten nacheinander ausgegeben werden müssen.

```
    188Ø GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
    R 4,1,Ø:PRINT "ICH VERSTEHE FOLGENDE V
    ERBEN:":PRINT :RETURN
```

Nachdem dann die betreffenden Worte ermittelt und ausgegeben worden sind, wartet Zeile 189Ø auf eine Betätigung der Return-Taste, um danach den Programmlauf fortzusetzen:

```
1815 FOR I=1 TO LEN(VERBEN$) STEP 1Ø
182Ø PRINT VERBEN$(I,I+9)
183Ø NEXT I
187Ø GOSUB 189Ø:GRAPHICS Ø:SETCOLOR 2,
1,Ø:SETCOLOR 4,1,Ø:GOTO 1Ø8Ø
189Ø PRINT :PRINT :PRINT "<RETURN> DRU
ECKEN ... ";: INPUT EINGABE$:RETURN
```

Obige Zeilen werden nun wie vorgesehen arbeiten, zumindest solange nicht mehr als achtzehn Worte ausgegeben werden müssen. Nach Überschreiten dieser Zahl wird die Verweildauer der ersten Vokabeln auf dem Bildschirm kaum von ausreichender Länge sein, um eine Hilfe für den Spieler zu sein.

Zur Lösung Problemes benutzen wir eine Kontrollvariable, deren Inhalt zu Beginn der Ausgabe auf null gesetzt und mit der Ausgabe eines jeden Wortes um eins erhöht wird. Sind nach achtzehn Worten entsprechend viele Zeilen beschrieben worden, löschen wir den Bildschirm und setzen den Zähler wieder auf eins zurück.

```
181Ø GEDRUCKT=Ø
182Ø PRINT VERBEN$(I,I+9):GEDRUCKT=GED
RUCKT+1
1825 IF GEDRUCKT=18 THEN GOSUB 189Ø:GE
DRUCKT=1:GOSUB 188Ø
```

Auf gleiche Art und Weise kann der Spieler auch über die vom Spiel verstandenen Objektbezeichnungen informiert werden:

```
18Ø OBJEKTE$="BAEUME     BAEUME FE
LSEN ...
1835 GOSUB 189Ø
184Ø GOSUB 1885
```

```
1845 GEDRUCKT=Ø
185Ø FOR I=1 TO LEN(OBJEKTE$) STEP 11
1855 PRINT OBJEKTE$(I,I+1Ø):GEDRUCKT=G
EDRUCKT+1
186Ø IF GEDRUCKT=18 THEN GOSUB 189Ø:GE
DRUCKT=1:GOSUB 1885
1865 NEXT I
1885 GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:PRINT "FOLGENDE GEGENSTAENDE S
IND MIR BEKANNT":RETURN
```

HELP

Die Entwicklung einer HELP-Routine soll nun ein letzter Schritt sein, um die Benutzerfreundlichkeit unserer Adventures zu erhöhen.

Glaubt der Spieler, daß er sich restlos verfahren hat, so wird er versuchen, durch die bloße Eingabe von Help in den Besitz hilfreicher Informationen zu gelangen.

Handelt es sich um ein einfaches Spiel, weil im Verlauf desselben genügend Hinweise gegeben werden, machen wir uns die Sache einfach:

```
1559 REM ********************** HELP
156Ø IF EINGABE$(1,2)<>"HEL" THEN GOTO
 16ØØ
157Ø PRINT "ICH KANN DIE SPIELREGELN W
IEDERHOLEN.":PRINT "WUENSCHEN SIE DAS
?";:INPUT EINGABE$
1571 IF EINGABE$(1,1)="J" THEN GOSUB 8
ØØ:GOTO 1Ø8Ø
1975 GOTO 1Ø8Ø
1979 REM ****************** ENDE HELP
```

Eine noch häufiger anzutreffende Standardantwort lautet ©Ich würde alles untersuchen ! - Try examining things !©. Meist werden jedoch, zumindest in den Abenteuerspielen neueren Datums, Ratschläge gegeben, die manchmal allerdings wenig hilfreich, weil für den Spieler nicht zu verstehen, sind.

Denn da es sich um einen Ein-Wort-Befehl handelt, steht die gewünschte Hilfe nicht in Beziehung zu einem bestimmten Objekt, was eine Auswertung der Situation in Hinblick auf die Schwierigkeiten des Spielers nicht gerade einfach macht.

Schließlich steht dem Programm als Arbeitsgrundlage nur der Aufenthaltsraum des Spielers zur Verfügung, und so ist es nicht weiter verwunderlich, daß viele Adventures in problematischen Räumen immer wieder die gleiche Mitteilung an den Abenteurer ausgeben, die er deshalb nicht verstehen kann, weil seine Gedanken bislang um ein anderes Problem kreisen.

Als typisches Beispiel ziehen wir eine Spielszene aus Raum vier heran. Der Spieler hat die Schatztruhe entdeckt und müht sich seit längerer Zeit vergeblich mit der hinderlichen Eisenkette ab. In dieser Situation wäre folgender Hinweis denkbar:

```
157Ø IF SPIELER=4 THEN PRINT "EIN VERL
AENGERTER ARM VERSTAERKT DIE KRAFT.":G
OTO 1Ø8Ø
```

Was könnte ein Spieler, der in Raum vier um Hilfe bittet, mit der Information, an die Hebelgesetze zu denken, anfangen, wenn er bereits bei der Suche nach der Truhe gescheitert ist und er somit die Kette noch nicht entdeckt haben kann ?

Will man den Spieler verwöhnen und in besonders schwierigen Situationen Schritt für Schritt an die Lösung heranführen, wird es, wie obiges Beispiel zeigt, nicht ohne die Formulierung umfangreicher Tests und Bedingungen gehen. Für diese Aufgabe scheinen wieder einmal die Flags wie auch die Aufbewahrungsorte der verschiedensten Gegenstände prädestiniert zu sein.

Wiederum am Beispiel der Truhe probieren wir einmal diese Art der genau dosierten Hilfestellung aus, was allerdings nicht heißen soll, daß diese Szene als besonders schwierig zu gelten hat:

```
6Ø6 M6$="WIE KANN ICH DIE KETTE ZERSTO
ERENÖ ?"
6Ø7 M7$="SIE VERHINDERT EIN OEFFNEN DE
R TRUHE."

197Ø IF SPIELER=4 AND OB(1Ø)=Ø THEN PR
INT "FAST WAERE ICH IN EIN LOCH GEFALL
EN.":GOTO 1Ø8Ø

1971 IF SPIELER=4 AND OB(11)=SPIELER A
ND  NOT FL(2) THEN PRINT M6$;:PRINT M7
$:GOTO 1Ø8Ø
```

Erinnern wir uns, daß beim Passieren des betreffenden Raumes zunächst nur die Büsche sichtbar sind, und das Erdloch zwischen den Büschen erst entdeckt werden muß. Erlangt der Spieler jedoch die Information eines Beinahe-Sturzes in eine Grube, wird er sich wohl näher mit diesem Hindernis beschäftigen.

Ein weiterer Versuch mit Help bringt zu diesem Zeitpunkt keinen besonderen Vorteil. Erst nachdem die Truhe als sichtbares Objekt zur Ausstattung des Raumes gehört, erhält der Spieler den Hinweis, daß die Begutachtung des Inhaltes die Zerstörung der Kette und das Öffnen des Deckels erfordert.

Diese Aktionen wird er dann wohl hoffentlich alleine durchführen können, falls nicht, wäre es für den Spieler eine Überlegung wert, ob er nicht lieber einen Roman lesen oder es mit Arkade-Action-Spielen versuchen sollte.

Den Abschluss einer solchermaßen durchkonstruierten Hilfestellung wird wieder eine Sicherheitszeile, wie wir sie bereits bei der Programmierung der anderen Aktionen kennengelernt haben, bilden, eine Zeile, die immer dann ausgeführt wird, wenn keine speziell definierte Situation eingetreten ist:

```
1971:
1972:
1975 PRINT "ERST SEHEN, DANN DENKEN UN
D ZULETZT HANDELN !": GOTO 1Ø8Ø
```

MOTIVIERUNG DES SPIELERS

Unser nächster Vorschlag für bessere Adventures wird unseren Adventuretreiber um einen letzten Ein-Wort-Befehl erweitern.

Abenteuerprogramme benötigen für ihre Lösung nun einmal einen gewissen Zeitaufwand, wenn man jedoch tagelang spielt und überhaupt keinen Fortschritt sieht, fragt man sich eines Tages doch, ob man nicht ein anderes Spiel laden soll.

Aus diesem Grunde halte ich Adventures, die mit einer Punktwertung arbeiten und dem Spieler anzeigen, wie weit er noch von seinem endgültigen Ziel entfernt ist, für bedeutend reizvoller.

Jedoch gilt es zu überlegen, auf welcher Basis die Wertung durchgeführt werden soll. Die übliche Version sieht für

jeden gefundenen Schatz beziehungsweise für jede gelöste Teilaufgabe in einem Mission Adventure eine gewisse Anzahl von Punkten vor. Neben der Angabe ©Von 1ØØ möglichen Punkten hast du 4Ø© wird dem Spieler auch noch der prozentuale Anteil mitgeteilt, wodurch der Spielstand einfacher und genauer ausgewertet wird. Als Beispiel kann wieder einmal Scott Adams ©Adventureland© angeführt werden, dreizehn Schätze ergeben eine krumme Anzahl von Punkten, bei mehr als 5Ø Prozentpunkten weiß der Spieler jedoch, daß er die Hälfte der Strecke geschafft hat.

Eine alternative Lösung wird dagegen jeden Schritt voran belohnen und auch nicht davor zurückschrecken, für jede geleistete Hilfe eine gewisse Anzahl von Punkten abzuziehen. Ebenso wird jeder Fehler mit Todesfolge negativ zu Buche schlagen, so daß im extremen Fall sogar ein negativer Wert erzielt werden kann. Doch ist es gerade dieser große Spielraum, der zu wiederholten Lösungsversuchen reizt, denn es müsste doch möglich sein, mehr Punkte zu erzielen als der Freund, der das Ziel ebenfalls erreicht hat.

Soll ein Adventure dermaßen ausgestattet werden, beginnt die Arbeit des Programmierers mit der Auswahl geeigneter Sequenzen, schließlich soll nicht bereits das einfache Auffinden eines Gegenstandes Punkte wert sein. Eine Belohnung darf der Spieler jedoch erwarten, wenn er gefährliche Monster ausgetrickst oder geheime Durchgänge gefunden hat.

Bevor wir uns nun detailliert mit der praktischen Ausführung entsprechender Versionen befassen, vermitteln wir als Erstes dem Treiber das Verständnis für den Befehl: ©SCORE©:

```
15ØØ IF EINGABE$(1,3)<>"INV" THEN GOTO
 156Ø
1559 REM ********************** SCORE
156Ø IF EINGABE$(1,3)<>"SCO" THEN GOTO
 16ØØ
```

Anschließend führen wir zur Speicherung des erzielten Punktewertes die Variable WERTUNG ein. Dabei ist es wichtig, dafür Sorge zu tragen, daß sie bei jedem Neustart, somit auch beim Tod der Hauptfigur, auf null gesetzt wird.

Nur der erreichte Stand des Punktekontos sagt natürlich gar nichts aus, wenn er nicht in Beziehung zur maximalen Punktzahl steht, deshalb:

```
14Ø WMAX=2Ø :REM MAXIMALE PKTE DER MIN
IVERSION
143 WERTUNG=Ø :REM Ø PUNKTE
1561 PRINT "VON ";WMAX;" PUNKTEN HAST
 DU ";WERTUNG;" PUNKTE."
1565 GOTO 1Ø8Ø
```

Die Punkte erzielt der Spieler mit dem Aufnehmen der Gold- und Silberstücke. Ergänzen wir darum die betreffenden Zeilen in unserem Aktionsteil:

```
6Ø14 IF N=12 THEN PRINT "O.K.":OB(12)=
-1:WE=WE+1Ø:GOTO 1Ø8Ø
6Ø17 IF N=17 AND FL(3) THEN PRINT "O.K
.":OB(17)=-1:WE=WE+1Ø:GOTO 1Ø8Ø
```

Leider kann ein Spieler, der es nur auf Punkte abgesehen hat, mit der derzeitigen Ausführung des Programmes beliebig hohe Summen erlangen.

Erinnern wir uns, daß wir bei der Ausgestaltung der Aktion ©Nimm© darin übereingekommen waren, daß mit nehmen ein ©in die Hand nehmen© gemeint war. Dies ermöglichte der Spielpraxis neben der Aufnahme von Gegenständen zwecks Aufbewahrung im Inventory auch ein Nehmen aus dem Inventory, um eine Aktion, beispielsweise das Öffnen einer Tür mittels eines Schlüssels, durchzuführen.

Gerade diese Auslegung erweist sich nun als hinderlich, nichts hält den Spieler davon ab, mehrmals ©Nimm Silbermuenzen© einzugeben und mit dieser Eingabe jeweils zehn Punkte auf seinem Konto hinzuzuaddieren.

Als Ausweg bieten sich einmal eine Änderung der allgemeinen Vorbedingung (Zeile 6ØØØ) wie auch die Einführung zusätzlicher Bedingungen in die Aktion selbst an. Dieses Prinzip kennen wir bereits von der Ausführung der Aktion ©Untersuche©, hier mußte eine ähnliche Regelung verhindern, daß ein Gegenstand immer wieder an seinem Ursprungsort sichtbar wurde.

```
6Ø14 IF N=12 AND OB(N)=4 THEN PRINT "O
.K." :OB(12)=-1:WE=WE+1Ø:GOTO 1Ø8Ø
6Ø17 IF N=17 AND OB(N)=5 AND FL(3) THE
N PRINT "O.K.":OB(17)=-1:WE=WE+1Ø:GOTO
 1Ø8Ø
6Ø2Ø IF N=12 AND OB(N)=-1 THEN PRINT "
DAS SILBER HABE ICH BEREITS !":GOTO 1Ø
8Ø
6Ø21 IF N=17 AND OB(N)=-1 THEN PRINT "
ICH BIN BEREITS IM BESITZ DES GOLDES !
":GOTO 1Ø8Ø
```

Entscheiden Sie sich hingegen, die Aktion ©Nehmen© immer nur als Bereicherung des Spielers aufzufassen, können Sie sich auf eine Verschärfung der Eingangsbedingung und Ausgabe eines entsprechenden Hinweises beschränken:

```
6ØØØ IF N<>SP THEN GOTO 69ØØ
6998 PRINT "DAS HABE ICH DOCH SCHON !"
:GOTO 1Ø8Ø
```

Für unsere eigenen, das heißt zumindest für die Adventures dieses Buches, wollen wir nur eine Punktierung der in den Besitz des Spielers gebrachten Schätze vornehmen. Dennoch werden wir nicht auf eine Bewertung der einzelnen Züge des Spielers verzichten müssen.

Im Gegenteil, wir werden sogar noch einen Schritt weiter gehen und jeden einzelnen Spieler bewerten, ihm die erreichte Durchschnittswertung mitteilen.

Als Grundlage für diese Auswertung bietet sich die Anzahl der benötigten Züge an. Ein erfahrener Abenteurer wird sich nicht mit der Untersuchung jeder Kleinigkeit aufhalten, ebenso weiß er, aus typischen Hinweisen seinen Nutzen zu ziehen.
Aus dieser Summe und den dabei erreichten Punkten berechnen wir einen Durchschnittswert und nehmen anhand dieses Quotienten die Beurteilung des Spielers vor.

```
146 ZUG=Ø:WERTUNG=Ø

1Ø8Ø GRAPHICS Ø:ZUG=ZUG+1

1561 PRINT "VON ";WMAX;" PUNKTEN HAST
DU IN ";ZUG;" ZUEGEN":PRINT WERTUNG;"
PUNKTE ERREICHT !"
1562 PRINT "DAS ENTSPRICHT EINEM SCHNI
TT VON ":PRINT WERTUNG/ZUG;" PUNKTEN."
1565 GOTO 1Ø8Ø
```

Auch das Kriterium für ein erfolgreiches Spielende definieren wir neu. Anstelle einer genau definierten Bedingung, die spielabhängig ist, vergleichen wir nun die erreichte Punktzahl mit dem maximal möglichen Wert. Ist die Differenz gleich null, hat der Spieler die Aufgabe gelöst.

Zweckmäßigerweise erfolgt dieser Test gleich zu Beginn eines jeden Zuges, bevor überhaupt irgendwelche Daten an den Spieler übermittelt werden.

```
1Ø85 IF WERTUNG=WMAX THEN GOTO 48ØØ
```

Falls Sie ebenfalls die Punktzahl als Kriterium verwenden, vergessen Sie bitte nicht, alle anderen Zeilen, die mit der gleichen Aufgabe beschäftigt sind, aus dem Programm zu entfernen (Zeilennummern im Bereich von 1331 bis 1389).

Mit der konsequenten Einführung eines Punktesystems stellt sich uns zum Abschluss noch die Aufgabe, den Siegestitel an diesen Aufbau anzupassen. Die ermittelten Daten sollten noch einmal zusammengefasst werden, damit auch eine Benotung des Spielers stattfinden kann.

Dazu wird es sich allerdings nicht vermeiden lassen, daß Sie, wenn das Adventure fertiggestellt ist und keinerlei Fehler mehr aufweist, einen kompletten Lauf durchspielen. Bei diesem Spiel verzichten Sie auf alle nicht unbedingt ·erforderlichen Eingaben und ermitteln die Anzahl der für einen Sieg mindestens erforderlichen Züge. Berechnen Sie das Verhältnis zwischen Punkte und notwendigen Eingaben und legen Sie diesen oder einen etwas geringeren Wert als Kennzeichen für ein sehr gutes Spiel zugrunde.
Wählen Sie entsprechend abgestufte Zahlen für schlechtere Beurteilungen.

So kann die Benotung eines Spielverlaufes der Miniversion von Goldrausch etwa folgendermaßen vorgenommen werden:

```
48ØØ GRAPHICS Ø
48Ø5 ENDWERTUNG=WERTUNG/ZUG
481Ø PRINT "HERZLICHEN GLUECKWUNSCH !"
482Ø PRINT :PRINT :PRINT "SIE HABEN DI
E IHNEN GESTELLTE AUF-":PRINT "GABE GE
LOEST.":PRINT
483Ø PRINT "IN ";ZUG;" SPIELZUEGEN HAB
EN SIE PRO ":PRINT "ZUG ";ENDWERTUNG;"
 PUNKTE GEMACHT.":PRINT
484Ø PRINT "DAMIT HABEN SIE EIN ";
485Ø IF ENDWERTUNG<Ø.5 THEN MØ$="MISER
ABLES"
486Ø IF ENDWERTUNG>Ø.5 THEN MØ$="MAESS
IGES"
487Ø IF ENDWERTUNG>1.5 THEN MØ$=" SEHR
 GUTES"
488Ø IF ENDWERTUNG>1.Ø THEN MØ$="GUTES
"
489Ø PRINT MØ$;" ERGEBNIS ERZIELT."
4899 PRINT :PRINT :END
```

Ein solchermaßen gestalteter Titel, vielleicht zusätzlich durch die Verwendung von Grafikzeichen aufgemacht, wird manchen Abenteurer, der das Adventure endlich gemeistert hat, dazu veranlassen, es noch einmal zu spielen.

Und daß alles nur, um als guter Spieler bezeichnet zu werden.

Mit diesen Bemerkungen wäre das Thema ©Score© eigentlich beendet, wenn wir nicht noch eine technische Ergänzung machen müßten, deren Fehlen ansonsten die ganze schöne Planung zunichtemachen kann und mit Sicherheit die Spieler unserer Programme verärgern würde.

Vielleicht haben Sie bereits an unsere Save-Game-Routine gedacht, der natürlich auch ein Abspeichern und Laden der oben eingeführten Daten Wertung und Zug ermöglicht werden muß, wenn der Spieler nicht immer wieder mit null Punkten beginnen soll.

```
1627 PRINT #1,WERTUNG
1628 PRINT #1,ZUG

1727 INPUT #1,WERTUNG
1728 INPUT #1,ZUG
```

Alle bisher in diesem Kapitel gemachten Vorschläge dienten einzig und allein dem Ziel, den Komfort der Programme zu erhöhen.

Ich glaube, daß Sie mit mir einer Meinung sind, wenn ich die Behauptung aufstelle, daß Adventures, die nach dem Konzept dieses Buches entwickelt wurden und sämtliche Extras enthalten, durchaus zur Oberklasse der Textadventures gezählt werden dürfen. Schließen wir daher die technische Entwicklung mit diesen Zeilen ab, und wenden wir uns der Frage zu, wie wir die Spiele selbst attraktiver und auch schwieriger machen können.

Dabei werden wir es halten wie gewohnt, nach trockener Theorie folgen entsprechende Programmzeilen, die wiederum für unser Erstlingswerk Goldrausch gedacht sein werden.

Die folgenden Seiten werden Ihnen insbesondere zeigen, wie Sie dem Spieler weitere Steine in den Weg legen können, so daß er tatsächlich die angebotenen Kurzbefehle Save und Load nutzen muß, um irgendwann einmal ans Ziel zu gelangen.

Um ausreichend Platz für Monster, Falltüren und was es sonst noch gibt, zu haben, werden wir uns als Nächstes

wieder einmal mit der Geografie befassen und unsere Miniwelt zur Grundlage eines ausgewachsenen Adventures entwickeln.
Falls Sie allerdings beabsichtigen, selber in den Genuss eines Adventurespieles zu kommen, sollten Sie, bevor Sie sich mit den nächsten Abschnitten befassen, alle noch fehlenden Programmzeilen eingeben oder besser noch, von jemand anderem eingeben lassen.

Ein entsprechendes Listing, welches alle gemachten Vorschläge berücksichtigt, finden Sie im nächsten Kapitel.

ORIENTIERUNG ODER DESORIENTIERUNG

Es wurde bereits zu Beginn des Buches erwähnt, daß die meisten Abenteuerspiele sich innerhalb einer Welt von ungefähr vierzig Räumen abspielen. Dennoch entsteht der Eindruck einer viel größeren Welt, gab es da doch den riesigen Wald, einen Sumpf, eine unterirdische Höhlenlandschaft, ein Labyrinth und, und, und.

Tatsächlich sind jedoch gerade die so weiträumig erscheinenden Gebiete Gruppen von einigen wenigen Räumen, die auf so geschickte Art und Weise miteinander verbunden wurden, daß der Spieler meistens nicht einmal bemerkt, auf welche Tour er da geschickt wurde.

Die Palette der Möglichkeiten reicht dabei vom einfachen Raum, der immer wieder in sich selbst mündet, bis zu einer Kette von Räumen, an deren Ende der Spieler wieder an den Beginn versetzt wird.

Nutzen wir an dieser Stelle die in Goldrausch ähnlichen Räume 1 und 2, um gleich zu Spielbeginn ein weitläufiges Areal zur Verfügung zu haben.

Lassen Sie Raum 1 in nördlicher und westlicher Richtung wieder in sich selbst zurück münden, verfahren Sie gleichermaßen mit Raum 2, und versuchen Sie dann, sich den Spieler vorzustellen, der in Raum 1 beginnt und nach Westen wandert. Woher soll er wissen, daß er sich nach jeder Eingabe immer noch im gleichen Teil des Waldes befindet ?
Falls er nicht das Glück hat, das Spiel mit der Eingabe ©O© zu beginnen, wird er eine Reihe von Eingaben benötigen, um den Weg aus dem Wald zu finden, was ihm einen guten Schnitt bei der Punktwertung bereits verdirbt.

```
5Ø1 DATA 1,1,1,2,Ø,Ø
5Ø2 DATA 2,2,3,1,Ø,Ø
```

Noch schwieriger wird sich die Suche nach dem rechten Weg gestalten, wenn der südliche Ausgang von Raum 2 den Spieler nicht wieder in Raum 2 gehen, sondern ihn Raum 1 betreten lässt.

```
5Ø1 DATA 1,1,1,2,Ø,Ø
5Ø2 DATA 2,1,1,3,Ø,Ø
```

Starten Sie das Programm, und lassen Sie sich von der Wirkung dieser kleinen Änderung überraschen.

Mit drei Räumen lassen sich auf diese Art und Weise bereits kleine Labyrinthe erstellen, wenn die Verbindungen wahllos, ohne Bezug zu einem Kompass, ausgeführt werden, und nur ein einziger Zu- wie auch Abgang existieren.

Die übliche, bislang auch von uns verwandte Verbindungstechnik hält sich an die realen, durch Himmelsrichtungen vorgegebenen Wege. Dabei entsteht eine Welt, die eine problemlose Identifizierung der Räume und, besonders wenn der Spieler eine entsprechende Karte anfertigt, Orientierung gestattet.

Kleine Sprünge über einige Räume hinweg machen dieses Konzept jedoch schnell unbrauchbar.
Stellen wir uns drei hintereinanderliegende Räume vor, die alle mit der Beschreibung ©Ich bin in einer großen Höhle© aufwarten.
Ein Spieler, der von Westen in Höhle 1 gelangt, wird, da er nichts besonderes sieht, zunächst mit großen Schritten die nähere Umgebung kennenlernen wollen und vermutlich seine einmal eingeschlagene Richtung nicht ändern, da es ihm ein Bedürfnis ist, möglichst einfach den Weg zurückzufinden.

Nach Durchqueren von Raum 2 und 3 gelangt er wiederum in den ersten Raum, was er ohne entsprechende Handlungen aber unmöglich feststellen kann.

Durch Verwendung dieser Sprungtechnik ist es möglich, ohne großen Aufwand riesig erscheinende Labyrinthe zu erstellen. Sechs Räume sind völlig ausreichend, dem Spieler das Finden des richtigen Weges fast unmöglich zu machen. Selbst der Programmierer wird während der Testphase auf ausreichende Schwierigkeiten treffen !

Denn selbst die Anfertigung einer Karte wird bei dieser Arbeitsweise zum Problem, nicht jedoch bei einem alternativen Verfahren, das an dieser Stelle auch kurz vorgestellt werden soll.

Eine Vorgehensweise nach dieser Methode sieht zunächst ein maßstabsgerecht auf vorzugsweise kariertem Papier erstelltes Labyrinth vor, wie Sie es häufig auch in Rätselecken der verschiedensten Zeitschriften finden.
Bei der anschließenden Programmierung entspricht dabei jedes Kästchen einem Raum, wodurch der beträchtliche Aufwand bereits abzusehen ist.
Ebenso leicht ist es einzusehen, daß der Spieler während seiner Exkursion mit jedem neuen Raum ein neues Kästchen nimmt, die Wände markiert und die Durchgänge blank lässt.

Nehmen Sie stattdessen sechs Räume, von denen Sie jeweils einen als Eingangs-, als Sammel- und als Ausgangsraum festlegen. Dabei ist es für den Eingangsraum wichtig, daß er allein und auch nur in einer einzigen Richtung wieder zurück an den zuletzt besuchten Ort führt.
Entsprechendes gilt für den letzten Raum, nur ein einziger Durchgang erlaubt das Betreten von Neuland, alle anderen Wege müssen ins Labyrinth zurückführen.
Dem Sammelraum kommt dabei eine zentrale Funktion zu. Von jedem Raum des Labyrinthes führen zwei oder mehr Wege in diesen Raum, alle Ausgänge bringen den Spieler jedoch wieder in den ersten oder auch zweiten Raum des Irrgartens.

Um die Sache noch zu komplizieren, werden die Ausgänge des Raumes, der verbindungsmäßig vor dem letzten Raum liegt, noch in der Art angelegt, daß nur ein einziger Weg in den letzten, alle anderen aber in den Sammel- oder in den ersten Raum führen.
Es erübrigt sich zu sagen, daß für alle Räume eine identische Beschreibung vorgesehen wird und daß möglichst alle sechs Richtungen benutzt werden.

Der Spieler wird nun fast soviel Glück wie ein Lottosieger benötigen, um das Labyrinth verlassen zu können, ehe er im vorletzten Raum aus sechs Wegen den einzig richtigen findet, gleiches gilt für den letzten Raum. Dabei wirft ihn jeder Fehler wieder in einen anderen Raum zurück, welcher wiederum sechs Möglichkeiten bietet.

Versuchen Sie, sich im folgenden Labyrinth zurechtzufinden - obwohl Sie selbst die Verbindungen erst kurze Zeit zuvor eingegeben haben, wird es Ihnen schwerfallen:

Das Labyrinth von Goldrausch:

```
          11Ø AR=14: ...
```

```
2Ø7 PRINT "AM EINGANGSSTOLLEN EINES AL
TEN BERG-":PRINT "WERKES.":RETURN
219 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
224 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
225 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
226 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
227 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
228 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
229 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN


5Ø7 DATA 7,Ø,1,5,Ø,Ø
519 DATA 8,Ø,Ø,5,Ø,Ø
524 DATA 1Ø,7,8,8,8,8
525 DATA 11,8,8,13,11,8
526 DATA 11,8,1Ø,11,11,8
527 DATA 9,8,1Ø,11,1Ø,8
528 DATA 8,1Ø,1Ø,11,Ø,Ø
529 DATA Ø,11,12,11,Ø,Ø
```

ORTSWECHSEL

Außer Irrgärten in beliebiger Größe stehen uns weitere Mittel zur Verfügung, um dem Spieler die Orientierung zu erschweren.

Sehr großer Beliebtheit erfreuen sich dabei plötzliche Transfers der Hauptperson in einen anderen Raum, was dem Abenteurer durch Mitteilungen wie ©Alles um mich herum dreht sich© deutlich gemacht wird.
Doch ist auch diese Nachricht bestenfalls als freundliche Geste des Programmierers aufzufassen, und nicht als Pflicht zu verstehen.

Auslöser der Aktionen solcher Art sind meist magische Gegenstände, zu deren Bedienung seit Aladins Wunderlampe ein kräftiges Reiben erforderlich ist.
Aber auch Zaubersprüche und -tränke verfehlen ihre Wirkung nicht, weshalb dem Spieler eines Adventureprogrammes nur geraten werden kann, allergrößte Vorsicht walten zu lassen, wenn in irgendeinem Zusammenhang das Wort ©Magie© fallen sollte.

Programmtechnisch werfen diese Ortswechsel bei Verwendung unseres Adventuresystemes keinerlei Schwierigkeiten auf, denn es reicht aus, der Variablen SPIELER die Nummer des neuen Raumes zuzuweisen.

©Reibe Lampe© könnte daher folgendermaßen implementiert werden (Lampe = Objekt 3):

```
xxxx IF N=3 AND OB(3)=-1 THEN PRINT "D
IE WELT DREHT SICH ...":SPIELER=9:GOTO
 1Ø8Ø
```

Wenn dann auch noch die Raumbeschreibung von 9 der des alten Raumes entspricht, dürfte die Verwirrung des Spielers nach dem nächsten Zug unausweichlich bleiben. Dies gilt

besonders für den Fall, wenn die Meldung fortfällt oder ein anderer Text gewählt wird:

```
xxxx IF N=3 AND OB(3)=-1 THEN PRINT "O
.K. - NICHTS PASSIERT.":SPIELER=9:GOTO
 1Ø8Ø
```

Eine weitere Steigerung des Schwierigkeitsgrades bedeutet es, wenn der Ortswechsel des Spielers nicht einmal mehr von den Manipulationen irgendwelcher Gegenstände abhängig ist, sondern einzig und allein vom Zufall gesteuert wird.

Wir können mit der Konstruktion unseres Adventureprogrammes vorschreiben, daß das Durchqueren eines bestimmten Raumes unumgänglich ist.
Entspricht der Inhalt von SPIELER der Nummer dieses Raumes, wird eine zufällige Entscheidung darüber getroffen, ob der Spieler versetzt wird oder ob ihm zwecks Ausführung weiterer Handlungen ein Aufenthalt gestattet wird.

Welcher Spieler, der nach erfolgter Durchquerung eines Irrgartens und anschließendem Betreten eines scheinbar harmlosen Raumes mehr als einmal wieder in das Labyrinth zurückgeworfen wurde, wird das Risiko einer Wiederholung dieses Vorganges eingehen ?

Wieder zeigt ein Ausschnitt aus dem Adventure Goldrausch, wie dieses Handicap eingebaut werden kann:

```
11Ø AR-16: ...

221 PRINT "MIT EINEM STOLLEN.":RETURN
222 PRINT "MIT EINEM FELSENDOM.":RETUR
N

519 DATA 8,Ø,15,5,Ø,Ø
521 DATA 11,11,11,7,11,Ø
522 DATA Ø,15,Ø,Ø,Ø,Ø
```

```
1345 IF SPIELER=15 THEN IF RND(1)>.7 T
HEN SPIELER=16
```

Versetzt in den kritischen Raum wird der Spieler durch Verlassen des Vorraumes zum Labyrinth, zurzeit Raum acht, ein kurvenreicher Gang, in westlicher Richtung.

Aus diesem Raum kann er sich sofort wieder zurückziehen, oder er kann versuchen, herauszufinden, was die weitere Umgebung bringt.

Wenn der Zufall es will - und ob er will, legen wir mit dem Vergleichswert in der RND-Funktion fest -, gelangt er in den nächsten Raum (16), wenn nicht, findet er sich in einem Raum des Labyrinthes (Sammelraum 11) wieder, welcher ihm keine Anhaltspunkte zur Orientierung bietet.

War ihm das Glück nicht hold, wird er nach einigen Versuchen die Entscheidung treffen, daß es sich um einen weiteren Zugang des Irrgartens handelt, diesen Raum fortan meiden und somit einen Teil des Schatzes niemals finden.

VERSTECKTE ZUGÄNGE

können dem Spieler mindestens ebenso große Schwierigkeiten bereiten. Selbst wenn in der Zeile ©Ich kann nach ... © nur die Richtungen Osten und Westen genannt werden, muß deshalb die Welt im Norden nicht zu Ende sein:

ICH SEHE EINE DUESTERE FELSENHOEHLE,
EINEN GRIMMIG DREINBLICKENDEN BAEREN,
**NUGGETS*.*

ICH KANN NACH WESTEN, OSTEN.

O.K.

WAS SOLL ICH TUN ?N
DAHIN FUEHRT KEIN WEG

WAS SOLL ICH TUN ?O

Viele Spieler, soweit es sich um Anfänger handelt sogar die meisten, werden sich auf die gemachten Angaben verlassen und die Höhle niemals betreten. Denn dadurch, daß eine Untersuchung derselben, wie auch andere Aktionen, von der augenblicklichen Position möglich sind, wird jeder Verdacht auf eine großangelegte Höhlenwelt sofort zerstreut.

Erst eine Eingabe wie ©Betrete Höhle© führt zu neuen Erkenntnissen:

```
17Ø VERBEN$="UNTERSUCHE...  ...BETRETE
  "

22ØØ ON VN GOTO 5ØØØ,6ØØØ,7ØØØ,8ØØØ,9Ø
ØØ,1ØØØØ,13ØØØ

13Ø1Ø IF N=13 AND SPIELER=5 THEN SPIEL
ER=12: PRINT "O.K.":GOTO 1Ø8Ø
```

ACHTUNG: Bevor Sie sich von der Funktion dieser Zeilen überzeugen, ist es erforderlich alle weiteren Räume einzugeben, wie auch die Verbindungen der bisher vorhandenen zu ändern (Zeilen 2ØØ - 3ØØ, 5ØØ - 6ØØ).
Ebenso muß Zeile 11Ø an die neue Situation angepasst werden.

Das schöne an dieser Realisation ist, daß der Zugang nicht einmal getarnt wird, im Gegensatz zu einer anderen Art der Ausführung, welche sich gleichfalls, zumindest seitens der Adventureproduzenten, großer Beliebtheit erfreut.

Aufgabe des Spielers ist es nun nicht mehr, nur einen versteckten Raum aufzuspüren und diesen mittels richtig gewählter Worte zu betreten, sondern er muß in der Lage sein, einen Plan zu entwickeln, wie ein begehbarer Weg dorthin konstruiert werden kann.

Im einfachsten Fall handelt es sich um eine Tür, die zunächst verschlossen ist. Eine entsprechende Anzeige durch den Treiber wird sowohl durch die Objektbeschreibungen wie auch die Daten der für diesen Raum geltenden Zeile der Richtungstabelle gewährleistet:

```
18Ø OBJEKTE$=" ...TUER"
3xx T$="EINE TUER":RETURN
41Ø DATA .,.,...,Ø

    REM RAUM 1
2xx PRINT "IN EINER ZELLE.":RETURN
5xx DATA Ø,Ø,Ø,Ø,Ø,Ø
```

Dieses übliche Verfahren sieht nun vor, daß der Spieler bei einer Untersuchung der Zelle die Tür, mit einem massiven Schloss versehen, entdeckt.

©Oeffne Tuer© bzw. ©Oeffne Schloss© weisen auf den fehlenden Schlüssel hin, wodurch der Spieler sich veranlasst fühlt, alle weiteren Einrichtungsgegenstände des Raumes auf ihre Tauglichkeit zur Öffnung des Schlosses hin zu überprüfen.

Bei den meisten Adventures reicht die bloße Anwesenheit des Schlüssels im Inventory aus, die Übrigen verlangen auch noch eine zweckentsprechende Benutzung des Schlüssels, eine

Eigenschaft, die wir uns nun dank unserer Praxis erklären können.
Während die einen sich mit einem einfachen Test begnügen (OB(N)=-1), arbeiten die anderen, aufwendigeren Programme zusätzlich mit Flags.

Wie auch immer, steht die Tür erst offen, taucht eine entsprechende Richtungsangabe innerhalb der Kopfzeile auf, worauf wir auf eine Manipulation der für diesen Raum zuständigen Zeile des Feldes DURCHGANG (,) schließen.

```
xxxx IF N=y AND OB(z)=-1 AND SPIELER=r
 THEN PRINT "O.K.":DURCHGANG(r,ri)=nr:
GOTO 1Ø8Ø
```

```
y = Objektnummer Tür
z = Objektnummer Schlüssel
r = augenblicklicher Raum
ri= betreffende Himmelsrichtung
nr= neu zu erreichender Raum
```

Obige Zeile, eingesetzt in den für die Behandlung des Verbs ©Oeffne© vorgesehenen Programmteil, vergewissert sich davon, daß der Spieler im Besitz des richtigen Schlüssels (z) ist und vor der richtigen Tür steht (r). Anschließend werden die Angaben in der Richtungstabelle entsprechend geändert.

LIMITIERUNGEN UND ZÄHLER

halte ich für besonders geeignet, um ein Adventure aus der Masse der übrigen hervorzuheben. Vermitteln doch besonders sie jedem Abenteuerprogramm die Faszination der Realität und gestalten einige Abschnitte des Adventures zu wahren Denksportaufgaben.

Ihr Aufgabenbereich ist dabei weniger in einer aktiven Spielbeteiligung zu sehen, als vielmehr in einer Hintergrundkontrolle der Rahmenbedingungen, in die das Spiel eingebettet wird.

An erster Stelle muß hier wohl die Einschränkung des Inventories genannt werden, da ein Gewichtslimit inzwischen zum guten Ton dieser Computerspiele gehört.
Verständlich - erstens wirkt es unglaubhaft, wenn ein einzelner Mensch, wir gehen davon aus, daß es sich nicht um ein Supermann-Adventure handelt, beliebig viele Objekte mit sich herumschleppen kann, und zweitens kann das Spiel auch zu einfach werden, wenn stets alle Handlungsgegenstände sogleich zur Verfügung stehen.

Technisch gesehen überwacht ein Zähler die Anzahl der Gepäckstücke, die der Spieler mit sich führt, und ist die vom Programmierer vorgesehene Tragkraft der Hauptperson ausgelastet, wird die Aufnahme weiterer Gegenstände verweigert.

Es muß also vor Beginn aller Handlungen die maximal erlaubte Stückzahl festgelegt werden:

```
11Ø ... IMAX=5
```

Die Inventory-Routine lassen wir ansonsten wie sie ist, denn es ist der Vorgang des Nehmens, dem wir erhöhte Aufmerksamkeit schenken müssen.
Allen Aktionen mit dem Verb ©Nimm© wird nun eine weitere gemeinsame Voraussetzung zugrunde gelegt, weshalb sich zum zweiten Male eine Änderung der Zeile 6ØØØ anbietet.
Leider sieht diese Planung nicht nur eine Umstrukturierung des betreffenden Blockes vor, sondern sie macht auch unseren Traum eines universell einsetzbaren Treibers mit allen Funktionen und Restriktionen zunichte.

Aus diesem Grunde ändern wir die Sprungtabelle in Zeile 22ØØ und führen den erforderlichen Test vor Beginn der eigentlichen Befehlsausführung durch:

```
22ØØ ON VN GOTO 5ØØØ,221Ø,7ØØØ,8ØØØ,9Ø
ØØ,1ØØØØ
22Ø9 REM ******** TEST OB PLATZ IM INV
221Ø ANZAHL=Ø
222Ø FOR I=1 TO AO
223Ø IF OB(I)=-1 THEN ANZAHL=ANZAHL+1
224Ø IF ANZAHL=IMAX THEN PRINT"NEIN DA
NKE. - ICH TRAGE SCHON GENUG !":GOTO 1
Ø8Ø
225Ø NEXT I
226Ø GOTO 6ØØØ: REM *AUSFUEHRUNG NIMM
227Ø REM ***************** ENDE INVTEST
```

Während diesem Zähler eine passive Rolle zukommt, und die Routine nur im Rahmen einer genau definierten Aktion aufgerufen wird, beschneiden andere Hintergrundkontrollen den Spieler nicht nur in der Freiheit seiner Handlungen, sondern können sogar zum Spielende führen.

Diese Behauptung gilt insbesondere für die folgenden Programmzeilen, deren Einbau in unser System dem Tüpfelchen auf dem ©i© entspricht.

AVAILABLE LIGHT

ist die gebräuchliche Bezeichnung für eine bei einigen Adventureprogrammen anzutreffende Eigenart, die mit zusätzlichen Schwierigkeiten die Spieler zur Verzweiflung treiben kann.

Für Adventurespieler aus Passion scheinen die üblichen Hindernisse nicht ausreichend zu sein, und da bekanntermaßen nichts schwieriger ist, als das Leben in der Alltagswelt, muß zur Beseitigung dieses untragbaren Zustandes nichts weiter getan werden, als eben diese Welt noch perfekter nachzuahmen.

Ohne ©verfügbares Licht© stehen wir im Dunkeln und werden nur schwerlich etwas sehen, geschweige denn präzise handeln können. Kommt dann noch die Tatsache hinzu, daß wir uns nicht in unserem Schlafgemach, wo dieser Zustand weitaus weniger hinderlich wäre, sondern in einer naturbelassenen, wilden und zerklüfteten Felsenlandschaft befinden, wird jeder weitere Schritt zu einem tödlichen Risiko.

Das kommt uns Adventureproduzenten natürlich sehr gelegen, und beim Einbau der nächsten Zeilen in unsere Adventures bereitet uns die Vorstellung der Reaktionen unserer Freunde, denen wir unser Werk selbstverständlich sofort nach Fertigstellung präsentieren wollen, eine zusätzliche Freude.

Glücklicherweise eignet sich jedes Abenteuerprogramm für einen Einbau dieses Hindernisses. Oft stehen weiträumige Höhlen oder unterirdische Gangsysteme zur Verfügung, andere Programme spielen sich in einem Haus ab; alles Orte, die einer künstlichen Beleuchtung bedürfen. Sollte tatsächlich einmal die Situation eintreten, daß die ganze Handlung unter freiem Himmel abläuft, muß eben der Lauf der Sonne nachvollzogen werden.

Um einen Sonnenauf- und -untergang realisieren zu können, reicht der Einsatz eines weiteren Zählers.

Zunächst legen wir fest, wie lange der Tag (die Nacht) dauern soll, wobei wir uns der Eingaben des Spielers als Zeiteinheit bedienen.
Nehmen wir an, 5Ø Züge nach Sonnenaufgang beginnt jeweils die Nacht. Dieser Wert wird ebenso typisch für jedes einzelne Adventure sein, wie die Anzahl der erreichbaren Punkte, weshalb die entsprechende Variable LM (Lichtmenge) gleichfalls bei Programmstart initialisiert wird:

```
11Ø ... IMAX=5:LM=5Ø:LICHT=-1
```

Ihr nun schon recht umfangreiches Wissen zum Thema Adventure erlaubt es Ihnen, die Variable LICHT als Signalschalter zu interpretieren, und zwar in dem Sinne, daß eine -1 eine für alle Handlungen ausreichende Lichtmenge, Ø aber absolute Dunkelheit bedeutet.

Die beiden möglichen Stellungen dieses Schalters zeigen der Ausgaberoutine unseres Treibers, ob der Spieler etwas sehen kann, oder ob er im Dunkeln tappen muß:

```
1Ø91 REM ****************** KEIN LICHT
1Ø92 IF LICHT THEN 11ØØ
1Ø93 PRINT "ICH WEISS NICHT GENAU WO I
CH BIN."
1Ø94 PRINT "ES IST ZU DUNKEL UM ETWAS
ZU SEHEN."
1Ø95 PRINT: PRINT "AUCH DIE AUSGAENGE
SEHE ICH NICHT MEHR."GOTO 132Ø
```

Nach Zeile 1Ø92 werden die bislang benutzten Ausgaberoutinen abgearbeitet werden, falls Licht vorhanden ist.

Trifft das Gegenteil zu, informieren die Zeilen 1Ø93 bis 1Ø95 auf die gewohnte Art und Weise darüber, daß es nichts zu sehen gibt.
Mit dem Drucken der Trennungslinie (132Ø) kann der Programmverlauf durch die üblichen Zeilen fortgesetzt werden.

Ansonsten müssen wir nur noch für die jeweils richtige Stellung des Schalters LICHT sorgen, was nach Ablauf der mit LM festgelegten Anzahl von Spielzügen zu geschehen hat.

Der Zugzähler kann zu dieser Vergleichsoperation nicht herangezogen werden, weil mit einem Tag-Nachtwechsel auch ein Rücksetzen des Zählers auf null erforderlich wird; führen wir daher die Variable Lichtwechsel (LW) ein:

```
11Ø ... IMAX=5:LM=5Ø:LIGHT=-1:LW=Ø

1Ø8Ø ... ZUG=ZUG+1:LW=LW+1
1Ø84 IF LW=LM THEN GOSUB 3ØØØ


2999 REM * UNTERPROGRAMM LICHTSCHALTER
3ØØØ IF LICHT=-1 THEN LICHT=Ø:GOTO 3Ø2
Ø
3Ø1Ø IF LICHT=Ø THEN LICHT=-1
3Ø2Ø LW=Ø
3Ø3Ø RETURN
```

Zu Beginn einer neuen Eingaberunde wird überprüft, ob die vorgesehene Zeitdauer (LM) einer Periode erreicht ist. Wenn ja, wird das Unterprogramm Schalter aufgerufen und der Zustand von LICHT gewechselt. Mehr ist nicht erforderlich, den Abenteurer einem Tag und Nachtwechsel auszusetzen, was die Suche nach einer Lampe erforderlich machen kann.

Für Goldrausch wollen wir uns jedoch nicht mit der Simulation des Sonnenlaufes begnügen, sondern machen das Geschehen vom Besitz einer Lampe abhängig.
Dadurch stellen wir dem Spieler die zusätzliche Aufgabe, immer genügend Brennstoff für die Lichtquelle bereitzuhalten.
In unserem Falle wird es sich dabei um Öl handeln, welches er in der Höhle des Bären finden kann. Dort füllt er es in die Flasche, die er zuvor wieder an sich genommen haben muß, und anschließend in die Lampe.
Jedes Füllen setzt dabei den Zähler Lichtwechsel wieder auf den Maximalwert.

Um die Sache zusätzlich zu komplizieren, wird die Laterne, zu dem Zeitpunkt, zu dem der Spieler sie findet, noch einen Rest Öl enthalten. Betritt er dann, ohne die Lampe nachzufüllen, die Tiefen der Mine, ist er unrettbar verloren. Ein weiterer Schritt, und der Abenteurer stolpert und bricht sich das Genick.

Da wir für das restliche Territorium keine Dunkelheit vorgesehen haben, besteht der erste Schritt zur Verwirklichung dieser Version in einer Änderung der Startwerte.
Eine Umschaltung erfolgt immer nur dann, wenn der Zähler Lichtwechsel gleich dem Inhalt von Lichtmenge ist.

Um dem Spieler zunächst jedoch eine fast unbegrenzte Anzahl von Spielzügen zu erlauben, legen wir LM mit null und LW mit eins fest, und stellen dadurch sicher, daß LW nie gleich LM sein kann:

```
11Ø .. IMAX=5:LM=Ø:LICHT=-1:LW=1:L1=2Ø
```

L1 kommt hinzu, weil die noch vorhandene Füllung geringer sein soll, als die später erfolgenden Nachfüllungen. Die übrigen Zeilen können vom vorangegangenen Beispiel übernommen werden.

Aktiviert wird diese Routine erst mit dem ersten Zünden der Lampe. Die Ausführung dieses Befehls wird neben der Ausgabe von ©O.K.© auch die Zähler korrekt initialisieren:

```
11Ø3Ø IF N=23 AND OB(23)=-1 THEN PRINT
•"O.K. - DIE LAMPE BRENNT.":LM=L1:LW=1
:GOTO 1Ø8Ø
```

Für die nächsten zwanzig Aktionen steht dem Spieler nun ausreichend Licht zur Verfügung. Mit dem Nachfüllen der Lampe kann er diesen Wert auf fünfzig vergröbern:

```
12Ø1Ø IF N=23 AND OB(23)=-1 AND FL(6)
THEN L1=5Ø:LM=5Ø:LW=1:PRINT "O.K.":GOT
O 1Ø8Ø
```

Um ein Nachfüllen des Brennstoffvorrates sowohl bei brennender als auch bei gelöschter Flamme zu ermöglichen, weisen wir der Variablen L1, wie auch LM, die Anzahl der nun durchführbaren Spielzüge zu.

Für den Fall, daß der Spieler sich an die Oberfläche begibt oder aus anderen Gründen die Lampe löscht, müssen wir den augenblicklichen ©Tankinhalt© retten.
Ergänzen Sie dazu zunächst VERBEN$ um das Wort ©Loesche© und sehen Sie in Zeile 22ØØ ein Sprungziel für die Behandlung dieser Aktion vor.

```
14ØØØ IF N=23 AND OB(N)=-1 THEN L1=LM-
LW:PRINT "O.K.":GOTO 1Ø8Ø
```

Selbstverständlich müssen diese Werte auch mit dem Abspeichern eines Spieles gerettet werden:

```
1625 PRINT #2,SPIELER:PRINT #2,L1:PRIN
T #2,LM:PRINT #2,LW
```

```
1725 INPUT #2,SPIELER:INPUT #2,L1:INPU
T #2,LM:INPUT #2,LW
```

Damit haben wir sichergestellt, daß die Sache mit dem Licht auch wirklich ihren vorgezeichneten Verlauf nehmen muß.

Um dem Spieler aber die plötzliche, unangenehme Überraschung des unabwendbaren Endes zu ersparen, denn sollte er sich in den Tiefen der Erde befinden, während ihm das Öl ausgeht, wird er seine Haut nicht mehr retten können, ist eine Warnung fairerweise angebracht.

Folgende Zeile wird ihn kurz vor Brennschluss der Lampe von der drohenden Gefahr unterrichten:

```
135Ø IF LM-LW<15 THEN PRINT "DAS LICHT
 DER LAMPE WIRD SCHWAECHER."
```

Ich hoffe, die vorangegangenen Beispiele werden Ihnen bei der Verwirklichung Ihrer eigenen Ideen helfen.
So wird es Ihnen sicherlich nicht schwerfallen, in Ihren Programmen beispielsweise dafür Sorge zu tragen, daß der Spieler rechtzeitig. Nahrung zu sich nehmen muß, damit er weiterhin Dinge in die Hand nehmen und Aktionen durchführen kann.

Ebenso können Sie mittels eines weiteren Zählers Reaktionen von der verstrichenen Zeit abhängig machen.

ZUFÄLLE

Erdbeben, die eine Gesteinslawine auslösen, plötzlich auftauchende Monster, die natürlichen Feinde der Hauptperson im Adventure, oder auch sich im unpassenden Moment öffnende Falltüren sind weitere allseits beliebte Methoden, um den Spieler vom rechten Wege abzubringen.
Fast immer wird das Auftreten dieser Ereignisse vom Zufall gesteuert und das sogar in doppelter Hinsicht.

Ein Ungeheuer kann einen bestimmten Raum beleben und dem Spieler den Kopf abreißen:

```
136Ø IF SPIELER=1Ø THEN MØ$="DIESMAL H
ATTE ICH KEINEN HONIG FUER   DEN BAERE
N.":GOTO 45ØØ
```

Genauso gut kann die Falle aber nur manchmal zuschnappen und wird zu anderen Zeiten dem Spieler kein Haar krümmen:

```
137Ø IF SPIELER=2Ø AND RND(1)>.8 THEN
MØ$="DER BODEN BRACH UNTER MIR EIN.":G
OTO 45ØØ
```

Die dritte Version stellt eine Kombination der bereits vorgestellten Elemente dar.
Grundsätzlich wird man auch von einer Beweglichkeit der Feinde des Abenteurers ausgehen können, weshalb auch dem Monster ein gewisser Abschnitt der Adventurewelt als Spielwiese zur Verfügung gestellt werden sollte:

```
138Ø IF (SPIELER=13 OR SPIELER-14 OR S
PIELER=16) AND RND(1)>.8 THEN MØ$="WIE
DER DER BAER.":GOTO 45ØØ
```

VOM TEXT- ZUM GRAFIKADVENTURE

Neu vorgestellte Adventureprogramme weisen neben den besprochenen Features meist auch eine hervorragende optische Präsentation auf. Für jeden Raum des Adventures existiert ein Bild, welches den Spieler mehr oder weniger gut über spielwichtige Dinge und deren Umfeld informiert.

Zur programmtechnischen Ausführung dieser Grafikadventures stehen dem Programmierer prinzipiell drei Wege offen.

Zunächst kann er auf der Diskette für jeden Raum und jeden Gegenstand eine entsprechende Grafik speichern und diese bei Bedarf in den Bildschirmspeicher des Computers laden. Dieser Weg wird heutzutage bei fast allen Grafikadventures eingeschlagen, einerseits weil sich mittels Videodigitalisierer oder anderer Hilfsmittel, wie beispielsweise auch dem Koalapad, großartige Ergebnisse erzielen lassen, andererseits besteht der zusätzliche Programmieraufwand nur aus einem einzigen Unterprogramm, welches abhängig von der Raumnummer (SPIELER) eine Datei laden muß, und daß im günstigsten Fall aus einer einzigen Zeile besteht.
Natürlich wird entsprechend viel Platz auf der Diskette benötigt (jetzt wissen Sie, warum manche Adventures über drei und mehr Disketten gehen), wie der Spieler auch die Ladezeit geduldig ertragen muß.

Die zweite Methode arbeitet mit einem Grafikinterpreter, einem Programmteil des Adventures, welcher über einen Line- und Paint-Befehl verfügt, und dem jeweils die Daten aller Umrisse und Farben zur Verfügung gestellt werden müssen.
Dieses Verfahren geht am sparsamsten mit dem zur Verfügung stehendem Speicherplatz um, denn für jeden Punkt ist im Allgemeinen nur ein Byte erforderlich, und wenn die Linien als Linienzüge ausgeführt werden, reicht zu ihrer Speicherung ebenfalls jeweils eine Speicherstelle.

Übrigens arbeitet der im folgenden vorgestellte Grafik-Programmer nach diesem Prinzip, wenn er auf Druck von CTRL Z ein Bild neu zeichnet.

Das dritte Verfahren schließlich besticht durch seine Schnelligkeit, denn jede für den Aufbau der Grafik notwendige Anweisung wird direkt ausgeführt. Voraussetzung ist natürlich, daß das Betriebssystem des Computers entsprechende Befehle zur Verfügung stellt und daß der Arbeitsspeicher die erforderlichen Anweisungen aufnehmen kann.

Bei der weiteren Entwicklung unserer Adventures wollen wir uns auf die letzten beiden Methoden beschränken, denn zu deren Anwendung bedürfen wir weder technischer Hilfsmittel noch aufwendiger Unterprogramme.

Beide Lösungen erfordern natürlich einige kleine Änderungen an unserem Treiberprogramm, denn die Ausgaben werden nun an anderen Stellen des Bildschirmes gemacht werden müssen.

Laden Sie also bitte noch einmal die Miniversion von Goldrausch, und ändern Sie GRAPHICS Ø (Zeile 1ØØØ) in GRAPHICS 8 um.
Nach Einschalten der Grafikbetriebsart stehen uns nur noch vier Textzeilen im unteren Teil des Monitorbildes zur Verfügung. Dies bedeutet zunächst einmal eine Einschränkung unserer Ausgaben; nur das unbedingt notwendige Maß kann noch erlaubt werden.
Auf die Leerzeile und den Trennungsstrich können wir verzichten, weshalb wir zunächst die Zeilen 131Ø und 132Ø löschen. Auch der Position-Befehl in 139Ø ist nicht mehr nötig, denn die letzte Zeile wird sowieso mit Sicherheit erreicht werden.
Starten wir nun das Programm, so bedeuten die zehn Leerzeilen in 1Ø9Ø nur noch sinnloses Geflimmer, weshalb diese Programmzeile ebenfalls gelöscht werden darf.

Da wir immer noch mehr als vier Zeilen beschreiben, lassen wir auch die Raumbeschreibung wegfallen. Wenn wir uns mit der Grafik etwas Mühe geben, dürfte diese sowieso überflüssig sein.
Löschen Sie daher die Zeile 11ØØ wie auch alle Raumbeschreibungen innerhalb der Programmzeilen von 2ØØ bis 299.

Unser nächster Schritt besteht nun darin, die Ausgabe der freien Wege und die Aufforderung des Spielers zur Eingabe zusammenzufassen.
Damit nicht mehr als eine Bildschirmzeile benötigt wird, ändern wir die Zeilen 1Ø11 bis 1Ø16 so ab, daß jeweils nur noch der Anfangsbuchstabe, das Komma und ein Blank in T$ geschrieben werden. Anschließend ergänzen Sie Zeile 131Ø um ein Semikolon, damit der Zeilenvorschub unterdrückt wird, und löschen außerdem Programmzeile 126Ø, denn selbst bei Ausgabe aller sechs Himmelsrichtungen wird das Zeilenende nicht erreicht werden.

Nach dem Programmstart müßten Sie somit folgendes Bild erhalten:

```
ICH SEHE VIELE GROSSE BAEUME.
O.?
```

```
O.K.
ICH SEHE VIELE GROSSE BAEUME,
EINIGE FELSBROCKEN.
W, O.?
```

In der obersten Zeile stehen demnach jetzt die Mitteilungen, gefolgt von den Objektbeschreibungen. Für deren Ausgabe stehen dem Adventure zwei Bildschirmzeilen zur Verfügung, eine dritte Zeile wird die an den Spieler ausgegebene Meldung wieder aus dem Bild scrollen. Achten Sie daher bei Ihren eigenen Adventurespielen darauf, im Falle eines Grafikadventures möglichst kurze Beschreibungen zu wählen. Dies ist inzwischen durchaus gerechtfertigt, denn jeder Spieler wird sowieso nur noch auf das Bild und kaum auf den Text achten.
Auch unsere Miniversion von Goldrausch werden Sie unter diesem Gesichtspunkt editieren müssen.

DER BILDAUFBAU

Den Aufbau der jeweiligen Grafik überlassen Sie einigen Unterprogrammen, die vom Treiber aus aufgerufen werden.
Allerdings gilt es zu beachten, daß diese Druckroutine nicht nach jeder Eingabe des Spielers aufgerufen werden muß, denn warum sollte nach jeder Aktion innerhalb eines Raumes dieser neu gezeichnet werden ?

Führen wir daher eine weitere Kontrollvariable ein:

```
1Ø9Ø IF ALT<>SPIELER THEN GOSUB 3ØØØØ
11ØØ ALT=SPIELER
```

muß ein Raum auf dem Bildschirm dargestellt werden, so wird der Block mit den Grafikroutinen angesprungen.

Anschließend wird dessen Nummer an die Variable ALT übergeben. Entfernt der Spieler sich dann vom Ort des Geschehens, so ist die Bedingung in Zeile 1Ø9Ø erfüllt und es wird wiederum Zeile 3ØØØØ aufgerufen, welche anhand der Raumnummer die richtige Zeichenroutine wählt:

```
3ØØØØ ON SPIELER GOSUB 31ØØØ,32ØØØ,33Ø
ØØ
3ØØ1Ø RETURN

31ØØØ REM RAUM 1 ZEICHNEN
31999 RETURN

32ØØØ REM RAUM 2 ZEICHNEN
31999 RETURN
```

Anschließend wird der Programmlauf mit der Ausgabe der Objektbeschreibungen in gewohnter Weise fortgesetzt.

Sollten Sie es nun vorziehen, den Aufbau der Grafiken mittels eines Interpreters vorzunehmen, so bauen Sie ab Zeile 3ØØØØ die entsprechenden Routinen des später vorgestellten Grafik-Programmers ein.
Sie benötigen nur die zwei Module Bild laden und Bild zeichnen, welche bei Programmzeile 7ØØØ beziehungsweise bei Zeile 5ØØØ beginnen.

5. KAPITEL
- ADVENTUREPRAXIS -

Dieses Kapitel wird Ihnen weniger zeigen, wie man©s macht, sondern zeigt Ihnen am Beispiel vollständiger Programme, was man machen kann.

Es folgen die kompletten Listings von drei Adventurespielen, die alle unter Verwendung der in diesem Buch entwickelten Programmteile und Routinen geschrieben wurden.

Zum einen handelt es sich dabei um *Goldrausch*, ein Adventure, zu dem schon fast zu viel gesagt wurde; als zweites Programm wäre das *Verzauberte Schloss* zu erwähnen. Die Krönung des Ganzen ist aber *Space Mission*, ein trickreiches Spiel, welches als Grafikadventure realisiert wurde.

Leider ist es jedoch so, daß Sie, wenn Sie dieses Buch durchgearbeitet haben, die Programme fast wie eine Anweisung zur Lösung lesen können. Deshalb wäre es empfehlenswert, die Arbeit des Eintippens mit jemandem zu teilen.

Dies gilt insbesondere für die Programmzeilen ab Nummer 5ØØØ, da hier das Spielgeschehen programmiert ist.

Im Anschluß an die Spiele finden Sie dann die versprochenen Programmgeneratoren, den Grafik-Programmer wie auch ©Venturefix©.
Beide Programme zusammen erlauben es Ihnen, menügesteuert Adventures zu erzeugen und diese sogar mit einem Minimum an Arbeit zu eindrucksvollen Grafikprogrammen auszubauen.
Die notwendigen Erläuterungen und Bedienungshinweise stehen den betreffenden Programmlistings voran.

SPIELANLEITUNG: ADVENTURES

Nachdem Sie das betreffende Programm geladen und mit RUN gestartet haben, erscheinen nach dem Titelbild und weiteren allgemeinen Spielhinweisen die ersten Beschreibungen und Mitteilungen auf dem Monitor.
Sie müssen zwischen der oberen und unteren Bildschirmhälfte unterscheiden: In der oberen Hälfte finden Sie eine Beschreibung des Ortes, an dem Sie sich gerade befinden, daran anschließend werden Ihnen alle sichtbaren Gegenstände aufgezählt, und in einer weiteren Zeile werden sämtliche Richtungen genannt, in die sie sich bewegen können.
Die untere Hälfte dient hingegen dazu, Ihre Befehle an die Spielfigur einzugeben, ebenso werden dort die von der Durchführung Ihrer Eingabe abhängigen Informationen an Sie ausgegeben werden.

IHRE EINGABEN

Ihre üblichen Eingaben werden meist aus zwei Worten, einem Verb und einem Objekt, bestehen. Überprüfen Sie zunächst, welche Eingabelängen das vorliegende Adventure fordert, meist sind die ersten drei oder vier Buchstaben zur Identifikation eines Wortes ausreichend.
Achten Sie darauf, Bezeichnungen zu verwenden, wie sie auch innerhalb der Beschreibungen verwendet werden.
Das bedeutet allerdings nicht, daß Sie auf eine Mitteilung wie:

    ICH SEHE EINE ROTE TÜR.

    WAS SOLL ICH TUN:

mit OEFFNE ROTE TUER reagieren, meistens wird die Eingabe OEFFNE TUER völlig ausreichend sein.

Zusätzlich werden Ihnen einige weitere Eingaben erlaubt sein, die sogenannten Ein-Wort-Befehle, welche weniger dem Spielverlauf dienlich sind, als daß sie Ihnen gewisse Serviceleistungen anbieten, um die Spielweise zu vereinfachen.

RÄUMLICHE FORTBEWEGUNG

Sie können sich generell immer in die angegebenen Richtungen bewegen. Dazu reicht es aus, den Anfangsbuchstaben der jeweiligen Richtung einzugeben (außer für Bewegungen nach oben, hier müssen Sie zur Unterscheidung von Osten OB eingeben): Norden, Süden, Osten, Westen: N, S, Ø, W; Oben, Unten: OB, U.

Es gibt jedoch auch Ausnahmen und manchmal kann eine Eingabe wie BETRETE ... nutzbringend sein.

MANIPULATION VON OBJEKTEN

Natürlich wollen Sie in der imaginären Welt nicht nur auf eine Besichtigungsreise gehen, sondern Sie müssen handeln und eine Aufgabe lösen.
Dies setzt voraus, daß Sie die Gegenstände dieser Welt zunächst erst einmal entdecken, anschließend können Sie sie UNTERSUCHEN, NEHMEN, BENUTZEN usw. (Wenn Sie einmal nicht mehr weiter wissen, sollten Sie sich an diese Worte erinnern !).
Um möglichst schnell ans Ziel zu kommen, sollten Sie bei Ihrem weiteren Vorgehen möglichst logisch und überlegt vorgehen.

Nehmen wir an, Sie finden irgendwo einen Schlüssel. Sie denken ©prima, doch was soll ich damit ?©
Im wirklichen Leben würden Sie ihn sich ansehen, würden ihn genau untersuchen und dann entscheiden, ob Sie ihn

liegenlassen oder ob sie ihn für brauchbar oder sogar wertvoll halten und daher mitnehmen. Diese einzelnen Schritte müssen sie im Adventure unbedingt nachvollziehen.

Geben Sie ein: UNTERSUCHE SCHLÜSSEL. Als Antwort erhalten Sie vielleicht: ES IST DER SCHLÜSSEL ZU MEINER WOHNUNG oder: ES IST EIN AUTOSCHLÜSSEL oder: ER IST AUS PUREM GOLD. Natürlich würden Sie keinen dieser Schlüssel liegen lassen, sondern alle mitnehmen, weshalb Sie auch: NIMM SCHLÜSSEL eingeben. Auf dem Bildschirm erscheint als Mitteilung zumindest ein O.K. (Dieses O.K. werden Sie oft sehen, es bedeutet, daß der Adventureinterpreter Ihre Eingabe verstanden und ausgeführt hat.), wenn nicht sogar weitere Reaktionen folgen.
Natürlich wird der Schlüssel nun nicht mehr in der Kopfzeile ICH SEHE erscheinen, denn da Sie ihn eingesteckt haben, befindet er sich jetzt in Ihrer Manteltasche oder Ähnlichem, jedoch nicht mehr im Raum.

Sollten sich auf diese Art und Weise zahlreiche Gegenstände in Ihren Taschen angehäuft haben, verlieren Sie vermutlich die Übersicht, weshalb Ihnen auch die Möglichkeit,

INVENTUR

zu machen geboten wird. Immer wenn Sie wissen wollen, ob Sie nicht zu viel unnützes Zeug mit sich herumtragen, oder was Schuld daran war, daß Sie im Treibsand versunken sind, oder aber, welche Gegenstände geeignet wären, eine bestimmte Aufgabe zu erfüllen, so geben Sie einfach INV ein.

Als Antwort erscheint sofort: *ICH TRAGE FOLGENDES MIT MIR:* sowie eine Liste all dieser Objekte.

SPIELUNTERBRECHUNG

Sie werden es wohl kaum schaffen, ein Adventure in einem Anlauf zu lösen, und natürlich wollen Sie am nächsten Tag nicht wieder von vorne beginnen.
Daher erlauben es fast alle dieser Spiele, den augenblicklichen Spielstand abzuspeichern.

Wenn Sie ein Spiel vorläufig beenden wollen, so geben Sie einfach SAVE ein. Bessere Programme fragen Sie dann nach einem Namen, unter dem die augenblickliche Situation abgespeichert werden soll, einfache erfüllen diese Aufgabe sofort.

Dieses Abspeichern empfiehlt sich auch zwischendurch: So könnte beispie1sweise die Situation auftreten, daß Sie mit falschem Schuhwerk eine steile Felswand besteigen wollen. Hier ist die Wahrscheinlichkeit sehr groß, daß Sie abstürzen und sich Ihr Genick brechen; als Folge davon müßten Sie von vorne beginnen und alle Eingaben wiederholen.
Vor dieser uninteressanten Tätigkeit hätte Sie ein rechtzeitiges Abspeichern bewahrt !

FORTSETZUNG EINES UNTERBROCHENEN SPIELS

Natürlich müssen Sie zunächst die entsprechende Kassette / Diskette einlegen, dann geben Sie einfach LOAD ein. Eventuell werden Sie gefragt, unter welchem Namen Sie den Spielstand abgespeichert hatten, andernfalls reicht das Drücken der <RETURN>-Taste allein, um das Spiel an der unterbrochenen Stelle fortzusetzen.

SONSTIGE HINWEISE

Wenn das Titelbild Sie nicht entsprechend informiert, müssen Sie zunächst herausfinden, welchem Typ das geladene Adventure angehört.
Entweder haben Sie eine Aufgabe zu lösen, einen Ausgang zu suchen oder Schätze zu sammeln.
Untersuchen Sie alles, was Sie sehen, und verhalten Sie sich immer logisch (so brauchen Sie zum ÖFFNEn einer VERSCHLOSSENEn TÜR zunächst einen SCHLÜSSEL, eine BRECHSTANGE o. ä.), zumindest so lange, bis Sie darüber informiert werden, daß Magie im Spiel ist. Dann dürfen Sie auch völlig sinnlose Eingaben machen und darauf hoffen, möglichst schnell auf die richtige Kombination zu kommen.
Achten Sie vor allem auf Mitteilungen der Art DAS GEHT IM MOMENT NICHT, denn damit soll Ihnen deutlich gemacht werden, daß Sie sich auf dem richtigen Weg befinden.
Leider sind aber zur ·Durchführung der Aktion noch nicht alle Bedingungen erfüllt (z. B. fehlt der SCHLÜSSEL).

Sollte ein Verb oder das Objekt nicht verstanden werden, so wird Ihnen darüber Mitteilung gemacht, und Sie sollten versuchen, die gleiche Aktion mit anderen Worten zu umschreiben.

Eine letzte Möglichkeit wird durch ICH VERSTEHE NICHT, WAS DU MEINST ausgedrückt. Die von Ihnen benutzten Worte sind zwar bekannt, geben in Ihrer Zusammenstellung jedoch keinen Sinn. Oder aber, die geschilderte Handlung wurde vom Programmierer des vorliegenden Adventures nicht vorgesehen.

TIPPS ZUR LÖSUNG EINES ADVENTURES

Wenn Sie in verfahrenen Situationen überhaupt nicht mehr weiter wissen, können Sie es zunächst mit HELP versuchen.

Gelangen Sie auf diese Weise nicht in den Besitz weiterer Informationen, rufen Sie sich alle bisher entdeckten Gegenstände in die Erinnerung zurück und gehen alle Kombinationsmöglichkeiten mit den bislang bekannten Verben durch. Vielleicht stoßen sie so auf eine erfolgversprechende Aktion, denn es kommt mit ziemlicher Sicherheit jedem Objekt eine Aufgabe zu; es nur zur Dekoration eines Raumes einzusetzen, dazu wäre der Programmieraufwand zu groß.

Für diese Aufgabe hilfreich ist in jedem Falle eine Liste aller vom Programm verstandenen Worte; möglicherweise wird diese sogar vom Adventure selbst auf den Befehl VOKabeln / VOCabulary hin erstellt.

Als ebenso nützlich erweist sich eine Karte, auf der Sie alle Räume mit Ihren Verbindungen eintragen, wobei Sie natürlich auch die Fundorte der Gegenstände nicht vergessen.

Sollten Sie unverhofft, oder auch gewollt, weil unvermeidlich, in ein Labyrinth geraten, so ist die Erstellung einer solchen Karte gar nicht so einfach, wie Sie es sich jetzt vielleicht vorstellen.
Denn statt wirklich vorhandener Räume hat man oft die dreifache Anzahl kartographiert.
Der einzige Trick, um dieses zu verhindern, besteht darin, in jedem Raum des Irrgartens einen Gegenstand aus dem Inventory abzulegen und jedem einzelnen Aufenthaltsort damit ein unverwechselbares Kennzeichen zu geben.

Und den wichtigsten Tipp noch einmal:
Speichern Sie Ihr Spiel zwischendurch immer wieder ab !

Das verzauberte Schloss

Dieses Adventure möchte ich Ihnen besonders ans Herz legen, hier wird besonders tief in die Trickkiste gegriffen, um dem Abenteurer einige Überraschungen bieten zu können.
Geplant ist es eigentlich, daß der Spieler sich zunächst über die gestellte Aufgabe klarwerden muß, ein Problem, welches Sie bedauerlicherweise bereits während des Eintippens lösen werden.

Zahllose der in diesem Buch vorgestellten Techniken wurden verwandt, einige Programmteile sollen hier als Beispiele genannt werden.
Auffällig ist zunächst die Zeile 181, hier sehen Sie, wie Sie vorzugehen haben, wenn die bei der Programmierung zur Verfügung stehenden drei Bildschirmzeilen nicht für die Eingabe der notwendigen Beschreibungen ausreichen.
Weiterhin fallen die Zeilen 6ØØ bis 61Ø auf, Mitteilungen werden durch Aufruf von Unterprogrammen ausgegeben, eine Technik, die Sie insbesondere bei häufig auszugebenden Meldungen anwenden sollten.
Flags wurden hier verwandt, um eine genaue Kontrollmöglichkeit der verschiedenen Stadien vor Verlassen des Schlosses zu haben.
Sind alle Voraussetzungen erfüllt, wird ein entsprechender Durchgang geöffnet, und die Adventurewelt vergrößert sich um 16 Räume.
Neben INVENTUR, SAVE und LOAD ist auch der Befehl HELP implementiert, allerdings nur in allgemeiner Form.
Dennoch wird ein Adventureneuling die Hilfe zu schätzen wissen.

Anmerkungen zum Listing "Das verzauberte Schloss"

Das im Buch abgedruckte Listing enthält einige Fehler, die wie folgt korrigiert wurden:

Zeile 16Ø
VERBEN$ muss auf 9Ø Zeichen dimensioniert werden.

Zeilen 5Ø8, 5Ø9
Korrektur unsinniger Raumdaten.

Zeilen 45ØØ, 461Ø
Korrektur von Rechtschreibfehlern.

Zeile 482Ø
Überzähliger PRINT-Befehl entfernt.

Zeile 1415Ø
Fehlender Sprung zur Schlussroutine ergänzt.

Die Zeile 5Ø27 muss wie folgt eingetippt werden, damit sie vom Atari-BASIC korrekt übernommen wird:

```
5Ø27IF N=23ANDSPIELER=14THEN MØ$="ES I
ST EINE ZAUBERKUECHE. DER ZAUBERERKOMM
T, MACHT EINEN STEIN AUS MIR.":G.45ØØ
```

Darüber hinaus bleibt weit mehr Verbesserungspotential.

```
1 REM DAS VERZAUBERTE SCHLOSS Ver 1.1
2 REM (C) 1984  BY J. WALKOWIAK
4 REM ----------------------------
1Ø GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
4,1,Ø:FOR I=1 TO 5:PRINT :NEXT I
15 PRINT "
    "
16 PRINT "
    "
17 PRINT "
   "
18 PRINT "
  "
19 PRINT "
  "
2Ø PRINT "
  "
21 PRINT "           DAS VERZAUBERTE
  "
22 PRINT "               SCHLOSS
  "
23 PRINT "
  "
24 PRINT "            EIN ADVENTURE
  "
25 PRINT "                 VON
  "
26 PRINT "           JOERG WALKOWIAK
  "
27 PRINT "
 ";
8Ø FOR I=1 TO 3ØØØ
9Ø NEXT I
1ØØ REM ********* DATEN DES ADVENTURES
11Ø AR=26:AO=45:AF=3
12Ø SPIELER=9
14Ø TRAP 49ØØ
15Ø DIM RAUM$(8Ø),DURCHGANG(AR,6),RICH
TUNG$(36),EINGABE$(2Ø),T$(4Ø),OB(AO),L
EERZEILE$(38),DZ$(5Ø),ZENDE$(3),FL(5)
16Ø DIM VERBEN$(9Ø),EVERB$(2Ø),EOBJEKT
```

```
$(2Ø),OBJEKTE$(22Ø),MØ$(6Ø),M1$(3Ø),M2
$(3Ø),M3$(3Ø),M4$(38),M5$(38)
17Ø VERBEN$="UNTERSUCHENIMM      LIES
     OEFFNE    ZERSTOERE BENUTZE   FUE
LLE    LEG       WECKE     "
18Ø OBJEKTE$="REGATISCARI TOPFFLASBRUN
BUCHWAECSEILBEWOGAESZETTKUECMESSARMBSC
HLTOR ZUGB         BAEU    REGA    BAEU
"
181 OBJEKTE$(LEN(OBJEKTE$)+1)="
 ELSTHOEHMETARUES    SUMPSCHLEIERBODEW
ASSGRAB    SPINFLAS    GRUBSCHLALTA"
19Ø GOTO 7ØØ
2ØØ REM ********* RAUMBESCHREIBUNGEN
2Ø1 PRINT "IN DER SCHLOSSBIBLIOTHEK.":
RETURN
2Ø2 PRINT "IM STUDIERZIMMER.":RETURN
2Ø3 PRINT "IN DER KUECHE.":RETURN
2Ø4 PRINT "IM SCHLOSSHOF.":RETURN
2Ø5 PRINT "IN EINEM DER WACHTTUERME.":
RETURN
2Ø6 PRINT "IM SCHLOSSHOF.":RETURN
2Ø7 PRINT "VOR DER ZUGBRUECKE.":RETURN

2Ø8 PRINT "IM GROSSEN FESTSAAL.":RETUR
N
2Ø9 PRINT "IN EINEM PRIVATGEMACH.":RET
URN
21Ø PRINT "AUF EINER ZUGBRUECKE.":RETU
RN
211 PRINT "IM SCHLOSSHOF.":RETURN
212 PRINT "AUF EINEM WALDWEG.":RETURN
213 PRINT "IM ZAUBERWALD.":RETURN
214 PRINT "IN EINER DUESTEREN HOEHLE."
:RETURN
215 PRINT "IM WALD.":RETURN
216 PRINT "IM ZAUBERWALD.":RETURN
217 PRINT "IM ZAUBERWALD.":RETURN
218 PRINT "AM FUSSE EINES GEBIRGES.":R
ETURN
219 PRINT "IN EINER DUESTEREN HOEHLE."
```

```
:RETURN
22Ø PRINT "IN EINEM SCHLOSS.":RETURN
221 PRINT "IM WALD.":RETURN
222 PRINT "IM SUMPF.":RETURN
223 PRINT "IM SUMPF.":RETURN
224 PRINT "AN EINER QUELLE.":RETURN
225 PRINT "IN EINER DUESTEREN HOEHLE."
:RETURN
226 PRINT "IN DER SCHLANGENGRUBE.":RET
URN
3ØØ REM ***************** GEGENSTAENDE
3Ø1 T$="UNZAEHLIGE BUECHERREGALE":RETU
RN
3Ø2 T$="EINEN TISCH":RETURN
3Ø3 T$="MEINEN LEHRER ARI AT":RETURN
3Ø4 T$="EINEN GROSSEN TOPF":RETURN
3Ø5 T$="EINE FLASCHE":RETURN
3Ø6 T$="DEN SCHLOSSBRUNNEN":RETURN
3Ø7 T$="EIN BUCH":RETURN
3Ø8 T$="SCHLAFENDE WAECHTER":RETURN
3Ø9 T$="DEN SEILZUG DER ZUGBRUECKE":RE
TURN
31Ø T$="BEWOHNER DES SCHLOSSES":RETURN

311 T$="MEINE GAESTE":RETURN
312 T$="EINEN ZETTEL":RETURN
313 T$="EINEN KUECHENTISCH":RETURN
314 T$="EIN SCHLACHTERMESSER":RETURN
315 T$="EINE ARMBRUST":RETURN
316 T$="EINEN SCHLUESSEL":RETURN
317 T$="EIN SCHWERES EISENTOR":RETURN
318 T$="EINE ZUGBRUECKE":RETURN
319 T$="NICHTS BESONDERES":RETURN
32Ø T$="NICHTS BESONDERES":RETURN
321 T$="BAEUME":RETURN
322 T$="BAEUME":RETURN
323 T$="MEHRERE VOLLGEPACKTE REGALE":R
ETURN
324 T$="NICHTS BESONDERES":RETURN
325 T$="BAEUME":RETURN
326 T$="BAEUME":RETURN
```

```
327 T$="BAEUME":RETURN
328 T$="EINE GROSSE ELSTER":RETURN
329 T$="EINEN HOEHLENEINGANG":RETURN
33Ø T$="GOLDEN GLAENZENDES METALL":RET
URN
331 T$="EINE ALTE RUESTUNG":RETURN
332 T$="NICHTS BESONDERES":RETURN
333 T$="EINEN SUMPF":RETURN
334 T$="SCHLAMM":RETURN
335 T$="SCHLANGENEIER":RETURN
336 T$="DEN TRUEGERISCHEN BODEN":RETUR
N
337 T$="SPRUDELNDES WASSER":RETURN
338 T$="EINEN SCHLOSSGRABEN":RETURN
339 T$="NICHTS BESONDERES":RETURN
34Ø T$="SPINNWEBEN":RETURN
341 T$="EINE FLASCHE MIT BLAUEM WASSER
":RETURN
342 T$="NICHTS BESONDERES":RETURN
343 T$="EINE SCHLANGENGRUBE":RETURN
344 T$="SCHLANGEN":RETURN
345 T$="EINEN STEINERNEN ALTAR":RETURN

4ØØ REM **** POSITIONEN OBJEKTE ***
41Ø DATA 1,1,2,3,Ø,4,Ø,5,5,8,8,Ø,3,Ø,5
,Ø,6,7,9,11,12,13,14,25,15,16,17,17,18
,19,Ø,21,Ø,23,26,23,24,1Ø,2Ø,Ø,Ø,19,22
42Ø DATA 26,25
5ØØ REM RICHTUNGSTABELLE
5Ø1 DATA Ø,3,Ø,2,Ø,Ø
5Ø2 DATA Ø,Ø,1,Ø,Ø,Ø
5Ø3 DATA 1,8,Ø,4,Ø,Ø
5Ø4 DATA Ø,8,3,6,Ø,Ø
5Ø5 DATA Ø,Ø,Ø,Ø,Ø,11
5Ø6 DATA 11,Ø,4,Ø,Ø,Ø
5Ø7 DATA Ø,Ø,6,Ø,Ø,Ø
5Ø8 DATA 4,Ø,3,9,Ø,Ø
5Ø9 DATA Ø,Ø,8,Ø,Ø,Ø
51Ø DATA Ø,Ø,7,12,Ø,Ø
511 DATA Ø,6,Ø,Ø,5,Ø
512 DATA 12,16,1Ø,13,Ø,Ø
```

```
513 DATA 21,17,12,15,Ø,Ø
514 DATA Ø,18,14,Ø,Ø,Ø
515 DATA 15,15,15,16,Ø,Ø
516 DATA 12,21,15,17,Ø,Ø
517 DATA 13,22,16,18,Ø,Ø
518 DATA 14,Ø,17,19,Ø,Ø
519 DATA 25,Ø,18,19,Ø,Ø
52Ø DATA 2Ø,2Ø,2Ø,21,Ø,Ø
521 DATA 16,21,2Ø,21,Ø,Ø
522 DATA 17,Ø,Ø,23,Ø,Ø
523 DATA Ø,Ø,22,24,Ø,Ø
524 DATA 23,24,Ø,23,Ø,Ø
525 DATA 25,19,25,14,14,Ø
526 DATA Ø,Ø,Ø,Ø,22,Ø
6ØØ PRINT "ICH VERSTEHE NICHT, WAS DU
MEINST !":RETURN
6Ø1 PRINT "ICH SEHE NICHTS BESONDERES.
":RETURN
6Ø2 PRINT "SO STARK BIN ICH NICHT.":RE
TURN
6Ø3 PRINT "SIE SCHEINEN TIEF ZU SCHLAF
EN.":RETURN
6Ø4 PRINT "DAS HALTE ICH FUER WENIG SI
NNVOLL.":RETURN
6Ø6 PRINT "DIE ELSTER HAT MIR ETWAS GE
STOHLEN !":RETURN
6Ø7 PRINT "DA LIEGT EINE ALTE RUESTUNG
.":RETURN
6Ø8 PRINT "IN DER HAND HAELT ER EINEN
ZETTEL.":RETURN
6Ø9 PRINT "IN EINER TASCHE IST EIN SCH
LUESSEL.":RETURN
61Ø PRINT "EINE INSCHRIFT BESAGT:":PRI
NT "NIMM UNSER OFPER UND SEI UNS GNAED
IG.":RETURN
699 REM ********* 2. TITEL: EINLEITUNG
7ØØ PRINT " ":PRINT :PRINT " WUENSCHEN
 SIE RATSCHLAEGE FUER IHR
WEITERES VORGEHEN ";:INPUT EINGABE$
798 IF EINGABE$(1,1)="J" THEN GOSUB 8Ø
Ø
```

```
799 GOTO 9ØØ
8ØØ GRAPHICS Ø:SETCOLOR 2,1,2:SETCOLOR
 4,1,Ø
82Ø PRINT "           ATARI - VENTURES
          ":PRINT "     (C) 1984 BY JO
ERG WALKOWIAK     ":REM INVERS
83Ø PRINT :PRINT "STellen Sie sich ein
en Roboter vor,   den Sie mit zahlreic
hen Kommandos"
84Ø PRINT "steuern koennen. Ich bin di
eser Ro-   boter, und ich werde mich f
uer Sie"
85Ø PRINT "den Gefahren der verwegenst
en Aben-   teuer  aussetzen."
855 PRINT "Damit Sie mich sinnvoll agi
eren las-  en koennen, werde ich Ihnen
 die":PRINT "Situation in der ich mich
 gerade"
86Ø PRINT "befinde, jeweils genau besc
hreiben.":PRINT "Anschliessend sagen S
ie mir mit zwei"
863 PRINT "Worten wie zum Beispiel UNT
ERSUCHE "
87Ø PRINT "TUER, NIMM MESSER, was ich
tun soll.":PRINT :PRINT "Darueber hina
us verstehe ich die Be-  fehle:";
88Ø PRINT "   INVENTUR und HELP,"
885 PRINT "             SAVE, LOAD,"
887 PRINT "             sowie ENDE.":PR
INT
89Ø PRINT "BITTE DRUECKEN SIE <RETURN>
 UND ...";:INPUT EINGABE$:RETURN :REM
INVERS
9ØØ FOR I=1 TO AO
91Ø READ LAGEORT:OB(I)=LAGEORT
92Ø NEXT I
93Ø FOR I=1 TO AF:FL(I)=Ø:NEXT I
94Ø FOR RAUM=1 TO AR
95Ø FOR RICHTUNG=1 TO 6
96Ø READ ZIEL:DURCHGANG(RAUM,RICHTUNG)
=ZIEL
```

```
97Ø NEXT RICHTUNG
98Ø NEXT RAUM
1ØØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:GOTO 1Ø2Ø
1Ø1Ø REM ******** KLARTEXT: RICHTUNGEN
1Ø11 T$="NORDEN, ":RETURN
1Ø12 T$="SUEDEN, ":RETURN
1Ø13 T$="WESTEN, ":RETURN
1Ø14 T$="OSTEN,  ":RETURN
1Ø15 T$="OBEN, ":RETURN
1Ø16 T$="UNTEN, ":RETURN
1Ø2Ø LEERZEILE$="
                "
1Ø3Ø T$=""
1Ø4Ø ZENDE$=CHR$(3Ø):ZENDE$(LEN(ZENDE$
)+1)=CHR$(3Ø):ZENDE$(LEN(ZENDE$)+1)=".
"
1Ø8Ø PRINT
1Ø9Ø FOR ZEILE=Ø TO 12:POSITION 2,ZEIL
E:PRINT LEERZEILE$;:NEXT ZEILE
11ØØ POSITION 2,Ø:PRINT "ICH BIN ";:GO
SUB 2ØØ+SPIELER
111Ø DZ$="ICH SEHE "
112Ø FOR I=1 TO AO
113Ø IF OB(I)<>SPIELER THEN GOTO 117Ø
114Ø GEDRUCKT=-1:GOSUB 3ØØ+I:T$
(LEN(T$)+1)=", "
115Ø IF LEN(DZ$)+LEN(T$)>=38 THEN PRIN
T DZ$:DZ$=T$(1,LEN(T$)):T$="":GOTO 117
Ø
116Ø DZ$(LEN(DZ$)+1)=T$:T$=""
117Ø NEXT I
118Ø IF  NOT GEDRUCKT THEN DZ$
(LEN(DZ$)+1)="NICHTS BESONDERES. "
119Ø DZ$(LEN(DZ$)+1)=ZENDE$
12ØØ PRINT DZ$
121Ø PRINT LEERZEILE$
122Ø DZ$="ICH KANN NACH ":GEDRUCKT=Ø
123Ø FOR RICHTUNG=1 TO 6
124Ø IF DURCHGANG(SPIELER,RICHTUNG)=Ø
THEN GOTO 128Ø
```

```
125Ø GOSUB 1Ø1Ø+RICHTUNG:GEDRUCKT=-1
126Ø IF LEN(DZ$)+LEN(T$)>=38 THEN PRIN
T DZ$:DZ$=T$(1,LEN(T$)):T$="":GOTO 128
Ø
127Ø DZ$(LEN(DZ$)+1)=T$:T$=""
128Ø NEXT RICHTUNG
129Ø IF  NOT GEDRUCKT THEN DZ$(LEN(DZ$
)+1)="NIRGENDWO. "
13ØØ DZ$(LEN(DZ$)+1)=ZENDE$
131Ø PRINT DZ$
132Ø PRINT "--------------------------------
-------":REM 37X CONTROL+R
134Ø IF SPIELER=25 THEN GOSUB 14ØØØ
139Ø POSITION 2,23:PRINT "WAS SOLL ICH
TUN ";:INPUT EINGABE$
1392 IF SPIELER=17 THEN GOSUB 15ØØØ
1394 IF SPIELER=23 THEN GOSUB 151ØØ
1396 IF SPIELER=15 THEN GOSUB 152ØØ
14ØØ IF LEN(EINGABE$)>2 THEN 148Ø
141Ø IF EINGABE$="N" AND DURCHGANG(SPI
ELER,1)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,1):PRINT "O.K.":GOTO 1Ø8Ø
142Ø IF EINGABE$="S" AND DURCHGANG(SPI
ELER,2)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,2):PRINT "O.K.":GOTO 1Ø8Ø
143Ø IF EINGABE$="W" AND DURCHGANG(SPI
ELER,3)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,3):PRINT "O.K.":GOTO 1Ø8Ø
144Ø IF EINGABE$="O" AND DURCHGANG(SPI
ELER,4)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,4):PRINT "O.K.":GOTO 1Ø8Ø
145Ø IF EINGABE$="OB" AND DURCHGANG(SP
IELER,5)<>Ø THEN SPIELER=DURCHGANG(SPI
ELER,5):PRINT "O.K.":GOTO 1Ø8Ø
146Ø IF EINGABE$="U" AND DURCHGANG(SPI
ELER,6)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,6):PRINT "O.K.":GOTO 1Ø8Ø
147Ø IF LEN(EINGABE$)<3 THEN PRINT "DA
HIN FUEHRT KEIN WEG !":GOTO 1Ø8Ø
148Ø IF LEN(EINGABE$)>8 THEN GOTO 2ØØØ
1499 REM ************** START INVENTUR
```

```
15ØØ IF EINGABE$(1,3)<>"INV" THEN GOTO
 16ØØ
151Ø PRINT "ICH TRADE FOLGENDES MIT MI
R:"
152Ø FOR I=1 TO AO
153Ø IF OB(I)=-1 THEN GOSUB 3ØØ+I:PRIN
T T$
154Ø NEXT I
155Ø GOTO 1Ø8Ø
156Ø REM ************** ENDE INVENTUR
1599 REM ****************** SAVE GAME
16ØØ IF EINGABE$(1,3)<>"SAV" THEN GOTO
 17ØØ
16Ø5 PRINT "UNTER WELCHEM NAMEN ";:INP
UT EINGABE$:IF LEN(EINGABE$)>8 THEN PR
INT "BITTE ETWAS KUERZER !":GOTO 16Ø5
161Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
=".DAT"
1615 T$(LEN(T$)+1)=EINGABE$
162Ø OPEN #1,8,Ø,T$
1625 PRINT #1,SPIELER
163Ø FOR I=1 TO AO
1631 PRINT #1,OB(I)
1632 NEXT I
1635 FOR RAUM=1 TO AR
1636 FOR RICHTUNG=1 TO 6
1637 PRINT #1,DURCHGANG(RAUM,RICHTUNG)
1638 NEXT RICHTUNG
1639 NEXT RAUM
1645 FOR I=1 TO AF
1646 PRINT #1,FL(I)
1647 NEXT I
165Ø CLOSE #1
167Ø PRINT "O.K.":GOTO 1Ø8Ø
1699 REM ****************** LOAD GAME
17ØØ IF EINGABE$(1,3)<>"LOA" THEN GOTO
18ØØ
17Ø5 PRINT "WELCHES SPIEL ";:INPUT EIN
GABE$:IF LEN(EINGABE$)>8 THEN PRINT "D
AS KANN ES NICHT GEBEN !":GOTO 17Ø5
171Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
```

```
=".DAT"
1715 T$(LEN(T$)+1)=EINGABE$
172Ø OPEN #1,4,Ø,T$
1725 INPUT #1,SPIELER
173Ø FOR I=1 TO AO
1731 INPUT #1,LAGEORT:OB(I)=LAGEORT
1732 NEXT I
1735 FOR RAUM=1 TO AR
1736 FOR RICHTUNG=1 TO 6
1737 INPUT #1,ZIEL:DURCHGANG(RAUM,RICH
TUNG)=ZIEL
1738 NEXT RICHTUNG
1739 NEXT RAUM
1745 FOR I=1 TO AF
1746 INPUT #1,FLAG:FL(I)=FLAG
1747 NEXT I
175Ø CLOSE #1
177Ø PRINT "O.K.":GOTO 1Ø8Ø
18ØØ IF EINGABE$(1,3)<>"HEL" THEN GOTO
 19ØØ
181Ø GOSUB 188Ø
182Ø GEDRUCKT=Ø
183Ø FOR I=1 TO LEN(VERBEN$) STEP 1Ø
184Ø PRINT VERBEN$(I,I+9):GEDRUCKT=GED
RUCKT+1
185Ø IF GEDRUCKT=18 THEN GOSUB 189Ø:GE
DRUCKT=1:GOSUB 188Ø
186Ø NEXT I
187Ø GOSUB 189Ø:GRAPHICS Ø:SETCOLOR 2,
1,Ø:SETCOLOR 4,1,Ø:GOTO 1Ø8Ø
188Ø GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:PRINT "VERSUCHE FOLGENDE HANDL
UNGEN ... ":PRINT :RETURN
189Ø PRINT :PRINT :PRINT "<RETURN> DRU
ECKEN ... ";:INPUT EINGABE$:RETURN
19ØØ IF EINGABE$(1,3)<>"INS" THEN GOTO
 195Ø
191Ø GOSUB 8ØØ
192Ø GOTO 1Ø8Ø
195Ø IF EINGABE$(1,3)<>"END" THEN GOTO
 2ØØØ
```

```
196Ø GRAPHICS Ø:PRINT "DER AUTOR WUENS
CHT IHNEN MEHR GLUECK":PRINT "FUERS NA
ECHSTE MAL !":PRINT :PRINT :PRINT :END

1999 REM ************* EINGABE TRENNEN
2ØØØ LN=LEN(EINGABE$)
2Ø1Ø FOR I=1 TO LN
2Ø2Ø IF EINGABE$(I,I)<>" " THEN NEXT I
2Ø3Ø EVERB$=EINGABE$(1,I)
2Ø4Ø IF LEN(EVERB$)=LN-1 THEN GOTO 2Ø9
Ø
2Ø5Ø EOBJEKT$=EINGABE$(I+1,LN)
2Ø6Ø VN=Ø:N=Ø
2Ø7Ø EVERB$=EVERB$(1,3)
2Ø8Ø EOBJEKT$=EOBJEKT$(1,3)
2Ø9Ø FOR I=1 TO LEN(VERBEN$) STEP 1Ø
21ØØ VN=VN+1
211Ø IF VERBEN$(I,I+2)=EVERB$ THEN 214
Ø
212Ø NEXT I
213Ø PRINT "ICH VERSTEHE DAS VERB NICH
T !":GOTO 1Ø8Ø
214Ø FOR I=1 TO LEN(OBJEKTE$) STEP 4
215Ø N=N+1
216Ø IF OBJEKTE$(I,I+2)=EOBJEKT$ THEN
GOTO 22ØØ
217Ø NEXT I
218Ø PRINT "ICH VERSTEHE DAS OBJEKT NI
CHT.":GOTO 1Ø8Ø
22ØØ ON VN GOTO 5ØØØ,6ØØØ,7ØØØ,8ØØØ,9Ø
ØØ,1ØØØØ,11ØØØ,12ØØØ,16ØØØ
45ØØ GRAPHICS Ø:REM ****** SPIELER TOT
46ØØ PRINT "AUCH DAS NOCH !":PRINT :PR
INT MØ$
461Ø PRINT :PRINT "ICH BIN TOT !":PRIN
T
462Ø PRINT "SOLL ICH ES NOCH EINMAL VE
RSUCHEN ";:INPUT EINGABE$
463Ø IF EINGABE$(1,1)="J" THEN CLR :GO
TO 1ØØ
464Ø GOTO 196Ø
```

```
48ØØ GRAPHICS Ø
481Ø PRINT "HERZLICHEN GLUECKWUNSCH !"
482Ø PRINT :PRINT :PRINT "SIE HABEN DI
E IHNEN GESTELLTE AUFGABE":PRINT "GELO
EST, UND DUERFEN SICH AN EINEM"
483Ø PRINT "ANDEREN ADVENTURE VERSUCHE
N."
484Ø PRINT :PRINT :PRINT :END
49ØØ PRINT "ACHTUNG FEHLER !":GOTO 1Ø8
Ø
4999 REM ***** SPIELERZUEGE AUSFUEHREN
5ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN GOTO 59ØØ
5ØØ1 IF N=16 AND SPIELER=2Ø AND OB(31)
<>-1 AND OB(4Ø)<>-1 THEN GOSUB 6Ø7:OB(
31)=2Ø:OB(4Ø)=2Ø:GOTO 1Ø8Ø
5ØØ2 IF N=1Ø AND SPIELER=8 THEN GOSUB
6Ø3:GOTO 1Ø8Ø
5ØØ3 IF N=11 AND SPIELER=7 THEN GOSUB
6Ø3:GOTO 1Ø8Ø
5ØØ4 IF N=4 AND (SPIELER=3 OR OB(4)=-1
) THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5ØØ5 IF N=5 AND (SPIELER=3 OR OB(N)=-1
) THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5ØØ6 IF N=1 AND SPIELER=1 THEN GOSUB 6
Ø1:GOTO 1Ø8Ø
5ØØ8 IF N=3 AND SPIELER=2 AND OB(12)=Ø
 THEN GOSUB 6Ø8:OB(12)=2:GOTO 1Ø8Ø
5ØØ9 IF N=3 AND SPIELER=2 THEN GOSUB 6
Ø1:GOTO 1Ø8Ø
5Ø1Ø IF N=6 AND OB(5)=Ø THEN PRINT "AU
F DEM WASSER TREIBT EINE FLASCHE.":OB(
5)=SPIELER:GOTO 1Ø8Ø
5Ø11 IF N=7 AND (SPIELER=1 OR OB(N)=-1
) THEN PRINT "ES IST EIN BUCH UEBER ZA
UBERTRAENKE.":GOTO 1Ø8Ø
5Ø12 IF N=8 AND SPIELER=5 AND OB(16)=Ø
 THEN GOSUB 6Ø9:OB(16)=SPIELER:GOTO 1Ø
8Ø
5Ø13 IF N=9 AND  NOT FL(1) THEN GOSUB
6Ø1:GOTO 1Ø8Ø
```

```
5Ø14 IF N=9 AND FL(1) AND SPIELER=5 TH
EN PRINT "ER IST NICHT FUNKTIONSFAEHIG
.":GOTO 1Ø8Ø
5Ø15 IF N=12 AND OB(12)=-1 THEN PRINT
"DARAUF STEHT ETWAS GESCHRIEBEN.":OB(1
6)=SPIELER:GOTO 1Ø8Ø
5Ø16 IF N=12 AND OB(12)=2 THEN PRINT "
DAZU muß ICH IHN ERST NEHNEN.":GOTO 1Ø
8Ø
5Ø17 IF N=13 AND OB(14)=Ø THEN PRINT "
AUF DEM TISCH LIEGT EIN MESSER.":OB(14
)=3:GOTO 1Ø8Ø
5Ø18 IF N=2 AND (OB(7)=1 OR OB(7)=Ø) T
HEN PRINT "DARAUF LIEGT EIN ALTES BUCH
.":OB(7)=1:GOTO 1Ø8Ø
5Ø19 IF N=14 AND (OB(14)=SPIELER OR OB
(14)=-1) THEN PRINT "DIE KLINGE IST SE
HR SCHARF.":GOTO 1Ø8Ø
5Ø2Ø IF N=15 AND (OB(N)=SPIELER OR OB(
N)=-1) THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø21 IF N=16 AND (OB(16)=-1 OR OB(16)=
SPIELER) THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø22 IF N=17 AND SPIELER=6 AND FL(3)<>
-1 THEN PRINT "ES IST VERSCHLOSSEN.":G
OTO 1Ø8Ø
5Ø23 IF N=17 AND SPIELER=6 AND FL(3)=-
1 THEN PRINT "ES STEHT WEIT OFFEN.":GO
TO 1Ø8Ø
5Ø24 IF N=18 AND SPIELER=7 AND FL(1) T
HEN PRINT "DIE ZUGBRUECKE FAELLT NACH
UNTEN.":DURCHGANG(7,4)=1Ø:GOTO 1Ø8Ø
5Ø25 IF N=18 AND SPIELER=7 AND  NOT FL
(1) THEN PRINT "DIE ZUGBRUECKE IST HOC
HGEZOGEN.":GOTO 1Ø8Ø
5Ø26 IF N=21 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø27 IF N=23 AND SPIELER=14 THEN MØ$="
ES IST EINE ZAUBERKUECHE. DER ZAUBERER
KOMMT, MACHT EINEN STEIN AUS MIR.":GOT
O 45ØØ
5Ø28 IF N=28 AND SPIELER=17 THEN PRINT
 "ES IST WIRKLICH EIN PRACHTEXEMPLAR."
```

```
:GOSUB 13ØØ:GOTO 1Ø8Ø
5Ø29 IF N=29 AND SPIELER=18 THEN PRINT
 "EIN MERKWUERDIGER GERUCH DRINGT":PRI
NT "HERAUS.":GOTO 1Ø8Ø
5Ø3Ø IF N=3Ø AND (SPIELER=19 OR OB(3Ø)
=-1) THEN PRINT "ES IST WERTLOSES META
LL.":GOTO 1Ø8Ø
5Ø31 IF N=31 AND (SPIELER=2Ø OR OB(31)
=-1) THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø32 IF N=35 AND (SPIELER=26 OR OB(35)
=-1) THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø33 IF N=36 AND SPIELER=23 THEN PRINT
"ER SIEHT NICHT SEHR TRAGFEST AUS.":GO
TO 1Ø8Ø
5Ø34 IF N=37 AND SPIELER=24 THEN PRINT
"ES IST DAS BERUEHMTE BLAUE WASSER.":G
OTO 1Ø8Ø
5Ø35 IF N=38 AND SPIELER=1Ø THEN GOSUB
6Ø1:GOTO 1Ø8Ø
5Ø36 IF N=4Ø AND (SPIELER=2Ø OR OB(4Ø)
=-1) THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø37 IF N=43 THEN SPIELER=26:PRINT "O.
K.":GOTO 1Ø8Ø
5Ø38 IF N=45 THEN GOSUB 61Ø:GOTO 1Ø8Ø
5899 GOSUB 6Ø1:GOTO 1Ø8Ø
59ØØ IF N=21 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
59Ø1 IF N=16 AND SPIELER=2Ø AND OB(31)
<>-1 AND OB(4Ø)<>-1 THEN GOSUB 6Ø7:OB(
31)=2Ø:OB(4Ø)=2Ø:GOTO 1Ø8Ø
59Ø2 IF N=33 AND SPIELER=23 THEN GOSUB
131ØØ:PRINT "IM OSTEN SCHEINT EINE QUE
LLE ZU SEIN.":GOTO 1Ø8Ø
59Ø3 IF N=16 AND (SPIELER=23 OR OB(34)
=-1) THEN PRINT "ES IST HEILSCHLAMM. "
:GOSUB 131ØØ:GOTO 1Ø8Ø
59Ø4 IF N=2 AND SPIELER=3 AND OB(14)=Ø
THEN PRINT "AUF DEM TISCH LIEGT EIN ME
SSER.":OB(14)=3:GOTO 1Ø8Ø
59Ø5 IF N=2 AND SPIELER=3 AND OB(14)<>
Ø THEN GOSUB 6Ø1:GOTO 1Ø8Ø
599Ø PRINT "SO ETWAS SEHE ICH HIER NIC
```

```
HT !":GOTO 1Ø8Ø
6ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN GOTO 69ØØ
6ØØ1 IF N=1 THEN GOSUB 6Ø2:GOTO 1Ø8Ø
6ØØ2 IF N=6 THEN GOSUB 6Ø2:GOTO 1Ø8Ø
6ØØ3 IF N=17 THEN GOSUB 6Ø2:GOTO 1Ø8Ø
6ØØ4 IF N=2 THEN GOSUB 6Ø4:GOTO 1Ø8Ø
6ØØ5 IF N=3 THEN GOSUB 6Ø4:GOTO 1Ø8Ø
6ØØ6 IF N=8 THEN GOSUB 6Ø4:GOTO 1Ø8Ø
6ØØ7 IF N=1Ø THEN GOSUB 6Ø4:GOTO 1Ø8Ø
6ØØ8 IF N=11 THEN GOSUB 6Ø4:GOTO 1Ø8Ø
6ØØ9 ZA=Ø:FOR I=1 TO AO:IF OB(I)=-1 TH
EN ZA=ZA+1:IF ZA=4 THEN GOTO 6999
6Ø1Ø NEXT I
6Ø11 IF N=5 THEN OB(N)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
6Ø12 IF N=7 THEN OB(N)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
6Ø13 IF N=12 THEN OB(N)=-1:PRINT "O.K.
":GOTO 1Ø8Ø
6Ø14 IF N=15 THEN OB(N)=-1:PRINT "O.K.
":GOTO 1Ø8Ø
6Ø15 IF N=16 THEN OB(N)=-1:PRINT "O.K.
":GOTO 1Ø8Ø
6Ø16 IF N=14 THEN OB(N)=-1:PRINT "O.K.
":GOTO 1Ø8Ø
6Ø17 IF N=9 THEN GOSUB 6Ø2:GOTO 1Ø8Ø
6Ø18 IF N=23 THEN GOSUB 6Ø2:GOTO 1Ø8Ø
6Ø19 IF N=28 THEN PRINT "WOMIT SOLL IC
H SIE FANGEN ?":GOSUB 13ØØØ:GOTO 1Ø8Ø
6Ø2Ø IF N=18 THEN GOSUB 6ØØ:GOTO 1Ø8Ø
6Ø21 IF N=3Ø THEN OB(N)=-1:PRINT "O.K.
":GOTO 1Ø8Ø
6Ø22 IF N=31 THEN OB(31)=-1:PRINT "ICH
 HABE DIE RUESTUNG ANGELEGT.":GOTO 1Ø8
Ø
6Ø23 IF N=35 AND OB(31)=-1 THEN PRINT
"O.K.":OB(35)=-1:GOTO 1Ø8Ø
6Ø24 IF N=35 AND OB(31)<>-1 THEN MØ$="
EINE SCHLANGE HAT MICH GEBISSEN.":GOTO
45ØØ
```

```
6Ø25 IF N=36 THEN GOSUB 6Ø4:GOTO 1Ø8Ø
6Ø26 IF N=37 AND OB(5)=-1 THEN PRINT "
O.K.":OB(5)=Ø:OB(41)=-1:GOTO 1Ø8Ø
6Ø27 IF N=37 AND OB(5)<>-1 THEN PRINT
"ZUVOR BRAUCHE ICH EINE LEERE FLASCHE.
":GOTO 1Ø8Ø
6Ø28 IF N=38 THEN GOSUB 6ØØ:GOTO 1Ø8Ø
6Ø29 IF N=4Ø THEN OB(4Ø)=-1:PRINT "O.K
.":GOTO 1Ø8Ø
6Ø32 IF N=13 THEN GOSUB 6Ø4:GOTO 1Ø8Ø
6Ø33 IF N=4 THEN OB(4)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
69ØØ IF N=16 AND SPIELER=23 AND OB(4)<
>-1 THEN PRINT "ZUVOR BRAUCHE ICH EIN
GEFAESS !":GOTO 1Ø8Ø
69Ø1 IF N=16 AND SPIELER=23 AND OB(4)=
-1 THEN PRINT "O.K.":OB(34)=-1:GOTO 1Ø
8Ø
69Ø2 IF N=2 AND SPIELER=3 THEN GOSUB 6
Ø4:GOTO 1Ø8Ø
6998 PRINT "SO ETWAS SEHE ICH HIER NIC
HT !":GOTO 1Ø8Ø
6999 PRINT "TUT MIR LEID - ABER ICH TR
AGE SCHON":PRINT "GENUG !":GOTO 1Ø8Ø
7ØØØ IF N=13 AND OB(12)<>-1 THEN PRINT
"ICH HABE KEINEN ZETTEL.":GOTO 1Ø8Ø
7ØØ1 IF N=12 AND OB(12)=-1 THEN CO=INT
(RND(1)*1ØØ):PRINT "ICH SEHE, ZITTRIG
GESCHRIEBEN: ";CO:GOTO 1Ø8Ø
7ØØ2 IF N=7 AND OB(7)=-1 THEN GOTO 765
Ø
7ØØ3 IF N=7 AND OB(7)<>-1 THEN PRINT "
DAZU muß ICH IHN ERST HABEN.":GOTO 1Ø8
Ø
7649 GOSUB 6ØØ:GOTO 1Ø8Ø
765Ø PRINT "WELCHE SEITE ";:INPUT SE
7651 IF SE<>CO THEN GOSUB 6Ø1:GOTO 1Ø8
Ø
7652 GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:PRINT " RATSCHLAEGE FUER BESON
DERE NOTLAGEN":PRINT
```

```
7654 PRINT "SOLLTE ES DIR ANGESCHEHEN,
 daß DIE":PRINT "DEINEN VON EINEM ZAUB
ERER BOESER"
7655 PRINT "GESINNUNG IN DEN LANGEN SC
HLAF VER-":PRINT "SETZT WERDEN, SO HOE
RE MEINEN RAT:":PRINT
7657 PRINT :PRINT "VERSCHAFFE DIR DIE
VIER ZUTATEN ZU":PRINT "HUMBUGS HEILHA
SSER UND BRINGE SIE"
7659 PRINT "DEM ZAUBERER JENSEITS DES
ZAUBER-":PRINT "WALDES."
766Ø PRINT "SOLLTE ER DIR wOHL GESONNE
N SEIN,":PRINT "SO WIRD ER DIR HELFEN.
"
7661 PRINT "IST ER ES NICHT, SO KANNST
 DU NUR":PRINT "NOCH AUF EINEN PRINZEN
 HOFFEN, "
7663 PRINT "WELCHER AUF TRADITIONELLE
WEISE":PRINT "DEINE TOCHTER, UND MIT I
HR ALLE"
7665 PRINT "BEWOHNER DES SCHLOSSES, WA
CHKUESST !"
7666 OPEN #1,4,Ø,"K:"
7667 GET #1,A
7668 CLOSE #1
7669 GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:GOTO 1Ø8Ø
79ØØ OB(N)=SPIELER:PRINT "O.K.":GOTO 1
Ø8Ø
8ØØØ IF N=17 AND OB(16)<>-1 THEN PRINT
 "WOMIT ?":GOTO 1Ø8Ø
8ØØ1 IF N=17 AND OB(16)=-1 THEN PRINT
"O.K.":DURCHGANG(6,4)=7:FL(3)=-1:GOTO
1Ø8Ø
8ØØ2 IF N=35 AND OB(35)=-1 THEN PRINT
"DARIN WAR EINE ZAEHE FLUESSIGKEIT.":G
OTO 1Ø8Ø
8999 PRINT "ICH VERSTEHE NICHT, WAS DU
MEINST.":GOTO 1Ø8Ø
9ØØØ IF N=9 AND OB(14)=-1 AND SPIELER=
5 THEN PRINT "DIE HALTESEILE SIND ZERS
```

```
CHNITTEN.":FL(1)=-1:GOTO 1Ø8Ø
9ØØ1 IF N=9 AND OB(14)<>-1 AND SPIELER
=5 THEN PRINT "WOMIT ?":GOTO 1Ø8Ø
9999 GOSUB 6ØØ:GOTO 1Ø8Ø
1ØØØØ IF OB(N)<>SPIELER AND OB(N)<>-1
THEN PRINT "SO ETWAS SEHE ICH HIER NIC
HT.":GOTO 1Ø8Ø
1ØØØ1 IF N=9 THEN PRINT "DAZU SIND MIN
DESTENS ZWEI MANN":PRINT "NOETIG.":GOT
O 1Ø8Ø
1ØØØ2 IF N=14 AND SPIELER=5 THEN PRINT
 "DIE HALTESEILE SIND ZERSCHNITTEN.":F
L(1)=-1:GOTO 1Ø8Ø
1ØØØ3 IF N=31 THEN PRINT "O.K. - ICH B
IN IN DER RUESTUNG.":OB(31)=-1:GOTO 1Ø
8Ø
1Ø99Ø GOSUB 6ØØ:GOTO 1Ø8Ø
11ØØØ IF N=5 AND SPIELER=24 AND OB<5>=
-1 THEN PRINT "O.K.":OB(5)=Ø:OB(41)=-
1:GOTO 1Ø8Ø
11ØØ1 IF N=5 AND SPIELER<>24 AND OB(5)
=-1 THEN PRINT "WOMIT ?":GOTO 1Ø8Ø
1199Ø GOSUB 6ØØ:GOTO 1Ø8Ø
12ØØØ IF OB(N)<>-1 THEN GOTO 129ØØ
12ØØ1 IF N=4 THEN PRINT "O.K.":OB(N)=S
PIELER:GOTO 1Ø8Ø
12ØØ2 IF N=5 THEN PRINT "O.K.":OB(N)=S
PIELER:GOTO 1Ø8Ø
12ØØ3 IF N=7 THEN PRINT "O.K.":OB(N)=S
PIELER:GOTO 1Ø8Ø
12ØØ4 IF N=12 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12ØØ5 IF N=15 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12ØØ6 IF N=16 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12ØØ7 IF N=14 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12ØØ8 IF N=3Ø THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12ØØ9 IF N=41 THEN PRINT "O.K.":OB(N)=
```

```
SPIELER:GOTO 1Ø8Ø
12Ø1Ø IF N=45 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12Ø11 IF N=4Ø THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12Ø12 IF N=36 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12Ø13 IF N=35 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
12Ø14 IF N=31 THEN PRINT "O.K.":OB(N)=
SPIELER:GOTO 1Ø8Ø
129ØØ IF N=5 AND OB(41)=-1 THEN PRINT
"O.K.":OB(41)=SPIELER:GOTO 1Ø8Ø
129Ø1 IF N=16 AND OB(34)=-1 THEN PRINT
 "O.K.":OB(34)=SPIELER:GOTO 1Ø8Ø
1299Ø GOSUB 6ØØ:GOTO 1Ø8Ø
13ØØØ IF OB(3Ø)=-1 THEN OB(3Ø)=2Ø:GOSU
B 6Ø6:RETURN
13Ø1Ø IF OB(35)=-1 THEN OB(35)=2Ø:GOSU
B 6Ø6:RETURN
13Ø2Ø IF OB(5)=-1 THEN OB(5)=2Ø:GOSUB
6Ø6:RETURN
13Ø9Ø RETURN
14ØØØ IF OB(34)<>25 THEN RETURN
14Ø1Ø IF OB(35)<>25 THEN RETURN
14Ø2Ø IF OB(41)<>25 THEN RETURN
14Ø3Ø IF OB(4Ø)<>25 THEN RETURN
141ØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOL
OR 4,1,Ø:PRINT "EINE STIMME ERTOENT:":
PRINT :PRINT :PRINT
1411Ø PRINT "DER GROSSE HUMBUG FREUT S
ICH UEBER":PRINT
1412Ø PRINT "DEINE GABEN. ER WIRD SEIN
E ZAUBER-":PRINT :PRINT "KRAFT FUER DI
CH EINSETZEN.":PRINT
1413Ø PRINT :PRINT "GEHE NUN NACH HAUS
E, DENN DORT WIRD":PRINT :PRINT "BEREI
TS EIN FEST VORBEREITET, UM":PRINT
1414Ø PRINT "DAS ERWACHEN AUS DEM LANG
EN SCHLAF ZU":PRINT :PRINT "FEIERN."
1415Ø FOR Q=1 TO 1ØØØ:NEXT Q:GOTO 48ØØ
```

```
15ØØØ IF OB(3Ø)=-1 THEN OB(3Ø)=2Ø:GOSU
B 6Ø6:RETURN
15Ø1Ø IF OB(35)=-1 THEN OB(35)=2Ø:GOSU
B 6Ø6:RETURN
15Ø2Ø IF OB(5)=-1 THEN OB(5)=2Ø:GOSUB
6Ø6:RETURN
15Ø3Ø RETURN
151ØØ IF OB(31)<>-1 THEN RETURN
1511Ø MØ$="ICH BIN ZU SCHWER UND VERSI
NKE !":GOTO 45ØØ
152ØØ IF RND(1)>Ø.3 THEN RETURN
1521Ø MØ$="ICH BIN IN EINE FALLGRUBE G
ESTUERZT.":GOTO 45ØØ
16ØØØ IF N=1Ø AND SPIELER=8 THEN PRINT
 "ES GELINGT MIR NICHT !":GOTO 1Ø8Ø
16Ø1Ø IF N=11 AND SPIELER=8 THEN PRINT
 "ES GELINGT MIR NICHT !":GOTO 1Ø8Ø
16Ø2Ø GOSUB 6ØØ:GOTO 1Ø8Ø
```

GOLDRAUSCH

Viele Worte werden zu diesem Adventure wohl kaum noch nötig sein. Allerdings handelt es sich hier um die vollständige Version, das Abenteuer wird erst richtig beginnen, wenn es Ihnen gelungen ist, das bislang bekannte Territorium zu verlassen.

Sie werden in die dunkle Welt einer alten Goldmine eindringen und es sich zweimal überlegen, ob Sie einen spontanen Einfall probieren, und damit wertvolle Zeit, die hier an der Brenndauer einer Lampe gemessen wird, verstreichen lassen wollen, oder ob Sie lieber auf Nummer sicher gehen und nur durch irgendwelche Überlegungen gerechtfertigte Handlungen durchführen wollen.

Anmerkungen zum Listing "Goldrausch"

Das im Buch abgedruckte Listing wurde wie folgt korrigiert:

1395 -> 2 fehlende Leerzeichen ergänzt.

45ØØ, 461Ø -> 2 Rechtschreibfehler korrigiert.

```
1 REM GOLDRAUSCH, VERSION 1.Ø; ATARI
2 REM (C) WALKOWIAK, OKTOBER 1984
3 REM ----------------------------
1Ø GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
4,1,Ø:FOR I=1 TO 9:PRINT :NEXT I
2Ø PRINT "
  "
3Ø PRINT "         G O L D R A U S C H
  "
4Ø PRINT "
  "
5Ø PRINT "              VERSION 1.Ø
  "
6Ø PRINT "         (C) 1984 BY WALKOWIA
K "
7Ø PRINT "
 ";
8Ø FOR I=1 TO 3ØØØ
9Ø NEXT I
1ØØ REM ********* DATEN DES ADVENTURES
11Ø AR=31:AO=44:AF=7:SPIELER=1:WMAX=6Ø
:WERTUNG=Ø:ZUG=Ø:IMAX=4:LM=Ø:LI=-1:LW=
1:L1=2Ø
15Ø DIM RAUM$(4Ø),DURCHGANG(AR,6),RICH
TUNG$(36),EINGABE$(2Ø),T$(4Ø),OB(AO),L
EERZEILE$(38),DZ$(5Ø),ZENDE$(3),FL(AF)
16Ø DIM VERBEN$(13Ø),EVERB$(2Ø),EOBJEK
T$(2Ø),OBJEKTE$(18Ø),MØ$(6Ø),M1$(3Ø),M
2$(3Ø),M3$(3Ø),M4$(38),M5$(38)
17Ø VERBEN$="UNTERSUCHENIMM      LEG
     OEFFNE    BENUTZE   ZERSTOERE ZUE
NDE    FUELLE    BETRETE   LOESCHE   "
171 VERBEN$(LEN(VERBEN$)+1)="BEFESTIGE
 "
18Ø OBJEKTE$="BAEUBAEUFELSHUETFLASHONI
KISTREGASPREERDLTRUHSILBHOEHBAERBUESEI
SENUGGEISELORESCHISEILHACKLATEGANG"
181 OBJEKTE$(LEN(OBJEKTE$)+1)="HAKESCH
ABRETSCHUSAECWAENFLUELEICKLUMMUENSPINW
ANDSCHIGOLDUFERSEE STEIZUENLUNTPARA"
19Ø GOTO 6ØØ
```

```
2ØØ REM *********** RAUMBESCHREIBUNGEN
2Ø1 PRINT "IM WALD.":RETURN
2Ø2 PRINT "IM WALD.":RETURN
2Ø3 PRINT "IM WALD VOR EINEM FELSHANG.
":RETURN
2Ø4 PRINT "AUF EINER LICHTUNG.":RETURN

2Ø5 PRINT "AUF EINER LICHTUNG.":RETURN

2Ø6 PRINT "AM EINGANGSSTOLLEN EINES":P
RINT "ALTEN BERGWERKES.":RETURN
2Ø7 PRINT "IM EINGANSSTOLLEN.":RETURN
2Ø8 PRINT "IN EINER NISCHE DES GANGES.
":RETURN
2Ø9 PRINT "AM ENDE DES GANGES.":RETURN

21Ø PRINT "IN EINEM SEITENGANG.":RETUR
N
211 PRINT "AN EINER ALTEN ABBAUSTELLE.
":RETURN
212 PRINT "IN DER HOEHLE.":RETURN
213 PRINT "IN DER HOEHLE.":RETURN
214 PRINT "IN DER HOEHLE.":RETURN
215 PRINT "IN DER HOEHLE.":RETURN
216 PRINT "AUF EINEM GANG.":RETURN
217 PRINT "IN EINER NEBENHOEHLE.":RETU
RN
218 PRINT "AUF DEM BODEN DES SCHACHTES
.":RETURN
219 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
22Ø PRINT "IN EINEM BREITEN GANG.":RET
URN
221 PRINT "IN EINER ALTEN ABBAUSTRECKE
.":RETURN
222 PRINT "IN EINEM FELSENDOM.":RETURN

223 PRINT "IN EINEM UNTERIRDISCHEN":PR
INT "PARADIES.":RETURN
224 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
```

```
225 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
226 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
227 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
228 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
229 PRINT "AUF EINEM KURVENREICHEN GAN
G.":RETURN
23Ø PRINT "VOR EINEM UNTERIRDISCHEN SE
E.":RETURN
231 PRINT "IN EINER KLEINEN HOEHLE.":R
ETURN
3ØØ REM ***************** GEGENSTAENDE
3Ø1 T$="VIELE GROSSE BAEUME":RETURN
3Ø2 T$="VIELE GROSSE BAEUME":RETURN
3Ø3 T$="EINIGE FELSBROCKEN":RETURN
3Ø4 T$="EINE VERFALLENE HOLZHUETTE":RE
TURN
3Ø5 T$="EINE VERSCHMUTZTE KORBFLASCHE"
:RETURN
3Ø6 T$="HONIG":RETURN
3Ø7 T$="EINE HOLZKISTE":RETURN
3Ø8 T$="EIN KAPPRIGES REGAL":RETURN
3Ø9 T$="ETWAS SPRENGSTOFF":RETURN
31Ø T$="EIN DUESTERES ERDLOCH":RETURN
311 T$="EINE ROSTIGE EISENTRUHE":RETUR
N
312 T$="*SILBERMUENZEN*":RETURN
313 T$="EINE DUESTERE FELSENHOEHLE":RE
TURN
314 T$="EINEN GRIMMIGEN BAEREN":RETURN

315 T$="NIEDRIGE BUESCHE":RETURN
316 T$="MEHRERE EISENSTANGEN":RETURN
317 T$="*NUGGETS*":RETURN
318 T$="EINE EISENSTANGE":RETURN
319 T$="EINE ALTE LORE":RETURN
32Ø T$="VERROSTETE SCHIENENSTRAENGE":R
ETURN
```

```
321 T$="EINE SEIL":RETURN
322 T$="EINE SCHWERE SPITZHACKE":RETUR
N
323 T$="EINE ALTE LATERNE":RETURN
324 T$="DAS GANGENDE":RETURN
325 T$="EISENHAKEN":RETURN
326 T$="EINEN SCHACHT":RETURN
327 T$="EINE BRETTERWAND":RETURN
328 T$="ABBAUSCHUTT":RETURN
329 T$="ALTE SAECKE":RETURN
33Ø T$="FEUCHTE STEINWAENDE":RETURN
331 T$="UEBELRIECHENDE FLUESSIGKEIT":R
ETURN
332 T$="EINE LEICHE":RETURN
333 T$="*SILBERKLUMPEN*":RETURN
334 T$="*GOLDMUENZEN*":RETURN
335 T$="SPINNWEBEN":RETURN
336 T$="EINE GOLDENE WAND":RETURN
337 T$="EIN SCHILD":RETURN
338 T$="*GOLD*":RETURN
339 T$="EIN SEEUFER":RETURN
34Ø T$="DEN SEE":RETURN
341 T$="GOLDENE STEINE":RETURN
342 T$="ZUENDSCHNUR":RETURN
343 T$="EIN LUNTERO":RETURN
344 T$="":RETURN
4ØØ REM **** POSITIONEN OBJEKTE ***
41Ø DATA 1,2,2,3,Ø,Ø,3,Ø,Ø,Ø,Ø,Ø,5,Ø,4
,6,5,Ø,7,7,Ø,8,8,9,Ø,9,1Ø,11,Ø,13,17,1
8,Ø,Ø,2Ø,22,22,31,3Ø,3Ø,Ø,Ø,-1,Ø
5ØØ REM RICHTUNGSTABELLE
5Ø1 DATA 1,1,1,2,Ø,Ø,2,1,1,3,Ø,Ø,Ø,4,2
,Ø,Ø,Ø,3,Ø,5,Ø,Ø,Ø,Ø,Ø,6,4,Ø,Ø,7,Ø,1,5
,Ø,Ø,8,6,Ø,Ø,Ø,Ø,9,7,8,1Ø,Ø,Ø
5Ø2 DATA Ø,8,Ø,Ø,Ø,Ø,11,Ø,8,Ø,Ø,Ø,Ø,1Ø
,Ø,Ø,Ø,Ø,Ø,5,Ø,13,Ø,Ø,Ø,Ø,12,14,Ø,Ø,Ø,
16,13,15,Ø,Ø,Ø,Ø,14,Ø,Ø,Ø
5Ø3 DATA 14,Ø,Ø,17,Ø,Ø,Ø,Ø,16,Ø,Ø,Ø,Ø,
Ø,19,Ø,Ø,Ø,24,Ø,2Ø,18,Ø,Ø,Ø,Ø,21,19,Ø,
Ø,22,11,11,2Ø,11,Ø,Ø,21,Ø,Ø,27,Ø
5Ø4 DATA Ø,Ø,Ø,Ø,Ø,Ø,26,19,24,24,24,24
```

```
,27,24,24,29,26,24,27,24,26,27,27,24,2
5,24,26,27,26,24,24,26,26,27,Ø,Ø
5Ø5 DATA Ø,27,3Ø,27,Ø,Ø,28,Ø,Ø,Ø,Ø,Ø,3
Ø,Ø,Ø,Ø,Ø,Ø
6ØØ REM ***************** MITTEILUNGEN
6Ø1 M1$="ICH SEHE NICHTS BESONDERES."
6Ø2 M2$="SO STARK BIN ICH NICHT !"
6Ø3 M3$="WIE STELLST DU DIR DAS VOR ?"
6Ø4 M4$="DER BAER GREIFT SICH DEN HONI
G, UND"
6Ø5 M5$="VERSCHWINDET IN DER TIEFE DER
 HOEHLE."
699 REM ********* 2. TITEL: EINLEITUNG
7ØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
 4,1,Ø
71Ø PRINT "Herzlich Willkommen zu:  GO
LDRAUSCH":PRINT "ein Adventure fuer de
n Atari 8ØØ XL":?
72Ø FOR I=1 TO 37:PRINT CHR$(13);:NEXT
 I:PRINT
73Ø PRINT "Auf der Suche nach dem Glue
ck in der":PRINT "neuen Welt begegnete
n Sie vor einigen"
74Ø PRINT "Tagen einem alten todkranke
n Mann,":PRINT "dem Sie in seinen letz
ten Stunden"
75Ø PRINT "Beistand leisteten.":PRINT
"Aus Dankbarkeit berichtete er Ihnen"
76Ø PRINT "von seiner Goldmine und den
 dort ver-"
77Ø PRINT "steckten Resten seines Verm
oegens.":PRINT "Zahllosen Gefahren wid
erstanden Sie"
78Ø PRINT "auf dem Weg dorthin, bald w
erden Sie":PRINT "Ihr Ziel erreicht ha
ben, und es wird"
785 PRINT "sich zeigen, ob der Alte di
e Wahrheit gesprochen, oder im Fiebert
raum gere- det hatte.":PRINT
79Ø FOR I=1 TO 37:PRINT CHR$(13);:NEXT
 I:PRINT
```

```
795 PRINT "WUENSCHEN SIE RATSCHLAEGE F
UER IHR             WEITERES VORGEHEN
";:INPUT EINGABE$
798 IF EINGABE$(1,1)="J" THEN GOSUB 8Ø
Ø
799 GOTO 9ØØ
8ØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
 4,1,Ø
82Ø PRINT "           ATARI - VENTURES
          ":PRINT "     (C) 1984 BV JO
ERG WALKOWIAK     ":REM INVERS
83Ø PRINT :PRINT "Stellen Sie sich ein
en Roboter vor,   den Sie mit zahlreic
hen Kommandos"
84Ø PRINT "steuern koennen. Ich bin di
eser Ro-   boter, und ich werde mich f
uer Sie"
85Ø PRINT "den Gefahren der verwegenst
en Aben-   teuer aussetzen."
855 PRINT "Damit Sie mich sinnvoll agi
eren las-  en koennen, werde ich Ihnen
 die":PRINT "Situation in der ich mich
 gerade"
86Ø PRINT "befinde, jeweils genau besc
hreiben.":PRINT "Anschliessend sagen S
ie mir mit zwei"
863 PRINT "Worten wie zum Beispiel UNT
ERSUCHE "
87Ø PRINT "TUER, NIMM MESSER, was ich
tun soll.":PRINT :PRINT "Darueber hina
us verstehe ich die Be-  fehle."
88Ø PRINT "  INVENTUR, SCORE, ENDE,":P
RINT
881 PRINT "        SAVE, LOAD, VOK, HE
LP":PRINT
89Ø PRINT "BITTE DRUECKEN SIE <RETURN>
 UND ...";:INPUT EINGABE$:GRAPHICS Ø:R
ETURN :REM INVERS
9ØØ FOR I=1 TO AO
91Ø READ LAGEORT:OB(I)=LAGEORT
92Ø NEXT I
```

```
93Ø FOR I=1 TO AF:FL(I)=Ø:NEXT I
94Ø FOR RAUM=1 TO AR
95Ø FOR RICHTUNG=1 TO 6
96Ø READ ZIEL:DURCHGANG(RAUM,RICHTUNG)
=ZIEL
97Ø NEXT RICHTUNG
98Ø NEXT RAUM
1ØØØ GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:TRAP 1Ø8Ø:GOTO 1Ø2Ø
1Ø1Ø REM ******** KLARTEXT: RICHTUNGEN
1Ø11 T$="NORDEN, ":RETURN
1Ø12 T$="SUEDEN, ":RETURN
1Ø13 T$="WESTEN, ":RETURN
1Ø14 T$="OSTEN, ":RETURN
1Ø15 T$="OBEN, ":RETURN
1Ø16 T$="UNTEN, ":RETURN
1Ø2Ø LEERZEILE$="
                "
1Ø3Ø T$=""
1Ø4Ø ZENDE$=CHR$(3Ø):ZENDE$(LEN(ZENDE$
)+1)=CHR$(3Ø):ZENDE$(LEN(ZENDE$)+1)=".
"
1Ø8Ø PRINT :ZUG=ZUG+1:LW=LW+1
1Ø84 IF LW=LM THEN GOSUB 3ØØØ
1Ø85 IF WERTUNG=WMAX THEN GOTO 48ØØ
1Ø9Ø FOR ZEILE=Ø TO 1Ø:POSITION 2,ZEIL
E:PRINT LEERZEILE$:NEXT ZEILE
1Ø91 REM ****************** KEIN LICHT
1Ø92 IF LI=-1 THEN 11ØØ
1Ø93 PRINT "ICH WEISS NICHT GENAU, WO
ICH BIN.":PRINT "ES IST ZU DUNKEL, UM
ETWAS ZU SEHEN."
1Ø95 PRINT :PRINT "AUCH DIE AUSGAENGE
SEHE ICH NICHT":PRINT "MEHR !":GOTO 13
2Ø
11ØØ POSITION 2,Ø:PRINT "ICH BIN ";:GO
SUB 2ØØ+SPIELER
111Ø DZ$="ICH SEHE ":GEDRUCKT=Ø
112Ø FOR I=1 TO AO
113Ø IF OB(I)<>SPIELER THEN GOTO 117Ø
114Ø GEDRUCKT=-1:GOSUB 3ØØ+I:T$(LEN(T$
```

```
)+1)=", "
115Ø IF LEN(DZ$)+LEN(T$)>=38 THEN PRIN
T DZ$:DZ$=T$(1,LEN(T$)):T$="":GOTO 117
Ø
116Ø DZ$(LEN(DZ$)+1)=T$:T$=""
117Ø NEXT I
118Ø IF  NOT GEDRUCKT THEN DZ$(LEN(DZ$
)+1)="NICHTS BESONDERES. "
119Ø DZ$(LEN(DZ$)+1)=ZENDE$
12ØØ PRINT DZ$
121Ø PRINT LEERZEILE$
122Ø DZ$="ICH KANN NACH ":GEDRUCKT=Ø
123Ø FOR RICHTUNG=1 TO 6
124Ø IF DURCHGANG(SPIELER,RICHTUNG)=Ø
THEN GOTO 128Ø
125Ø GOSUB 1Ø1Ø+RICHTUNG:GEDRUCKT=-1
126Ø IF LEN(DZ$)+LEN(T$)>=38 THEN PRIN
T DZ$:DZ$=T$(1,LEN(T$)):T$="":GOTO 128
Ø
127Ø DZ$(LEN(DZ$)+1)=T$:T$=""
128Ø NEXT RICHTUNG
129Ø IF  NOT GEDRUCKT THEN DZ$(LEN(DZ$
)+1)="NIRGENDWO. "
13ØØ DZ$(LEN(DZ$)+1)=ZENDE$
131Ø PRINT DZ$
132Ø PRINT "------------------------
----------":REM 37 CONTROL R
134Ø IF WERTUNG=IMAX THEN GOTO 48ØØ
135Ø IF SPIELER=15 THEN MØ$="LEIDER WA
R DORT DER BAER.":FOR I=1 TO 6ØØ:NEXT
I:GOTO 45ØØ
136Ø IF LW<=Ø THEN LI=Ø
137Ø IF EX=ZUG AND SPIELER=2Ø THEN MØ$
="RUMMS ! - AUCH MICH HAT'S ZERRISSEN.
":GOTO 45ØØ
138Ø IF SPIELER>18 AND OB(23)<>-1 THEN
 LI=Ø
139Ø POSITION 2,23:PRINT "WAS SOLL ICH
 TUN ";:INPUT EINGABE$
1395 RL=LM-LW:IF (RL>Ø AND RL<15) THEN
 PRINT "IN ";LM-LW;" ZUEGEN STEHE ICH
```

```
IM DUNKLEN!"
14ØØ IF LEN(EINGABE$)>2 THEN 148Ø
141Ø IF EINGABE$="N" AND DURCHGANG(SPI
ELER,1)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,1):PRINT "O.K.":GOTO 1Ø8Ø
142Ø IF EINGABE$="S" AND DURCHGANG(SPI
ELER,2)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,2):PRINT "O.K.":GOTO 1Ø8Ø
143Ø IF EINGABE$="W" AND DURCHGANG(SPI
ELER,3)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,3):PRINT "O.K.":GOTO 1Ø8Ø
144Ø IF EINGABE$="O" AND DURCHGANG(SPI
ELER,4)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,4):PRINT "O.K.":GOTO 1Ø8Ø
145Ø IF EINGABE$="OB" AND DURCHGANG(SP
IELER,5)<>Ø THEN SPIELER=DURCHGANG(SPI
ELER,5):PRINT "O.K.":GOTO 1Ø8Ø
146Ø IF EINGABE$="U" AND DURCHGANG(SPI
ELER,6)<>Ø THEN SPIELER=DURCHGANG(SPIE
LER,6):PRINT "O.K.":GOTO 1Ø8Ø
147Ø IF LEN(EINGABE$)<3 THEN PRINT "DA
HIN FUEHRT KEIN WEG !":GOTO 1Ø8Ø
148Ø IF LEN(EINGABE$)>8 THEN GOTO 2ØØØ
1499 REM ************** START INVENTUR
15ØØ IF EINGABE$(1,3)<>"INV" THEN GOTO
 156Ø
151Ø PRINT "ICH TRAGE FOLGENDES MIT MI
R:"
152Ø FOR I=1 TO AO
153Ø IF OB(I)=-1 THEN GOSUB 3ØØ+I:PRIN
T T$
154Ø NEXT I
155Ø GOTO 1Ø8Ø
1551 REM ************** ENDE  INVENTUR
156Ø IF EINGABE$(1,3)<>"SCO" THEN GOTO
 16ØØ
1561 PRINT "VON ";WMAX;" PUNKTEN HAST
DU IN ";ZUG;" ZUEGEN":PRINT WERTUNG;"
PUNKTE ERREICHT ! DAS ENTSPRICHT"
1563 PRINT "EINEM SCHNITT VON ";WERT/Z
UG;" PUNKTEN.":GOTO 1Ø8Ø
```

```
1599 REM ****************** SAVE GAME
16ØØ IF EINGABE$(1,3)<>"SAV" THEN GOTO
 17ØØ
16Ø5 PRINT "UNTER WELCHEM NAMEN ";:INP
UT EINGABE$:IF LEN(EINGABE$)>8 THEN PR
INT "BITTE ETWAS KUERZER !":GOTO 16Ø5
161Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
=".DAT"
1615 T$(LEN(T$)+1)=EINGABE$
162Ø OPEN #1,8,Ø,T$
1625 PRINT #1,SPIELER
163Ø FOR I=1 TO AO
1631 PRINT #1,OB(I)
1632 NEXT I
1635 FOR RAUM=1 TO AR
1636 FOR RICHTUNG=1 TO 6
1637 PRINT #1,DURCHGANG(RAUM,RICHTUNG)
1638 NEXT RICHTUNG
1639 NEXT RAUM
1645 FOR I=1 TO AF
1646 PRINT #1,FL(I)
1647 NEXT I
165Ø CLOSE #1
167Ø PRINT "O.K.":GOTO 1Ø8Ø
1699 REM ****************** LOAD GAME
17ØØ IF EINGABE$(1,3)<>"LOA" THEN GOTO
 18ØØ
17Ø5 PRINT "WELCHES SPIEL ";:INPUT EIN
GABE$:IF LEN(EINGABE$)>8 THEN PRINT "D
AS KANN ES NICHT GEBEN !":GOTO 17Ø5
171Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
=".DAT"
1715 T$(LEN(T$)+1)=EINGABE$
172Ø OPEN #1,4,Ø,T$
1725 INPUT #1,SPIELER
173Ø FOR I=1 TO AO
1731 INPUT #1,LAGEORT:OB(I)=LAGEORT
1732 NEXT I
1735 FOR RAUM=1 TO AR
1736 FOR RICHTUNG=1 TO 6
1737 INPUT #1,ZIEL:DURCHGANG(RAUM,RICH
```

```
TUNG)=ZIEL
1738 NEXT RICHTUNG
1739 NEXT RAUM
1745 FOR I=1 TO AF
1746 INPUT #1,FLAG:FL(I)=FLAG
1747 NEXT I
175Ø CLOSE #1
177Ø PRINT "O.K.":GOTO 1Ø8Ø
18ØØ IF EINGABE$(1,3)<>"VOK" THEN GOTO
 19ØØ
181Ø GOSUB 188Ø:GEDRUCKT=Ø
183Ø FOR I=1 TO LEN(VERBEN$) STEP 1Ø
184Ø PRINT VERBEN$(I,I+9):GEDRUCKT=GED
RUCKT+1
185Ø IF GEDRUCKT=18 THEN GOSUB 189Ø:GE
DRUCKT=1:GOSUB 188Ø
186Ø NEXT I
187Ø GOSUB 189Ø:GRAPHICS Ø:SETCOLOR 2,
1,Ø:SETCOLOR 4,1,Ø:GOTO 1Ø8Ø
188Ø GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLO
R 4,1,Ø:PRINT "ICH KENNE FOLGENDE HAND
LUNGEN ...":PRINT :RETURN
189Ø PRINT :PRINT :PRINT "<RETURN> DRU
ECKEN ...":INPUT EINGABE$:RETURN
19ØØ IF EINGABE$(1,3)<>"INS" THEN GOTO
 195Ø
191Ø GOSUB 8ØØ
192Ø GOTO 1Ø8Ø
195Ø IF EINGABE$(1,3)<>"END" THEN GOTO
 197Ø
196Ø GRAPHICS Ø:PRINT "DER AUTOR WUENS
CHT IHNEN MEHR GLUECK":PRINT "FUERS NA
ECHSTE MAL !":PRINT :PRINT :PRINT :END

197Ø IF EINGABE$(1,3)<>"HEL" THEN GOTO
 2ØØØ
1971 IF SPIELER=4 AND OB(1Ø)=Ø THEN PR
INT "FAST WAERE ICH IN EINE GRUBE":PRI
NT "GEFALLEN.":GOTO 1Ø8Ø
1972 IF SPIELER=4 AND OB(11)<>SPIELER
AND  NOT FL(2) THEN PRINT "ICH BRAUCHE
```

```
 ETWAS, UM DIE KETTE ZU":PRINT "ZERREI
SSEN"
1975 PRINT "ERST SEHEN, DANN DENKEN UN
D ZULETZT":PRINT "HANDELN !!":GOTO 1Ø8
Ø
1979 REM ****************** ENDE HELP
1999 REM ************ EINGABE TRENNEN
2ØØØ LN=LEN(EINGABE$)
2Ø1Ø FOR I=1 TO LN
2Ø2Ø IF EINGABE$(I,I)<>" " THEN NEXT I
2Ø3Ø EVERB$=EINGABE$(1,I)
2Ø4Ø IF LEN(EVERB$)=LN-1 THEN GOTO 2Ø9
Ø
2Ø5Ø EOBJEKT$=EINGABE$(I+1,LN)
2Ø6Ø VN=Ø:N=Ø
2Ø7Ø EVERB$=EVERB$(1,3)
2Ø8Ø EOBJEKT$=EOBJEKT$(1,3)
2Ø9Ø FOR I=1 TO LEN(VERBEN$)-2 STEP 1Ø
21ØØ VN=VN+1
211Ø IF VERBEN$(I,I+2)=EVERB$ THEN 214
Ø
212Ø NEXT I
213Ø PRINT "ICH VERSTEHE DAS VERB NICH
T !":GOTO 1Ø8Ø
214Ø FOR I=1 TO LEN(OBJEKTE$) STEP 4
215Ø N=N+1
216Ø IF OBJEKTE$(I,I+2)=EOBJEKT$ THEN
GOTO 22ØØ
217Ø NEXT I
218Ø PRINT "ICH VERSTEHE DAS OBJEKT NI
CHT.":GOTO 1Ø8Ø
219Ø REM        UNT    NIM   LEG   OEF
BEN   ZER
22ØØ ON VN GOTO 5ØØØ,221Ø,7ØØØ,8ØØØ,9Ø
ØØ,1ØØØØ,11ØØØ,12ØØØ,13ØØØ,14ØØØ,15ØØØ
22Ø9 REM INVENTORY BEGRENZEN
221Ø ANZ=Ø:FOR I=1 TO AO
222Ø IF OB(I)=-1 THEN ANZ=ANZ+1
223Ø IF ANZ=IMAX THEN PRINT "NEIN DANK
E.- ICH TRAGE SCHON GENUG.":GOTO 1Ø8Ø
224Ø NEXT I
```

```
225Ø GOTO 6ØØØ
226Ø REM ENDE INVTEST
2999 REM UP LICHTSCHALTER
3ØØØ IF LI=-1 THEN LI=Ø:GOTO 3Ø2Ø
3Ø1Ø IF LI=Ø THEN LI=-1
3Ø2Ø LW=Ø:RETURN
45ØØ GRAPHICS Ø:REM ****** SPIELER TOT
46ØØ PRINT "AUCH DAS NOCH !":PRINT :PR
INT MØ$
461Ø PRINT :PRINT "ICH BIN TOT !":PRIN
T
462Ø PRINT "SOLL ICH ES NOCH EINMAL VE
RSUCHEN ";:INPUT EINGABE$
463Ø IF EINGABE$(1,1)="J" THEN CLR :GO
TO 1ØØ
464Ø GOTO 196Ø
48ØØ GRAPHICS Ø
481Ø PRINT "HERZLICHEN GLUECKWUNSCH !"
482Ø PRINT :PRINT :PRINT "SIE HABEN DI
E IHNEN GESTELLTE AUFGABE":PRINT :PRIN
T "GELOEST, UND DUERFEN SICH AN EINEM"
483Ø PRINT "ANDEREN ADVENTURE VERSUCHE
N."
494Ø PRINT :PRINT :PRINT :END
49ØØ PRINT "ACHTUNG FEHLER !":GOTO 1Ø8
Ø
4999 REM ***** SPIELERZUEGE AUSFUEHREN
5ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN GOTO 59ØØ
5ØØ2 IF N=1 THEN PRINT M1$:GOTO 1Ø8Ø
5ØØ3 IF N=3 THEN PRINT M1$:GOTO 1Ø8Ø
5ØØ4 IF N=4 THEN PRINT "IN EINER ECKE
STEHT EIN REGAL.":OB(8)=SPIELER:GOTO 1
Ø8Ø
5ØØ5 IF N=5 THEN PRINT "DIE FLASCHE IS
T GEFUELLT MIT HONIG.":GOTO 1Ø8Ø
5ØØ6 IF N=6 THEN PRINT M1$:GOTO 1Ø8Ø
5ØØ7 IF N=7 AND OB(9)=Ø THEN PRINT "IN
 DER KISTE LIEGT SPRENGSTOFF.":GOTO 1Ø
8Ø
5ØØ8 IF N=8 AND OB(5)=Ø THEN PRINT "AU
```

```
F DEM REGAL STEHT EINE KORBFLASCHE.":O
B(5)=SPIELER:GOTO 1Ø8Ø
5ØØ9 IF N=8 AND OB(5)<>Ø THEN PRINT M1
$:GOTO 1Ø8Ø
5Ø1Ø IF N=1Ø THEN PRINT "IN DEM ERDLOC
H LIEGT EINE EISENTRUHE.":OB(11)=SPIEL
ER:GOTO 1Ø8Ø
5Ø11 IF N=11 AND  NOT FL(1) THEN PRINT
 "SIE IST MIT EINER EISENKETTE VER-":P
RINT "SCHLOSSEN.":GOTO 1Ø8Ø
5Ø12 IF N=11 AND FL(1) AND  NOT FL(2)
THEN PRINT "AUSSEN SEHE ICH NICHTS BES
ONDERES.":GOTO 1Ø8Ø
5Ø13 IF N=11 AND FL(1) AND FL(2) AND O
B(12)=Ø THEN PRINT "SIE IST VOLLER SIL
BERMUENZEN.":OB(12)=SPIELER:GOTO 1Ø8Ø
5Ø14 IF N=12 THEN PRINT "GENAU DAS SUC
HE ICH !!":GOTO 1Ø8Ø
5Ø15 IF N=13 THEN PRINT "ICH HABE EINE
N BAEREN AUFGESCHRECKT.":OB(14)=SPIELE
R:GOTO 1Ø8Ø
5Ø17 IF N=15 THEN PRINT "DAZWISCHEN IS
T EIN ERDLOCH.":OB(1Ø)=SPIELER:GOTO 1Ø
8Ø
5Ø18 IF N=16 THEN PRINT "SIE SEHEN SEH
R STABIL AUS.":GOTO 1Ø8Ø
5Ø19 IF N=17 THEN PRINT "ES HANDELT SI
CH UM PURES GOLD !":GOTO 1Ø8Ø
5Ø2Ø IF N=19 AND OB(21)=Ø THEN PRINT "
DARAN HAENGT IMMER NOCH DAS ZUGSEIL.":
OB(21)=SPIELER:GOTO 1Ø8Ø
5Ø21 IF N=23 THEN PRINT "ES IST EINE A
LTE OELLAMPE":GOTO 1Ø8Ø
5Ø22 IF N=24 THEN PRINT "IN DER WAND S
TECKT EIN HAKEN.":OB(25)=SPIELER:GOTO
1Ø8Ø
5Ø23 IF N=25 THEN PRINT "TUT MIR LEID
- ER STECKT ZU FEST.":GOTO 1Ø8Ø
5Ø24 IF N=26 THEN MØ$="ER IST SEHR TIE
F - UND ICH WAR ZU     DICHT AM RAND."
:GOTO 45ØØ
```

```
5Ø25 IF N=27 THEN PRINT "IHR TISCHLER
WAR EIN KOENNER.":GOTO 1Ø8Ø
5Ø26 IF N=28 AND OB(N)=Ø THEN PRINT "I
CH HABE ZWEI SAECKE ENTDECKT.":OB(29)=
SPIELER:GOTO 1Ø8Ø
5Ø27 IF N=29 THEN PRINT "SIE SIND GEFU
ELLT MIT GOLDSTAUB.":GOTO 1Ø8Ø
5Ø28 IF N=31 THEN PRINT "SIE IST OELIG
 SCHMIERIG UND KLEBT AN":PRINT "DEN FI
NGERN.":GOTO 1Ø8Ø
5Ø29 IF N=32 THEN PRINT "SIE STINKT !"
5Ø3Ø IF N=32 AND OB(34)=Ø THEN PRINT "
IN DER JACKENTASCHE SIND GOLDMUENZEN."
:OB(34)=SPIELER:GOTO 1Ø8Ø
5Ø31 IF N=35 THEN PRINT "ES SIND GAR K
EINE SPINNWEBEN.":OB(N)=Ø:OB(42)=SPIEL
ER
5Ø32 IF N=35 THEN PRINT "ES IST EINE Z
UENDSCHNUR.":GOTO 1Ø8Ø
5Ø33 IF N=36 THEN PRINT "ES IST TATSAE
CHLICH REINES GOLD.":GOTO 1Ø8Ø
5Ø34 IF N=37 THEN PRINT "DARAUF STEHT:
":PRINT "WENN ERST MAL DIE GIER ERWACH
T,":PRINT "DER TOD AUCH OFT GESCHAEFTE
 MACHT"
5Ø35 IF N=37 THEN GOTO 1Ø8Ø
5Ø36 IF N=42 THEN PRINT "SIE FUEHRT HI
NAUF ZUR DECKE.":GOTO 1Ø8Ø
5Ø37 IF N=43 THEN PRINT "ES IST EINS D
ER UEBLICHEN WESTERN-":PRINT "FEUERZEU
GE."
5Ø38 IF N=32 AND FL(4) AND OB(33)<>-2
THEN PRINT "SIE LAG AUF SILBERSTUECKEN
.":OB(33)=SPIELER:GOTO 1Ø8Ø
5899 PRINT M1$:GOTO 1Ø8Ø
59ØØ REM ** GEGENSTAND NICHT VORHANDEN
59Ø1 IF N=1 AND SPIELER=2 THEN PRINT M
1$:GOTO 1Ø8Ø
59Ø2 IF N=6 AND OB(5)=-1 THEN PRINT "E
R IST SUESS UND GUT.":GOTO 1Ø8Ø
59Ø3 IF N=6 AND OB(5)<>-1 THEN PRINT "
```

```
ICH HABE KEINEN HONIG !":GOTO 1Ø8Ø
59Ø4 IF (N=9 AND (OB(7)=SPIELER OR OB(
7)=-1)) THEN PRINT "ER SIEHT SEHR EXPL
OSIV AUS !":GOTO 1Ø8Ø
591Ø IF N=1 AND SPIELER=5 AND OB(14)=5
 THEN PRINT "ICH SCHEINE SEINEN APPETI
T ANZUREGEN.":GOTO 1Ø8Ø
5911 IF N=2Ø AND SPIELER=22 THEN ? "WE
NN ERST MAL DIE GIER ERWACHT,":PRINT "
DER TOD AUCH OFT GESCHAEFTE MACHT.":GO
TO 1Ø8Ø
5912 IF N=44 AND SPIELER=23 THEN MØ$="
ICH BIN IM TOTENREICH.":GOTO 45ØØ
599Ø PRINT "SO ETWAS SEHE ICH HIER NIC
HT !":GOTO 1Ø8Ø
6ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN GOTO 69ØØ
6ØØ1 IF N=1 THEN PRINT M2$:GOTO 1Ø8Ø
6ØØ2 IF N=3 THEN PRINT M2$:GOTO 1Ø8Ø
6ØØ3 IF N=4 THEN PRINT M3$:GOTO 1Ø8Ø
6ØØ4 IF N-8 THEN PRINT M2$:GOTO 1Ø8Ø
6ØØ5 IF N=11 THEN PRINT M2$:GOTO 1Ø8Ø
6ØØ6 IF N=1Ø THEN PRINT M3$:GOTO 1Ø8Ø
6ØØ7 IF N=13 THEN PRINT M3$:GOTO 1Ø8Ø
6ØØ8 IF N=15 THEN PRINT M2$:GOTO 1Ø8Ø
6Ø1Ø IF N=5 THEN OB(5)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
6Ø11 IF N=6 THEN OB(5)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
6Ø12 IF N=7 THEN OB(7)=-1:PRINT "O.K."
:GOTO 1Ø8Ø
6Ø14 IF N=12 THEN OB(12)=-2:PRINT "O.K
.":WERTUNG=WERTUNG+1Ø:GOTO 1Ø8Ø
6Ø15 IF N=1 AND SPIELER=5 THEN MØ$="DE
R BAER HAT MICH ERSCHLAGEN.":GOTO 45ØØ
6Ø16 IF N=16 THEN OB(18)=-1:PRINT "O.K
.":GOTO 1Ø8Ø
6Ø17 IF N=17 AND FL(3) THEN PRINT "O.K
.":OB(17)=-2:WERTUNG=WERTUNG+1Ø:GOTO 1
Ø8Ø
6Ø18 IF N=17 AND  NOT FL(3) THEN MØ$="
```

```
EIN BAER STUERZT SICH AUF MICH.":GOTO
45ØØ
6Ø19 IF N=12 AND OB(N)=-2 THEN PRINT "
DAS SILBER HABE ICH BEREITS.":GOTO 1Ø8
Ø
6Ø2Ø IF N=17 AND OB(N)=-2 THEN PRINT "
DAS GOLD HABE ICH BEREITS.":GOTO 1Ø8Ø
6Ø21 IF N=19 THEN PRINT M2$:GOTO 1Ø8Ø
6Ø22 IF N=21 AND FL(4) THEN PRINT "O.K
.":OB(N)=-1:GOTO 1Ø8Ø
6Ø23 IF N=22 THEN PRINT "O.K.":OB(N)=-
1:GOTO 1Ø8Ø
6Ø24 IF N=24 THEN PRINT "O.K.":OB(N)=-
1:GOTO 1Ø8Ø
6Ø25 IF N=25 THEN PRINT "ER STECKT ZU
TIEF IM FELS.":GOTO 1Ø8Ø
6Ø26 IF N=27 THEN PRINT M3$:GOTO 1Ø8Ø
6Ø27 IF N=28 THEN PRINT "UND WAS SOLL
ICH DAMIT ?":GOTO 1Ø8Ø
6Ø3Ø IF N=31 AND OB(5)<>-1 THEN PRINT
"WOMIT DENN ?":GOTO 1Ø8Ø
6Ø31 IF N=31 AND OB(5)=-1 THEN PRINT "
O.K. - DIE FLASCHE IST VOLL.":FL(6)=-1
:GOTO 1Ø8Ø
6Ø32 IF N=32 THEN PRINT "O.K. - OH, WA
S IST DAS DENN" ?":FL(5)=-1:GOTO 1Ø8Ø
6Ø33 IF N=33 AND OB(N)=SPIELER THEN PR
INT "O.K.":OB(N)=-2:WERTUNG=WERTUNG+1Ø
:GOTO 1Ø8Ø
6Ø34 IF N=34 AND OB(N)=SPIELER THEN PR
INT "O.K.":OB(N)=-2:WERTUNG=WERTUNG+1Ø
:GOTO 1Ø8Ø
6Ø35 IF N=35 THEN PRINT "ES SIND GAR K
EINE SPINNWEBEN, ES WAR":PRINT "EINE Z
UENDSCHNUR.":OB(N)=Ø:OB(42)=-1:GOTO 1Ø
8Ø
6Ø37 IF N=37 THEN PRINT "O.K.":OB(N)=-
1:GOTO 1Ø8Ø
6Ø38 IF N=41 THEN PRINT "O.K.":OB(N)=-
1:GOTO 1Ø8Ø
6Ø39 IF N=42 THEN PRINT "O.K.":OB(N)=-
```

```
1:GOTO 1Ø8Ø
6Ø4Ø IF N=43 THEN PRINT "O.K.":OB(N)=-
1:GOTO 1Ø8Ø
6Ø41 IF N=21 AND  NOT FL(4) THEN PRINT
 "ES IST AN DER LORE BEFESTIGT.":GOTO
1Ø8Ø
6Ø42 IF N=23 THEN PRINT "O.K.":OB(N)=-
1:GOTO 1Ø8Ø
6Ø43 IF N=29 AND OB(N)=SPIELER THEN PR
INT "O.K.":OB(N)=-2:WERTUNG=WERTUNG+1Ø
:GOTO 1Ø8Ø
6Ø44 IF N=38 AND OB(N)=SPIELER THEN PR
INT "O.K.":OB(N)=-2:WERTUNG=WERTUNG+1Ø
:GOTO 1Ø8Ø
69ØØ IF N=9 AND (OB(7)=SPIELER OR OB(7
)=-1) THEN MØ$="BEI DER BERUEHRUNG IST
 DER SPRENGSTOFF EXPLODIERT !":GOTO 45
ØØ
691Ø IF N=2Ø AND SPIELER=22 THEN PRINT
 "WENN ERST MAL DIE GIER ERWACHT, DER
  TOD AUCH OFT GESCHAEFTE MACHT.":GOTO
 1Ø8Ø
6999 PRINT "SO ETWAS SEHE ICH HIER NIC
HT !":GOTO 1Ø8Ø
7ØØØ IF N=16 AND OB(18)=-1 THEN PRINT
"O.K.":OB(18)=SPIELER:GOTO 1Ø8Ø
7ØØ5 IF OB(N)<>-1 THEN PRINT "SO ETWAS
 HABE ICH DOCH GAR NICHT !":GOTO 1Ø8Ø
7Ø1Ø IF N=6 AND SPIELER=5 THEN OB(6)=Ø
:FL(3)=-1:PRINT M4$:PRINT M5$:OB(14)=Ø
:GOTO 1Ø8Ø
7Ø2Ø IF N=5 AND SPIELER=5 THEN OB(5)=Ø
:FL(3)=-1:PRINT M4$:PRINT M5$:OB(14)=Ø
:GOTO 1Ø8Ø
79ØØ OB(N)=SPIELER:PRINT "O.K.":GOTO 1
Ø8Ø
8ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN PRINT "SO ETWAS IST HIER NICHT.":G
OTO 1Ø8Ø
8ØØ5 IF N=4 AND SPIELER=3 THEN PRINT "
DIE HUETTE STAND BEREITS OFFEN.":GOTO
```

```
1Ø8Ø
8Ø1Ø IF N=5 THEN PRINT "O.K.":GOTO 1Ø8
Ø
8Ø2Ø IF N=11 AND  NOT FL(I) THEN PRINT
 "DAS LAESST DIE KETTE NICHT ZU !":GOT
O 1Ø8Ø
8Ø25 IF N=11 AND FL(1) THEN PRINT "O.K
.- DER DECKEL KLAPPT NACH HINTEN.":FL(
2)=-1:GOTO 1Ø8Ø
8Ø3Ø IF N=23 THEN PRINT "O.K.":GOTO 1Ø
8Ø
8Ø35 IF N=29 THEN PRINT "O.K.":GOTO 1Ø
8Ø
8Ø4Ø IF N=32 THEN PRINT "TUT MIR LEID
!":PRINT "ICH BIN NICHT FRANKENSTEIN."
:GOTO 1Ø8Ø
8Ø45 IF N=36 THEN PRINT "WIE ?":GOTO 1
Ø8Ø
8999 PRINT "ICH VERSTEHE NICHT, WAS DU
 MEINST.":GOTO 1Ø8Ø
9ØØØ IF OB(N)<>SPIELER AND OB(N)<>-1 T
HEN GOTO 99ØØ
9Ø1Ø IF N=16 AND SPIELER=4 THEN PRINT
"DIE KETTE ZERSPRINGT.":FL(1)=-1:GOTO
1Ø8Ø
9Ø2Ø IF N=19 THEN PRINT "WIE UND WOZU
?":GOTO 1Ø8Ø
9Ø3Ø IF N=21 THEN PRINT "WIE UND WOZU
?":GOTO 1Ø8Ø
99ØØ IF N=16 AND OB(18)=-1 AND SPIELER
=4 THEN PRINT "DIE KETTE ZERSPRINGT.":
FL(1)=-1:GOTO 1Ø8Ø
9999 PRINT "ICH VERSTEHE NICHT, WAS DU
 MEINST !":GOTO 1Ø8Ø
1ØØØØ IF N=18 AND SPIELER=4 AND OB(18)
=-1 THEN PRINT "DIE KETTE ZERSPRINGT."
:FL(1)=-1:GOTO 1Ø8Ø
1ØØ1Ø IF N=18 AND SPIELER=4 AND OB(18)
=-1 THEN PRINT "ICH HABE KEINE EISENS
TANGE.":GOTO 1Ø8Ø
1ØØ2Ø IF N=36 AND OB(22)=-1 THEN PRINT
```

```
 "O.K.":DURCHGANG(22,1)=23:GOTO 1Ø8Ø
1ØØ3Ø IF N=21 AND (OB(N)=SPIELER OR OB
(N)=-1) THEN PRINT "O.K.":FL(4)=-1:GOT
O 1Ø8Ø
1ØØ4Ø IF N=27 AND OB(22)=-1 THEN PRINT
 "O.K.":DURCHGANG(1Ø,Ø)=12:GOTO 1Ø8Ø
1ØØ5Ø IF N=27 AND OB(22)<>-1 THEN PRIN
T "WOMIT ?":GOTO 1Ø8Ø
1Ø999 PRINT "ICH VERSTEHE NICHT, WAS D
U MEINST !":GOTO 1Ø8Ø
11ØØØ IF OB(43)<>-1 THEN PRINT "ICH HA
BE NICHTS ZUM ZUENDEN.":GOTO 1Ø8Ø
11Ø1Ø IF N=8 AND LZ<=Ø THEN PRINT "IN
DER LAMPE IST KEIN OEL MEHR.":GOTO 1Ø8
Ø
11Ø2Ø IF N=42 AND (OB(43)=-1 OR LI=-1)
 THEN PRINT "ZZZIISCH":FL(7)=-1:EX=ZUG
+3:OB(N)=Ø
11Ø21 IF N=42 AND (OB(43)=-1 OR LI=-1)
 THEN DURCHGANG(3Ø,2)=31:DURCHGANG(31,
1)=3Ø:GOTO 1Ø8Ø
11Ø3Ø IF N=23 AND OB(23)=-1 THEN PRINT
 "O.K. - DIE LAMPE BRENNT.":LM=L1:LW=1
:GOTO 1Ø8Ø
11Ø4Ø IF N=43 THEN PRINT "O.K. - DER D
OCHT GLIMMT.":GOTO 1Ø8Ø
11998 PRINT "ICH VERSTEHE NICHT, WAS D
U MEINST.":GOTO 1Ø8Ø
12ØØØ REM FUELLE
12Ø1Ø IF N=23 AND (FL(6) AND OB(23)=-1
) THEN LM=6Ø:FL(6)=Ø:LW=Ø:PRINT "O.K."
:GOTO 1Ø8Ø
12Ø2Ø IF N=5 AND SPIELER=17 THEN FL(6)
=-1:PRINT "DIE FLASCHE IST VOLL.":GOTO
 1Ø8Ø
12Ø3Ø IF N=43 AND OB(43)=-1 THEN PRINT
 "DER DOCHT GLIMMT.":GOTO 1Ø8Ø
12998 PRINT "ICH VERSTEHE NICHT, WAS D
U MEINST.":GOTO 1Ø8Ø
13ØØØ REM BETRETE
13Ø1Ø IF N=13 AND SPIELER=5 THEN SPIEL
```

```
ER=12:PRINT "O.K.":GOTO 1Ø8Ø
13998 PRINT "ICH VERSTEHE NICHT, WAS D
U MEINST.":GOTO 1Ø8Ø
14ØØØ IF N=23 AND OB(N)=-1 THEN L1=LM-
LW:PRINT "O.K.":GOTO 1Ø8Ø
14998 PRINT "ICH VERSTEHE NICHT, WAS D
U MEINST.":GOTO 1Ø8Ø
15ØØØ IF N=21 AND SPIELER=9 THEN PRINT
 "O.K.":OB(N)=9:DURCHGANG(9,6)=18:DURC
HGANG(18,5)=9:GOTO 1Ø8Ø
15998 PRINT "ICH VERSTEHE NICHT, WAS D
U MEINST.":GOTO 1Ø8Ø
```

SPACE MISSION

Bitte lassen Sie sich durch den Reiz eines Grafikadventures nicht dazu verleiten, dieses Programm einzugeben, sofern Sie absoluter Newcomer auf diesem Gebiet sind und Ihre Programmierkenntnisse darüber hinaus nicht so weit fortgeschritten sein sollten, daß Sie sich die Antworten auf die sich Ihnen sehr schnell stellenden Probleme aus dem Listing holen können.

Denn im Gegensatz zu den übrigen Adventurespielen, bei denen gefährliche Situationen erst durch Ihr Walten und Schalten entstehen, haben Sie bei Space Mission nach der Eingabe von RUN nur neun (9!) Spielzüge Zeit, um Ihr Leben zu retten.

Und es wäre doch schade, wenn Sie die Lust an Spielprogrammen dieser Art verlieren, nur weil Sie kein Erfolgserlebnis haben !

Anmerkungen zum Listing "Space Mission"

Das im Buch abgedruckte Listing wurde wie folgt korrigiert:

145 -> MØ$ auf 6Ø gesetzt, da max. 53 Zeichen zugewiesen.

5ØØ9 -> Sprung korrigiert.

```
5Ø GRAPHICS Ø:SETCOLOR 2,1,Ø:SETCOLOR
4,1,Ø
55 PRINT "SPACE MISSION".? "(C) 1984 B
Y J.W."
115 AR=3Ø:AO=5Ø:AF=11:SP=11:IMAX=2:ZUG
=Ø
14Ø DIM RA$(35),DU(AR,6),RI$(36),EINGA
BE$(2Ø),T$(3Ø),OB(AO),DZ$(5Ø),ZE$(3),F
L(AF),OK$(4),TM$(6)
145 DIM VERBEN$(97),EV$(1Ø),EO$(1Ø),OB
JEKTE$(2Ø1),MØ$(6Ø):OK$="O.K.":TRAP 1Ø
8Ø
155 VERBEN$="UNTERSUCHENIMM      OEFFN
E    DRUECKE   LEGE      TRAGE"
16Ø VERBEN$(LEN(VERBEN$)+1)="     FUEL
LE    SETZE     BENUTZE   TAUSCHE   "
165 OBJEKTE$="PULTVIDESCHATERMHYPETELE
ANTIEINSEINSGEHACOMPSTECMODUINBUKABERO
HRSCHI"
17Ø OBJEKTE$(LEN(OBJEKTE$)+1)="LANDWAN
DMONITUERAUTOKNOPPRITTISCMOVETANKSAUER
AUMZEHNSTOPSCHRLEIT"
175 OBJEKTE$(LEN(OBJEKTE$)+1)="TUERSET
CMODUMEDIKOORFUNKMAGNSCHO*SD*VIEWBETTS
TATSELOTRANKONTPASSKART":GOTO 9ØØ
2Ø1 PRINT "IN DER HAUPTZENTRALE.":RETU
RN
2Ø2 PRINT "IM TRANSMITTERRAUM.":RETURN

2Ø3 PRINT "IN DER FUNKZENTRALE":RETURN

2Ø4 PRINT "AM PILOTENPULT.":RETURN
2Ø5 PRINT "*":RETURN
2Ø6 PRINT "IN DER ZUGANGSKONTROLLE.":R
ETURN
2Ø7 PRINT "IN EINEM ZWISCHENDECK.":RET
URN
2Ø8 PRINT "IN DER ASTRONOMISCHEN ABT."
:RETURN
2Ø9 PRINT "IN DER OBEREN POLKUPPEL.":R
ETURN
```

```
21Ø PRINT "IN DER MEDOSTATION":RETURN
211 PRINT "IN DER KANTINE.":RETURN
212 PRINT "IM TRANSMITTERRAUM.":RETURN

213 PRINT "IN EINEM FRACHTRAUM.":RETUR
N
214 PRINT "IM MASCHINENRAUM.":RETURN
215 PRINT "IN EINER LAGERHALLE.":RETUR
N
216 PRINT "VOR EINEM SICHERHEITSSCHOTT
.":RETURN
217 PRINT "AUF EINEM GANG.":RETURN
218 PRINT "AUF EINEM GANG.":RETURN
219 PRINT "AUF EINEM GANG.":RETURN
22Ø PRINT "IM ERSATZTEILLAGER.":RETURN

221 PRINT "IN EINEM ZWISCHENDECK.":RET
URN
222 PRINT "IM ANTIGRAVSCHACHT.":RETURN

223 PRINT "IM ANTIGRAVSCHACHT.":RETURN

224 PRINT "IM ANTIGRAVSCHACHT.":RETURN

225 PRINT "IM ANTIGRAVSCHACHT.":RETURN

226 PRINT "IM ANTIGRAVSCHACHT.":RETURN

227 PRINT "IM ANTIGRAVSCHACHT.":RETURN

23Ø PRINT "IN EINEM MANNSCHAFTSRAUM.":
RETURN
3Ø1 T$="DAS ZENTRALSCHALTPULT":RETURN
3Ø2 T$="VIDEOSCHIRM":RETURN
3Ø3 T$="NICHTS BESONDERES":RETURN
3Ø4 T$="TERMINAL":RETURN
3Ø5 T$="HYPERKOM":RETURN
3Ø6 T$="TELEKOM":RETURN
3Ø7 T$="":RETURN
3Ø8 T$="EINSTIEG":RETURN
3Ø9 T$="":RETURN
```

```
31Ø T$="":RETURN
311 T$="COMPUTER":RETURN
312 T$="STECKMODUL":RETURN
313 T$="MODULE":RETURN
314 T$="INBUSSCHLUESSEL":RETURN
315 T$="KABELBAEUME":RETURN
316 T$="ROHRE":RETURN
317 T$="SCHILD":RETURN
318 T$="":RETURN
319 T$="WANDSCHRANK":RETURN
32Ø T$="":RETURN
321 T$="TUER":RETURN
322 T$="DEN AUTOPILOTEN":RETURN
323 T$="EINEN SCHALTKNOPF":RETURN
324 T$="EINE PRITSCHE":RETURN
325 T$="NICHTS BESONDERES":RETURN
326 T$="":RETURN
327 T$="TANKS":RETURN
328 T$="SAUERSTOFFFLASCHE":RETURN
329 T$="EINEN RAUMANZUG":RETURN
33Ø T$="1Ø-ER SCHLUESSEL":RETURN
331 T$="":RETURN
332 T$="SCHRAUBENDREHER":RETURN
333 T$="LEITSTAND":RETURN
334 T$="EINE ROTE TUER":RETURN
335 T$="":RETURN
336 T$="EIN MODUL":RETURN
337 T$="MEDIKAMENTE":RETURN
338 T$="":RETURN
339 T$="FUNKGERAETE":RETURN
34Ø T$="MAGNETKARTE":RETURN
341 T$="NICHTS BESONDERES":RETURN
342 T$="":RETURN
343 T$="":RETURN
344 T$="":RETURN
345 T$="":RETURN
346 T$="":RETURN
347 T$="":RETURN
348 T$="NICHTS BESONDERES":RETURN
349 T$="":RETURN
35Ø T$="KARTONS":RETURN
```

```
401 DATA 1,1,30,12,0,0,0,0,0,7,0,-1,0,
0,7,9,8,0,10,0,9,0,9,10,11,0,21,0,0,0,
0,0,14,14,0,0,0,0,0,3,0,16,0,0,30,0,0,0
402 DATA 6,0,15,3,3,16,0,0,0,6,0,0,0,0
,0,1,1,0,0,0,0,3,0,0,6,0,0,0,0,0,0,0,0
,0,2,0,0,0,0,0,0,23,0,0,0,0,0,22,0,0
403 DATA 0,0,0,0,0,0,0,22,0,0,0,22,0,0,0
,0,30,24,0,0,12,12,18,13,0,0,0,0,12,0,
0,0,0,0,17,0,0,0,0,2,0,0,0,0,0
404 DATA 0,24,0,0,0,19,19,26,14,0,0,18
,18,27,12,0,0,17,17,20,0,0,0,0,0,0,19,
0,0,0,0,25,0,0,0,0,0,10,8,9,23
405 DATA 0,0,0,0,22,24,0,0,11,16,23,0,
0,0,0,0,24,26,0,0,0,17,25,27,0,0,0,18,
26,0,0,0,0,0,0,0,0,0,0,0,0,0
406 DATA 0,0,0,11,0,0
601 PRINT "ICH SEHE NICHTS BESONDERES.
":RETURN
602 PRINT "SO STARK BIN ICH NICHT.":RE
TURN
603 PRINT "WIE STELLST DU DIR DAS VOR
?":RETURN
604 PRINT "DAS HABE ICH DOCH SCHON !":
RETURN
605 PRINT "ICH VERSTEHE NICHT, WAS DU
MEINST.":RETURN
900 FOR I=1 TO AO:READ LAGEORT:OB(I)=L
AGEORT:NEXT I
910 FOR I=1 TO AF:FL(I)=0:NEXT I
920 FOR RA=1 TO AR:FOR RI=1 TO 6:READ
Z:DU(RA,RI)=Z:NEXT RI:NEXT RA
1000 GRAPHICS 8:SETCOLOR 2,1,0:GOTO 10
40
1021 T$="N, ":RETURN
1022 T$="S, ":RETURN
1023 T$="W, ":RETURN
1024 T$="O, ":RETURN
1025 T$="OB, ":RETURN
1026 T$="U, ":RETURN
1040 T$=""
1050 ZE$=CHR$(30):ZE$(LEN(ZE$)+1)=CHR$
```

```
(3Ø):ZE$(LEN(ZE$)+1) ="."
1Ø8Ø PRINT :ZUG=ZUG+1:IF ALT<>SP THEN
GRAPHICS 8:SETCOLOR 2,1,Ø:GOSUB 3ØØØØ
11ØØ POSITION 2,Ø:PRINT "ICH BIN ";:GO
SUB 2ØØ+SP:ALT=SP
111Ø DZ$="ICH SEHE: "
112Ø FOR I=1 TO AO
113Ø IF OB(I)<>SP THEN GOTO 117Ø
114Ø GD=-1:GOSUB 3ØØ+I:T$(LEN(T$)+1)="
, "
115Ø IF LEN(DZ$)+LEN(T$)>=38 THEN PRIN
T DZ$:DZ$=T$(1,LEN(T$)):T$="":GOTO 117
Ø
116Ø DZ$(LEN(DZ$)+1)=T$:T$=""
117Ø NEXT I
118Ø IF  NOT GD THEN DZ$(LEN(DZ$)+1)="
NICHTS BESONDERES. "
119Ø DZ$(LEN(DZ$)+1)=ZE$
12ØØ PRINT DZ$:GD=Ø:DZ$=""
123Ø FOR RI=1 TO 6
124Ø IF DU(SP,RI)=Ø THEN GOTO 128Ø
125Ø GOSUB 1Ø2Ø+RI:GD=-1
127Ø DZ$(LEN(DZ$)+1)=T$:T$=""
128Ø NEXT RI
129Ø IF  NOT GD THEN DZ$(LEN(DZ$)+1)="
NIRGENDWO"
13ØØ PRINT DZ$;
135Ø IF FL(1)=Ø THEN PRINT "ALARM!";:I
F ZUG>9 THEN MØ$="EIN ASTEROID HAT UNS
 GERAMMT.":GOTO 45ØØ
139Ø INPUT EINGABE$
14ØØ IF EINGABE$="N" AND DU(SP,1)<>Ø T
HEN SP=DU(SP,1):PRINT OK$:GOTO 1Ø8Ø
141Ø IF EINGABE$="S" AND DU(SP,2)<>Ø T
HEN SP=DU(SP,2):PRINT OK$:GOTO 1Ø8Ø
142Ø IF EINGABE$="W" AND DU(SP,3)<>Ø T
HEN SP=DU(SP,3):PRINT OK$:GOTO 1Ø8Ø
143Ø IF EINGABE$="O" AND DU(SP,4)<>Ø T
HEN SP=DU(SP,4):PRINT OK$:GOTO 1Ø8Ø
144Ø IF EINGABE$="OB" AND DU(SP,5)<>Ø
THEN SP=DU(SP,5):PRINT OK$:GOTO 1Ø8Ø
```

```
145Ø IF EINGABE$="U" AND DU(SP,6)<>Ø T
HEN SP=DU(SP,6):PRINT OK$:GOTO 1Ø8Ø
146Ø IF LEN(EINGABE$)<3 THEN PRINT "DA
HIN FUEHRT KEIN WEG !":GOTO 1Ø8Ø
147Ø IF LEN(EINGABE$)>8 THEN GOTO 2ØØØ
15ØØ IF EINGABE$(1,3)<>"INV" THEN GOTO
 16ØØ
15Ø5 PRINT "INVENTORY:"
151Ø FOR I=1 TO AO
1515 IF OB(1)=-1 THEN GOSUB 3ØØ+I:PRIN
T T$;" ";
152Ø NEXT I
1525 GOTO 1Ø8Ø
16ØØ IF EINGABE$(1,3)<>"SAV" THEN GOTO
 17ØØ
16Ø5 PRINT "NAME ";:INPUT EINGABE$: IF
 LEN(EINGABE$)>8 THEN GOTO 16Ø5
161Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
=".DAT"
1615 T$(LEN(T$)+1)=EINGABE$
162Ø OPEN #1,8,Ø,T$
1625 PRINT #1,SP
163Ø FOR I=1 TO AO:PRINT #1,OB(I):NEXT
 I
1635 FOR RA=1 TO AR:FOR RI=1 TO 6:PRIN
T #1,DU(RA,RI):NEXT RI:NEXT RA
1655 FOR I=1 TO AF:PRINT #1,FL(I):NEXT
 I
166Ø CLOSE #1:PRINT OK$:GOTO 1Ø8Ø
17ØØ IF EINGABE$(1,3)<>"LOA" THEN GOTO
 19ØØ
17Ø5 PRINT "WELCHES SPIEL ";:INPUT EIN
GABE$:IF LEN(EINGABE$)>8 THEN 17Ø5
171Ø T$="D:":EINGABE$(LEN(EINGABE$)+1)
=".DAT"
1715 T$(LEN(T$)+1)=EINGABE$
172Ø OPEN #1,4,Ø,T$
1725 INPUT #1,SP
173Ø FOR I=1 TO AO:INPUT #1,LO:OB(I)=L
O:NEXT I
1735 FOR RA=1 TO AR
```

```
174Ø FOR RI=1 TO 6
1745 INPUT #1,ZIEL:DU(RA,RI)=ZIEL
175Ø NEXT RI
1755 NEXT RA
176Ø FOR I=1 TO AF:INPUT #1,ZIEL:FL(I)
=ZIEL:NEXT I
1765 CLOSE #1:PRINT OK$:GOTO 1Ø8Ø
19ØØ IF EINGABE$(1,3)<>"INS" THEN GOTO
 191Ø
19Ø5 GOSUB 8ØØ:GOTO 1Ø8Ø
191Ø IF EINGABE$(1,3)<>"END" THEN GOTO
 2ØØØ
1915 GRAPHICS Ø:PRINT "DER AUTOR WUENS
CHT IHNEN MEHR GLUECK":PRINT "FUERS NA
ECHSTE MAL !":PRINT :PRINT :PRINT :END

2ØØØ LN=LEN(EINGABE$)
2Ø1Ø FOR I=1 TO LN
2Ø2Ø IF EINGABE$(I,I)<>" " THEN NEXT I
2Ø3Ø EV$=EINGABE$(1,I)
2Ø4Ø IF LEN(EV$)=LN-1 THEN GOTO 2Ø9Ø
2Ø5Ø EO$=EINGABE$(I+1,LN)
2Ø6Ø VN=Ø:N=Ø
2Ø7Ø EV$=EV$(1,4)
2Ø8Ø EO$=EO$(1,4)
2Ø9Ø FOR I=1 TO LEN(VERBEN$) STEP 1Ø
21ØØ VN=VN+1
211Ø IF VERBEN$(I,I+3)=EV$ THEN GOTO 2
14Ø
212Ø NEXT I
213Ø PRINT "ICH VERSTEHE DAS VERB NICH
T !":GOTO 1Ø8Ø
214Ø FOR I=1 TO LEN(OBJEKTE$)-2 STEP 4
215Ø N=N+1
216Ø IF OBJEKTE$(I,I+3)=EO$ THEN 22ØØ
217Ø NEXT I
218Ø PRINT "ICH VERSTEHE DAS OBJEKT NI
CHT !":GOTO 1Ø8Ø
22ØØ ON VN GOTO 5ØØØ,6ØØØ,7ØØØ,8ØØØ,9Ø
ØØ,1ØØØØ,11ØØØ,12ØØØ,13ØØØ,14ØØØ
233Ø RETURN
```

```
45ØØ GRAPHICS Ø
451Ø PRINT "":PRINT :PRINT MØ$
452Ø PRINT :PRINT "ICH BIN TOD !":PRIN
T
453Ø PRINT "SOLL ICH ES NOCH EINMAL VE
RSUCHEN ";:INPUT EINGABE$
454Ø IF EINGABE$(1,1)="J" THEN RUN
455Ø GOTO 196Ø
48ØØ GRAPHICS Ø
481Ø PRINT "HERZLICHEN GLUECKWUNSCH !"
482Ø PRINT :PRINT :PRINT "SIE HABEN DI
E IHNEN GESTELLTE AUFGABE"
483Ø PRINT :PRINT "GELOEST, UND DUERFE
N SICH AN EINEM"
484Ø PRINT :PRINT "ANDEREM ADVENTURE V
ERSUCHEN."
485Ø PRINT :PRINT :PRINT :END
5ØØØ IF OB(N)<>SP AND OB(N)<>-1 THEN G
OTO 59ØØ
5ØØ5 IF N=1 THEN PRINT "DAMIT BEDIENE
ICH U.A. DEN AUTO-":PRINT "PILOTEN.":O
B(22)=1:GOTO 1Ø8Ø
5ØØ6 IF N=2 AND FL(1)=Ø THEN PRINT "DE
R SCHIRM ZEIGT EINEN ASTEROIDEN":GOTO
1Ø8Ø
5ØØ7 IF N=2 AND FL(1) THEN PRINT "ICH
SEHE NICHTS BESONDERES.":GOTO 1Ø8Ø
5ØØ9 IF N=4Ø THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø1Ø IF N=6 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø13 IF N=25 AND OB(4Ø)=Ø THEN PRINT "
DARUNTER LAG EINE MAGNETKARTE.":OB(4Ø)
=11:GOSUB 3ØØØØ:GOTO 1Ø8Ø
5Ø14 IF N=8 AND SP=23 AND FL(7)=Ø THEN
 PRINT "ER IST VERSCHLOSSEN.":GOTO 1Ø8
Ø
5Ø15 IF N=8 AND SP=25 AND FL(8)=Ø THEN
 PRINT "ER IST VERSCHLOSSEN.":GOTO 1Ø8
Ø
5Ø16 IF N=8 AND SP=23 AND FL(7)=-1 THE
N PRINT "ER FUEHRT IN EIN ZWISCHENDECK
.":GOTO 1Ø8Ø
```

```
5Ø17 IF N=8 AND SP=25 AND FL(8)=-1 THE
N PRINT "ER FUEHRT IN EIN ZWISCHENDECK
.":GOTO 1Ø8Ø
5Ø19 IF N=39 THEN GOSUB 6Ø1:OB(5)=SP:O
B(6)=SP:OB(39)=Ø:GOTO 1Ø8Ø
5Ø2Ø IF N=5 THEN GOTO 2ØØØØ
5Ø21 IF N=44 AND OB(29)=Ø THEN PRINT "
DARUNTER LIEGT MEIN RAUMANZUG.":GOTO 1
Ø8Ø
5Ø25 IF N=4 THEN PRINT "DIE ZIFFERN VO
N Ø-9 SIND VORHANDEN.":GOTO 1Ø8Ø
5Ø26 IF N=1Ø AND OB(32)<>-1 THEN PRINT
 "ICH KANN ES NICHT OEFFNEN.":GOTO 1Ø8
Ø
5Ø27 IF N=1Ø AND OB(32)=-1 THEN PRINT
"DARIN IST DER ZENTRALRECHNER.":OB(11)
=7:OB(1Ø)=Ø:GOTO 1Ø8Ø
5Ø28 IF N=11 THEN PRINT "ICH SEHE ZAHL
REICHE MODULE.":OB(13)=SP:OB(15)=Ø:GOT
O 1Ø8Ø
5Ø29 IF N=13 THEN PRINT "ES SIND STECK
MODULE.":GOTO 1Ø8Ø
5Ø3Ø IF N=12 THEN PRINT "... EIN NORMA
LES COMPUTERMODUL.":GOTO 1Ø8Ø
5Ø31 IF N=14 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø32 IF N=15 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø33 IF N=16 THEN PRINT "SIE GEHOEREN
ZUM BELUEFTUNGSSYSTEM.":GOTO 1Ø8Ø
5Ø34 IF N=17 THEN PRINT "NUR EIN WEGWE
ISER.":GOTO 1Ø8Ø
5Ø35 IF N=2Ø THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø36 IF N=33 THEN GOSUB 6Ø1:OB(3Ø)=SP:
GOTO 1Ø8Ø
5Ø37 IF N=22 THEN PRINT "ICH SEHE TAST
EN: LAND MOVE STOP SETC":OB(N)=Ø:FOR I
=1 TO 2ØØØ:NEXT I:GOTO 1Ø8Ø
5Ø38 IF N=23 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø39 IF N=24 AND OB(32)=Ø THEN PRINT "
DAHINTER LIEGT EIN SCHRAUBENDREHER.":G
OTO 1Ø8Ø
5Ø4Ø IF N=32 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
```

```
5Ø41 IF N=27 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø42 IF N=28 AND FL(4)=Ø THEN PRINT "S
IE IST LEER.":GOTO 1Ø8Ø
5Ø43 IF N=28 AND FL(4)=-1 THEN PRINT "
SIE IST GEFUELLT.":GOTO 1Ø8Ø
5Ø44 IF N=29 THEN PRINT "ES IST MEINER
.":GOTO 1Ø8Ø
5Ø45 IF N=3Ø THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5Ø46 IF N-48 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
52ØØ IF N=24 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5899 GOSUB 6Ø1:GOTO 1Ø8Ø
59ØØ REM
59Ø1 IF N=7 AND SP=24 THEN GOSUB 6Ø1:G
OTO 1Ø8Ø
59Ø2 IF N=7 AND SP=23 THEN FL(5)=-1:AL
T=Ø:OB(8)=SP:GOTO 1Ø8Ø
59Ø3 IF N=7 AND SP=24 THEN GOSUB 6Ø1:G
OTO 1Ø8Ø
59Ø4 IF N=7 AND SP=25 THEN FL(6)=-1:OB
(9)=SP:ALT=1:GOTO 1Ø8Ø
59Ø5 IF N=7 AND SP=26 THEN GOSUB 6Ø1:G
OTO 1Ø8Ø
59Ø6 IF N=7 AND SP=27 THEN GOSUB 6Ø1:G
OTO 1Ø8Ø
59Ø7 IF N=7 AND SP=22 THEN GOSUB 6Ø1:G
OTO 1Ø8Ø
591Ø IF N=2Ø AND SP=8 THEN ? "PIA III
IST MIT 151Ø64 ANGEGEBEN.":GOTO 1Ø8Ø
5916 IF N=25 AND OB(4Ø)=-1 THEN GOTO 1
Ø8Ø
592Ø IF N=32 THEN GOSUB 6Ø1:GOTO 1Ø8Ø
5922 IF N=8 AND SP=25 AND DU(25,4)=Ø T
HEN PRINT "ER IST VERSCHLOSSEN.":GOTO
1Ø8Ø
5923 IF N=47 AND SP=2 THEN PRINT "DAS
GLEICHE MODELL !":GOTO 1Ø8Ø
5924 IF N=4 AND SP=2 THEN PRINT "DAS G
LEICHE MODELL !":GOTO 1Ø8Ø
5925 IF N=2Ø AND SP=2 THEN PRINT "ES
SIND DIE MEDIKAMENTE.":GOTO 1Ø8Ø
599Ø PRINT "SO ETWAS SEHE ICH HIER NIC
```

```
HT !":GOTO 1Ø8Ø
6ØØØ IF OB(N)<>SP AND OB(N)<>-1 THEN G
OTO 69ØØ
6ØØ5 IF N=4Ø THEN PRINT OK$:OB(N)=-1:G
OSUB 3ØØØØ:GOTO 1Ø8Ø
6Ø1Ø IF N=12 AND OB(12)=-1 THEN GOSUB
6Ø4:GOTO 1Ø8Ø
6Ø11 IF N=12 THEN OB(12)=-1:PRINT OK$:
GOTO 1Ø8Ø
6Ø12 IF N=13 THEN PRINT "ICH HABE EIN
MODUL.":OB(36)=-1:GOTO 1Ø8Ø
6Ø13 IF N=14 OR N=28 THEN PRINT OK$:OB
(N)=-1:GOTO 1Ø8Ø
6Ø14 IF (N=3Ø OR N=5Ø) THEN PRINT OK$:
OB(N)=-1:GOTO 1Ø8Ø
68ØØ IF N=32 THEN PRINT OK$:OB(N)=-1:G
OTO 1Ø8Ø
689Ø IF N=SP THEN GOSUB 6Ø2:GOTO 1Ø8Ø
69ØØ REM
691Ø IF N=29 THEN PRINT OK$:OB(N)=-1:G
OTO 1Ø8Ø
6915 IF N=2Ø AND SP=15 THEN PRINT OK$:
OB(5Ø)=-1:GOTO 1Ø8Ø
692Ø IF N=32 AND OB(N)=Ø THEN OB(32)=-
1:PRINT OK$:GOTO 1Ø8Ø
699Ø PRINT "SO ETWAS SEHE ICH HIER NIC
HT.":GOTO 1Ø8Ø
7ØØØ IF N=41 AND OB(4Ø)=-1 THEN PRINT
OK$:DU(16,4)=1:ALT=2:GOTO 1Ø8Ø
7ØØ5 IF N=41 AND OB(4Ø)<>-1 THEN PRINT
 "DAS GEHT IM MOMENT NICHT.":GOTO 1Ø8Ø
7Ø1Ø IF N=8 AND OB(14)<>-1 THEN PRINT
"ICH HABE NICHTS ZUM OEFFNEN.":GOTO 1Ø
8Ø
7Ø11 IF N=8 AND SP=23 AND OB(14)=-1 TH
EN PRINT OK$:DU(23,4)=7:FL(7)=-1:GOTO
1Ø8Ø
7Ø15 IF N=23 AND SP=9 THEN MØ$="ENTWEI
CHENDER SAUERSTOFF ENTREISST    MICH I
NS ALL.":GOTO 45ØØ
7Ø2Ø IF N=9 AND SP=14 THEN MØ$="EINE F
```

```
UNKENENTLADUNG AUS DEM REAKTOR- RAUM T
RAF MICH.":GOTO 45ØØ
7Ø99 GOSUB 6Ø5:GOTO 1Ø8Ø
8ØØØ REM
8Ø1Ø IF N=46 AND FL(2)=-1 THEN PRINT "
KLICK":DU(24,6)=25:GOTO 1Ø8Ø
8Ø12 IF N=46 AND FL(2)=Ø THEN PRINT "N
ICHTS GESCHIEHT.":GOTO 1Ø8Ø
8Ø2Ø IF N=45 THEN GOTO 2Ø2ØØ
8Ø21 IF N=43 THEN GOTO 2Ø3ØØ
8Ø22 IF N=42 THEN MØ$="DAS WAR DIE SEL
BSTVERNICHTUNG !":GOTO 45ØØ
8Ø24 IF N=18 THEN MØ$="DER AUTOPILOT L
ANDETE DAS SCHIFF IN   DER SONNE.":GOT
O 45ØØ
8Ø25 IF N=26 AND TM$="151Ø63" THEN MØ$
="DIE FEHLFUNKTION! FALSCHE KOORDINATE
N.":GOTO 45ØØ
8Ø26 IF N=26 AND FL(I)=Ø THEN PRINT "D
AS SCHIFF WEICHT DEM ASTEROIDEN AUS.";
:FL(1)=-1:GOTO 1Ø8Ø
8Ø27 IF N=26 AND FL(1)=-1 AND FL(1Ø)=Ø
 THEN MØ$="KEINE KURSDATEN GESETZT - Z
IEL WAR EINE SONNE.":GOTO 45ØØ
8Ø28 IF N=26 AND FL(1Ø)=-1 AND OB(5Ø)<
>13 THEN MØ$="GELYNCHT, WEIL OHNE MEDI
KAMENTE ANGE- KOMMEN.":GOTO 45ØØ
8Ø29 IF N=31 THEN FL(1)=Ø:PRINT OK$:GO
TO 1Ø8Ø
8Ø3Ø IF N=35 THEN PRINT "KOORDINATEN "
;:INPUT TM$:IF TM$="151Ø64" THEN FL(1Ø
)=-1:GOTO 1Ø8Ø
8Ø31 IF N=35 THEN GOTO 1Ø8Ø
8Ø32 IF N=26 AND FL(1Ø)=-1 AND OB(5Ø)=
13 THEN GOTO 48ØØ
8115 IF N=23 AND SP=9 THEN MØ$="ENTWEI
CHENDER SAUERSTOFF REISST MICH  INS AL
L.":GOTO 45ØØ
899Ø PRINT "SO EINEN KNOPF SEHE ICH HI
ER NICHT.":GOTO 1Ø8Ø
9ØØØ IF OB(N)<>-1 THEN GOTO 99ØØ
```

```
9Ø1Ø PRINT OK$:OB(N)=SP:GOTO 1Ø8Ø
99ØØ IF N=13 AND OB(36)=-1 THEN OB(36)
=SP:PRINT OK$:GOTO 1Ø8Ø
9915 IF N=2Ø AND OB(5Ø)=-1 AND  NOT (S
P=15 OR SP=13) THEN PRINT OK$:OB(5Ø)=S
P:FL(11)=-1:GOTO 1Ø8Ø
1ØØØØ IF N=29 THEN PRINT "ICH HABE DEN
 RAUMANZUG ANGELEGT.":OB(29)=-2:OB(14)
=-1:GOTO 1Ø8Ø
1ØØ1Ø GOTO 6ØØØ
11ØØØ IF N=28 AND SP<>9 THEN PRINT "WO
MIT ?":GOTO 1Ø8Ø
11Ø1Ø IF N=28 AND SP=9 AND OB(3Ø)<>-1
THEN PRINT "DAS GEHT IM MOMENT NICHT."
:GOTO 1Ø8Ø
11Ø2Ø IF N=28 AND SP=9 AND OB(3Ø)=-1 T
HEN PRINT OK$:FL(4)=-1:GOTO 1Ø8Ø
119ØØ GOSUB 6Ø5:GOTO 1Ø8Ø
12ØØØ IF N=38 AND SP=12 THEN INPUT TM$
:GOTO 1Ø8Ø
12ØØ5 IF N=38 AND SP=2 THEN INPUT TM$:
GOTO 1Ø8Ø
12Ø1Ø IF N=49 AND SP=6 THEN INPUT TM$:
IF TM$="22Ø559" THEN DU(2,1)=15:SP=15:
GOTO 1Ø8Ø
12Ø2Ø IF N=49 AND SP=6 AND TM$<>"22Ø55
9" THEN MØ$="DIE ABWEHRSYSTEME SPRACHE
N AN.":GOTO 45ØØ
13ØØØ IF N=47 AND  NOT (SP=12 OR SP=2)
 THEN PRINT "ICH SEHE KEINEN TRANSMITT
ER.":GOTO 1Ø8Ø
13Ø1Ø IF N=47 AND SP=12 AND TM$<>"6446
64" THEN MØ$="MEINE ATOME TREIBEN IM H
YPERRAUM.":GOTO 45ØØ
13Ø11 IF N=47 AND SP=2 AND TM$<>"64466
3" THEN MØ$="MEINE ATOME TREIBEN IM HY
PERRAUM.":GOTO 45ØØ
13Ø2Ø IF N=47 AND (SP=12 OR SP=2) AND
TM$="644664" THEN SP=2:GOTO 1Ø8Ø
13Ø21 IF N=47 AND (SP=12 OR SP=2) AND
TM$="644663" THEN SP=12:GOTO 1Ø8Ø
```

```
13Ø3Ø IF N=22 AND SP=1 THEN FL(1)=-1:P
RINT OK$:GOTO 1Ø8Ø
139ØØ GOSUB 6Ø5:GOTO 1Ø8Ø
14ØØØ IF N=13 AND SP=7 AND OB(13)=7 TH
EN PRINT OK$:OB(12)=Ø:OB(36)=-1:FL(2)=
-1:GOTO 1Ø8Ø
2ØØØØ GRAPHICS Ø:? "UNFALL AUF PIA 3 *
** MEDIKAMENT CIBA- RAD 92 ERFORDERLIC
H *** STATION 17Ø1  ANFLIEGEN ***"
2ØØ1Ø ? :? "KOORDINATEN 17Ø1: 644664":
? "PIA3: 151Ø63":? :? "PASSWORD: 22Ø55
9":? :? "***FEHLFUNKTION***":ALT=Ø:GOT
O 139Ø
2Ø2ØØ IF FL(1)=Ø THEN ? "AUTOPILOT OFF
"
2Ø21Ø IF FL(2)=Ø THEN ? "CPU DEFEKT"
2Ø22Ø IF FL(2)=-1 THEN ? "ALL SYSTEMS
GO"
2Ø23Ø IF FL(1)=Ø THEN ? "KOLLISIONSWAR
NUNG"
2Ø24Ø IF FL(2)=-1 THEN ? "STANDORT: 64
4663"
2Ø25Ø GOTO 139Ø
211ØØ GOSUB 31ØØØ:GOSUB 316ØØ:RETURN
213ØØ PLOT 1ØØ,8Ø:DRAWTO 2ØØ,8Ø:DRAWTO
 2ØØ,12Ø:DRAWTO 1ØØ,12Ø:DRAWTO 1ØØ,8Ø:
DRAWTO 11Ø,7Ø:DRAWTO 19Ø,7Ø
213Ø1 DRAWTO 2ØØ,8Ø
213Ø8 PLOT 18Ø,7Ø:DRAWTO 18Ø,6Ø:PLOT 1
2Ø,6Ø:DRAWTO 12Ø,7Ø:PLOT 11Ø,6Ø:DRAWTO
 19Ø,6Ø:DRAWTO 19Ø,Ø:DRAWTO 11Ø,Ø:DRAW
TO 11Ø,6Ø
21316 PLOT 1ØØ,1ØØ:DRAWTO Ø,1ØØ:PLOT 2
ØØ,1ØØ:DRAWTO 23Ø,1ØØ:DRAWTO 27Ø,14Ø:D
RAWTO 27Ø,2Ø:DRAWTO 23Ø,Ø
2132Ø PLOT 15Ø,7Ø:DRAWTO 15Ø,8Ø:RETURN

217ØØ PLOT 99,1Ø7:PLOT 99,132:PLOT 164
,132:PLOT 174,132:PLOT 244,132:PLOT 24
4,1Ø7:PLOT 174,1Ø7
217Ø8 PLOT Ø,Ø:DRAWTO 3Ø,1Ø:DRAWTO 3Ø,
```

```
8Ø:DRAWTO Ø,1ØØ
21711 PLOT 9Ø,14Ø:DRAWTO 9Ø,1ØØ:DRAWTO
 25Ø,1ØØ:DRAWTO 25Ø,14Ø:DRAWTO 9Ø,14Ø
21715 PLOT 9Ø,1ØØ:DRAWTO 11Ø,8Ø:DRAWTO
 23Ø,8Ø:DRAWTO 25Ø,1ØØ
21718 PLOT 25Ø,12Ø:DRAWTO 31Ø,12Ø
21719 PLOT 9Ø,12Ø:DRAWTO 5Ø,12Ø:DRAWTO
 1Ø,16Ø:PLOT 5Ø,12Ø:DRAWTO 5Ø,Ø:PLOT 1
3Ø,8Ø:DRAWTO 13Ø,6Ø
21723 PLOT 131,6Ø:DRAWTO 131,79:PLOT 1
23,87:DRAWTO 119,91:DRAWTO 122,91:DRAW
TO 126,87:DRAWTO 124,87:RETURN
22ØØØ PLOT 43,6Ø:PLOT 43,59:PLOT 1Ø,1Ø
:DRAWTO 4Ø,1Ø:DRAWTO 5Ø,3Ø:DRAWTO 5Ø,9
Ø:DRAWTO 1Ø,9Ø:PLOT 5Ø,3Ø:DRAWTO 1Ø,3Ø
22ØØ7 PLOT Ø,149:DRAWTO 24Ø,149:DRAWTO
 26Ø, 16Ø:PLOT 239,149:DRAWTO 239,Ø
22Ø1Ø PLOT 219,2Ø:DRAWTO 199,2Ø:DRAWTO
 199,Ø:DRAWTO 179,2Ø:DRAWTO 159,2Ø:DRA
WTO 159,4Ø:DRAWTO 179,4Ø:DRAWTO 179,6Ø
22Ø11 PLOT 199,2Ø:DRAWTO 199,Ø:DRAWTO
179,Ø:DRAWTO 179,2Ø
22Ø18 PLOT 179,6Ø:DRAWTO 199,6Ø:DRAWTO
 199,4Ø:DRAWTO 219,4Ø:DRAWTO 219,2Ø
22Ø25 PLOT 79,14Ø:DRAWTO 219,14Ø:DRAWT
O 199,12Ø:DRAWTO 59,12Ø:DRAWTO 79,14Ø:
RETURN
221ØØ GRAPHICS 8:SETCOLOR 2,1,Ø
221Ø1 PLOT 2Ø,Ø:DRAWTO 2Ø,1ØØ
221Ø2 DRAWTO Ø,12Ø
221Ø3 PLOT 1ØØ,8Ø:DRAWTO 18Ø,8Ø
221Ø4 DRAWTO 21Ø,5Ø:DRAWTO 13Ø,5Ø:DRAW
TO 1ØØ,8Ø
221Ø7 PLOT 11Ø,8Ø:DRAWTO 11Ø,11Ø:PLOT
17Ø,11Ø:DRAWTO 17Ø,8Ø:PLOT 2ØØ,8Ø:DRAW
TO 2ØØ,6Ø:PLOT 2Ø,6Ø:DRAWTO 12Ø,6Ø
221Ø8 PLOT 2ØØ,6Ø
2211Ø DRAWTO 26Ø,6Ø:DRAWTO 28Ø,8Ø:DRAW
TO 28Ø,Ø:DRAWTO 31Ø,3Ø:DRAWTO 31Ø,11Ø:
DRAWTO 314,114:PLOT 26Ø,6Ø:DRAWTO 26Ø,
Ø
```

```
22118 PLOT 2Ø,6Ø
2212Ø IF OB(4Ø)<>11 THEN GOTO 2213Ø
22122 PLOT 13Ø,7Ø:DRAWTO 14Ø,7Ø:DRAWTO
 135,75:DRAWTO 125,75:DRAWTO 13Ø,7Ø
2213Ø RETURN
222ØØ PLOT 21Ø,Ø
222Ø1 PLOT 218,157
222Ø2 PLOT 215,157
222Ø3 PLOT 212,157
222Ø4 PLOT 2Ø9,157
222Ø5 PLOT 2Ø6,157
222Ø6 PLOT 2Ø3,157
222Ø7 PLOT 2ØØ,157
222Ø8 PLOT 197,157
222Ø9 PLOT 194,157
2221Ø PLOT 191,157
22211 PLOT 21Ø,Ø:DRAWTO 25Ø,3Ø
22212 PLOT 25Ø,11Ø:DRAWTO 25Ø,3Ø
22213 PLOT 21Ø,Ø:DRAWTO 17Ø,3Ø
22214 PLOT 17Ø,3Ø:DRAWTO 17Ø,11Ø
22215 PLOT 17Ø,11Ø:DRAWTO 25Ø,11Ø
22216 PLOT 24Ø,16Ø:DRAWTO 22Ø,14Ø
22217 PLOT 22Ø,14Ø:DRAWTO 12Ø,14Ø
22218 PLOT 12Ø,14Ø:DRAWTO 1ØØ,16Ø
22219 PLOT 11Ø,16Ø:DRAWTO 12Ø,15Ø
2222Ø PLOT 12Ø,15Ø:DRAWTO 22Ø,15Ø
22221 PLOT 22Ø,15Ø:DRAWTO 23Ø,16Ø
22222 PLOT 61,Ø:DRAWTO 61,7Ø
22223 PLOT 61,7Ø:DRAWTO 111,2Ø
22224 PLOT 111,2Ø:DRAWTO 111,Ø
22225 PLOT 111,2Ø:DRAWTO 181,2Ø
22226 PLOT 238,2Ø:DRAWTO 318,2Ø
22227 PLOT 22Ø,8Ø:DRAWTO 25Ø,11Ø
22228 PLOT 17Ø,11Ø:DRAWTO 2ØØ,8Ø
22229 PLOT 2ØØ,8Ø:DRAWTO 22Ø,8Ø
2223Ø PLOT 25Ø,3Ø:DRAWTO 22Ø,6Ø
22231 PLOT 22Ø,6Ø:DRAWTO 2ØØ,6Ø
22232 PLOT 2ØØ,6Ø:DRAWTO 17Ø,3Ø
22233 PLOT 3Ø,1Ø:DRAWTO 3Ø,1ØØ
22234 PLOT 3Ø,1ØØ:DRAWTO 1Ø,12Ø:RETURN
```

```
223ØØ PLOT Ø,Ø
223Ø1 PLOT Ø,Ø:DRAWTO 4Ø,4Ø
223Ø2 PLOT 4Ø,4Ø:DRAWTO 26Ø,4Ø:DRAWTO
3ØØ,Ø
223Ø4 PLOT 28Ø,2Ø:DRAWTO 2Ø,2Ø
223Ø5 PLOT 4Ø,4Ø:DRAWTO 4Ø,1ØØ:DRAWTO
26Ø,1ØØ:DRAWTO 26Ø,4Ø
223Ø8 PLOT 26Ø,1ØØ:DRAWTO 3ØØ,14Ø
223Ø9 PLOT 4Ø,1ØØ:DRAWTO 3Ø,11Ø:DRAWTO
 3Ø,5Ø:DRAWTO Ø,2Ø:DRAWTO Ø,14Ø
22315 IF (OB(5Ø)=13 OR SP=15) THEN GOS
UB 22122
22316 RETURN
224ØØ PLOT 6Ø,Ø:DRAWTO 6Ø,1ØØ
224Ø2 PLOT 6Ø,1ØØ:DRAWTO 8Ø,8Ø
224Ø3 PLOT 8Ø,4Ø:DRAWTO 8Ø,12Ø
224Ø4 PLOT 8Ø,12Ø:DRAWTO 16Ø,12Ø
224Ø5 PLOT 16Ø,12Ø:DRAWTO 16Ø,8Ø
224Ø6 PLOT 16Ø,8Ø:DRAWTO 14Ø,8Ø
224Ø7 PLOT 14Ø,8Ø:DRAWTO 1ØØ,4Ø
224Ø8 PLOT 1ØØ,4Ø:DRAWTO 8Ø,4Ø
224Ø9 PLOT 8Ø,4Ø:DRAWTO 12Ø,Ø
2241Ø PLOT 1ØØ,4Ø:DRAWTO 14Ø,Ø
22411 PLOT 14Ø,8Ø:DRAWTO 22Ø,Ø
22412 PLOT 24Ø,Ø:DRAWTO 16Ø,8Ø
22413 PLOT 16Ø,12Ø:DRAWTO 26Ø,2Ø
22414 PLOT 26Ø,2Ø:DRAWTO 26Ø,Ø
22415 PLOT 26Ø,1Ø:DRAWTO 28Ø,1Ø
22416 PLOT 28Ø,1Ø:DRAWTO 28Ø,Ø:RETURN
226ØØ PLOT 1ØØ,Ø:DRAWTO 2ØØ,Ø:DRAWTO 2
ØØ,14Ø:DRAWTO 1ØØ,14Ø:DRAWTO 1ØØ,Ø
226Ø2 IF SP=9 THEN PLOT 15Ø,Ø:DRAWTO 1
5Ø,14Ø
226Ø5 IF DU(16,4)<>1 AND SP=9 THEN PLO
T 15Ø,Ø:DRAWTO 15Ø,14Ø
226Ø6 PLOT 22Ø,6Ø:DRAWTO 22Ø,8Ø:DRAWTO
 21Ø,8Ø:DRAWTO 21Ø,6Ø:DRAWTO 22Ø,6Ø
2261Ø PLOT 5Ø,142:DRAWTO 26Ø,142:DRAWT
O 26Ø,Ø
22612 PLOT 5Ø,142:DRAWTO 5Ø,Ø
22613 PLOT 5Ø,142:DRAWTO 32,16Ø
```

```
22614 PLOT 26Ø, 142:DRAWTO 278,16Ø
2263Ø IF SP=9 OR SP=6 THEN GOTO 22699
22631 IF DU(16,4)<>1 THEN GOTO 22699
22643 PLOT 17Ø,Ø:DRAWTO 17Ø,6Ø:DRAWTO
2ØØ,3Ø
22645 PLOT 17Ø,6Ø:DRAWTO 16Ø,6Ø:DRAWTO
 15Ø,7Ø:DRAWTO 15Ø,5Ø:DRAWTO 16Ø,4Ø:DR
AWTO 16Ø,6Ø
2265Ø PLOT 15Ø,7Ø:DRAWTO 1ØØ,7Ø
22651 PLOT 1ØØ,5Ø:DRAWTO 15Ø,5Ø
22652 PLOT 16Ø,4Ø:DRAWTO 1ØØ,4Ø
22653 PLOT 1ØØ,3Ø:DRAWTO 16Ø,3Ø
22654 PLOT 16Ø,3Ø:DRAWTO 16Ø,Ø
22699 RETURN
227ØØ PLOT 1Ø,Ø:DRAWTO 1Ø,1ØØ:DRAWTO Ø
,11Ø:PLOT 13Ø,5Ø:DRAWTO 1Ø,5Ø:PLOT Ø,1
2Ø:DRAWTO 31Ø,12Ø
227Ø4 PLOT 31Ø,9Ø:DRAWTO 25Ø,9Ø:DRAWTO
 25Ø,Ø
227Ø6 PLOT 25Ø,5Ø:DRAWTO 17Ø,5Ø:DRAWTO
 17Ø,Ø:PLOT 17Ø,1Ø:DRAWTO 13Ø,5Ø:DRAWT
O 13Ø,Ø:RETURN
23ØØØ ? "HIER IST ES ZU DUNKEL ZUM SEH
EN.":? "O.";:INPUT EINGABE$:IF EINGABE
$(1,4)<>"TAST" THEN 23Ø9Ø
23Ø1Ø IF EINGABE$="O" THEN SP=12
23Ø2Ø ? "HIER IST EINE SAUERSTOFFFLASC
HE.":OB(28)=2Ø
23Ø9Ø RETURN
232ØØ GOSUB 312ØØ:GOSUB 314ØØ:GOSUB 31
3ØØ:RETURN
233ØØ GOSUB 312ØØ:IF FL(5) AND SP=23 T
HEN GOSUB 314ØØ
2331Ø RETURN
234ØØ GOSUB 312ØØ:GOSUB 313ØØ:GOSUB 31
4ØØ:RETURN
236ØØ GOSUB 312ØØ:GOSUB 314ØØ:RETURN
237ØØ GOSUB 312ØØ:GOSUB 314ØØ:RETURN
24ØØØ PLOT 3,1Ø3:DRAWTO 73,1Ø3:DRAWTO
1Ø3,73:DRAWTO 1Ø3,53:DRAWTO 73,83:DRAW
TO 73,1Ø3
```

```
24ØØ6 PLOT 73,93:DRAWTO 3,93
24ØØ7 PLOT 93,63:DRAWTO 3,63
24ØØ8 PLOT 1Ø3,72:DRAWTO 223,72
24ØØ9 PLOT 223,Ø:DRAWTO 223,72:DRAWTO
263,112:DRAWTO 263,12:DRAWTO 253,2
24Ø13 PLOT 263,62:DRAWTO 3Ø3,62
24Ø14 PLOT 313,16Ø:DRAWTO 3Ø3,15Ø:DRAW
TO 3Ø3,5Ø:DRAWTO 253,Ø:RETURN
3ØØØØ IF SP>2Ø THEN 3ØØ3Ø
3ØØØ5 IF SP>1Ø THEN 3ØØ2Ø
3ØØ1Ø ON SP GOSUB 213ØØ,222ØØ,213ØØ,21
4ØØ,215ØØ,226ØØ,217ØØ,213ØØ,226ØØ,22ØØ0
3ØØ11 RETURN
3ØØ2Ø ON SP=1Ø GOSUB 221ØØ,222ØØ,223ØØ
,224ØØ,223ØØ,226ØØ,227ØØ,227ØØ,227ØØ,2
3ØØØ
3ØØ21 RETURN
3ØØ3Ø ON SP=2Ø GOSUB 231ØØ,232ØØ,233ØØ
,234ØØ,233ØØ,236ØØ,237ØØ,238ØØ,239ØØ,2
4ØØØ
3ØØ31 RETURN
312ØØ PLOT 8Ø,Ø:DRAWTO 8Ø,16Ø
312Ø2 PLOT 12Ø,14Ø:DRAWTO 14Ø,14Ø:PLOT
 2ØØ,14Ø:DRAWTO 22Ø,14Ø:PLOT 26Ø,16Ø:D
RAWTO 26Ø,Ø
312Ø3 PLOT 2ØØ,14Ø:DRAWTO 22Ø,14Ø
312Ø4 PLOT 26Ø,16Ø:DRAWTO 26Ø,Ø
312Ø5 PLOT 22Ø,1ØØ:DRAWTO 2ØØ,1ØØ:PLOT
 2ØØ,6Ø:DRAWTO 22Ø,6Ø:PLOT 22Ø,2Ø:DRAW
TO 2ØØ,2Ø
312Ø8 PLOT 14Ø,2Ø:DRAWTO 12Ø,2Ø:PLOT 1
2Ø,6Ø:DRAWTO 14Ø,6Ø
3121Ø PLOT 14Ø,1ØØ:DRAWTO 12Ø,1ØØ:RETURN
313ØØ PLOT 6Ø,8Ø
313Ø1 PLOT 2Ø,2Ø:DRAWTO 6Ø,4Ø:DRAWTO 6
Ø,1ØØ:DRAWTO 2Ø,14Ø:RETURN
314ØØ PLOT 28Ø,4Ø:DRAWTO 3ØØ,2Ø
314Ø2 PLOT 28Ø,4Ø:DRAWTO 28Ø,1ØØ:DRAWT
O 3ØØ,12Ø:IF FL(5) AND SP=23 THEN PLOT
 285,7Ø
314Ø5 RETURN
```

Grafik Programmer

GRAFIK-PROGRAMMER

Das Ihnen im folgenden als Listing vorgestellte Programm wurde von mir entwickelt, um einfache HIRES-Grafiken auf leichte und schnelle Art programmieren zu können.

Die Vorgehensweise ist denkbar einfach: Sie erzeugen unter Verwendung eines tastengesteuerten Cursors und mit Hilfe einiger vom Editor gestellter Fragen die Grafik auf dem Bildschirm. Die für die Erzeugung dieses Bildes notwendigen Daten werden gespeichert und später zur Erstellung eines Programmes verwendet, welches als Unterprogramm an andere Programme angefügt werden kann.
Somit ist es nicht erforderlich, daß Sie die jeweiligen Koordinaten mit irgendwelchen Hilfsmitteln, wie beispielsweise Bildschirmarbeitsblättern, ermitteln müssen.

ERSTELLUNG VON GRAFIKEN MIT GRAFPRO

Zunächst sollten Sie darauf achten, daß Ihr Atari sich im Großschriftmodus befindet.
Nach dem Titelbild fordert der Editor die gewünschte Betriebsart an, das heißt, Sie wählen den Graphics-Modus wie auch Vorder- und Hintergrundfarbe.
Betätigen Sie jeweils eine beliebige Taste; entspricht die Farbe Ihren Wünschen, so drücken Sie <RETURN).

Nachdem der Rechner in die gewünschte Betriebsart umgeschaltet hat, stehen Ihnen nun eine Reihe von Zeichen- wie auch Steuerbefehlen zur Verfügung.
Zeichenbefehle werden durch Drücken der für sie vorgesehenen Taste ausgeführt, bei Steuerbefehlen müssen Sie gleichzeitig die <CONTROL> - Taste betätigen.

CURSORBEWEGUNG

Die Cursor ist ein nicht zerstörender, schnell blinkender Punkt auf dem Grafikschirm. Zu seiner Steuerung dienen die Tasten

```
Q W E
A   D
Z X C
```

wobei deren Anordnung auf der Tastatur der jeweiligen Bewegungsrichtung entspricht.
Bei Programmstart befindet sich der Cursor immer auf der Position Ø,Ø, somit in der linken oberen Ecke.
Damit die Ansteuerung eines beliebigen Punktes auf dem Bildschirm nun nicht zu einer Geduldsarbeit wird, haben Sie die Möglichkeit, die Schrittweite des Cursors nach Eingabe von ***CTRL-C*** beliebig groß festzulegen.
Die Alternative sieht eine Betätigung der Taste P vor, anschließend können sie die neuen Koordinaten direkt eingeben.

ZEICHNEN

Um einen Punkt zu zeichnen, steuern Sie den Cursor auf die gewünschte Position, und drücken ***U*** für Übernahme.
Sofort wird der Punkt gesetzt, die entsprechenden Koordinaten werden für die spätere Programmierung gespeichert.

Wünschen Sie eine Linie zu zeichnen, so steuern Sie auf gewohnte Weise zunächst den Startpunkt an.
Betätigen Sie die ***L***-Taste, um diesen Punkt zu fixieren, steuern Sie den Endpunkt an und drücken Sie wiederum ***L***.

FEHLERBEHANDLUNG

Damit nicht versehentlich falsche Werte übernommen werden, ist es erforderlich, die Korrektheit eines jeden gesetzten Punktes und jeder gezeichneten Linie mit ***J*** zu bestätigen. Die Betätigung einer jeden anderen Taste macht den zuletzt durchgeführten Vorgang rückgängig.

Aus Gründen der Schnelligkeit kann es nun wünschenswert sein, auf diese Bestätigung zu verzichten. Mit CTRL-B können Sie diese Kontrollfunktion jederzeit ein- bzw. ausschalten.

Sollte Ihnen nun ein Fehler unterlaufen, so ist Ihr Werk dennoch nicht verloren. Nach Eingabe von CTRL-Z wird der Bildschirm gelöscht und das Bild neu gezeichnet, allerdings ohne die zuletzt gezeichnete Linie und ohne den zuletzt gesetzten Punkt.

BILD SPEICHERN, BILD LADEN

Betätigen Sie CTRL-L oder CTRL-S und geben Sie einen bis zu acht Buchstaben langen Namen für das Bild ein. Grafpro fügt automatisch die Ergänzung .BDT (Bilddaten) an und führt den gewünschten Vorgang aus.

BILD PROGRAMMIEREN

Nach gleichzeitigem Niederhalten von CONTROL und P ist wiederum die Eingabe eines Namens erforderlich. Anschließend werden Sie nach der Zeilennummer gefragt, mit der das Programm beginnen soll, und es wird ein entsprechendes Programm erzeugt.

CONTROL Q beendet den Programmlauf.

VERWENDUNG DER ERZEUGTEN PROGRAMME

Die generierten Programme entsprechen nicht dem üblichen Format, in dem ein BASIC-Programm gespeichert wird. Dementsprechend können sie auch nicht mit dem Befehl LOAD, sondern müssen mit ***ENTER*** geladen werden.

Beachten Sie bitte, daß bei einem Ladevorgang durch ENTER der bisherige Speicherinhalt nicht gelöscht wird, was einerseits natürlich erforderlich ist, um verschiedene Programmteile verbinden zu können, andererseits aber auch zu wirren Listings führen kann.

```
1 REM GRAFIK PROGRAMMER
2 REM (C) 1984 BY JOERG WALKOWIAK
3 REM RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
5 GRAPHICS Ø:SETCOLOR 2,Ø,Ø
6 PRINT "     GRAFIK EDITOR & PROGAMMI
ERER            (C) 1984 BY DATA BECKE
R        ":PRINT :PRINT :PRINT
1Ø SETCOLOR 2,Ø,Ø:GOTO 1ØØ
19 REM ***************** UNTERPROGRAMME
2Ø PRINT #1,ZEILE,Z$:ZEILE=ZEILE+1Ø:RE
TURN
4Ø Z$(LEN(Z$)+1)=H$:RETURN
8Ø Z$(LEN(Z$)+1)=STR$(X):RETURN
99 REM *************** INITIALISIERUNG
1ØØ CLR :DIM PX(2ØØ),PY(2ØØ),PC(2ØØ),V
X(1ØØ),VY(1ØØ),NX(1ØØ),NY(1ØØ),NAME$(8
),FILE$(12),Z$(5Ø),H$(4Ø)
11Ø PZ=Ø:LZ=Ø:C=1:PX(Ø)=Ø:PY(Ø)=Ø:PX(1
)=Ø:PY(1)=Ø
12Ø XMAX=32Ø:YMAX=16Ø
13Ø KONTROLLE=Ø
2ØØ OPEN #2,4,Ø,"K:"
2Ø5 HF=Ø:PRINT "BITTE WAEHLEN SIE DIE
...":PRINT :PRINT
2Ø7 POSITION 2,23:PRINT "WENN OKAY, DA
NN <RETURN> DRUECKEN …";
21Ø POSITION 2,9:PRINT "... HINTERGRUN
DFARBE":GET #2,E:IF E=155 THEN GOTO 23
Ø
215 HF=HF+1:IF HF=16 THEN HF=Ø
22Ø PRINT HF:SETCOLOR 2,HF,Ø:GOTO 21Ø
225 HE=Ø
23Ø POSITION 2,11:PRINT "... HELLIGKEI
T":GET #2,E:IF E=155 THEN GOTO 25Ø
235 HE=HE+1:IF HE=15 THEN HE=Ø
24Ø PRINT HE:SETCOLOR 2,HF,HE:GOTO 23Ø
245 GET #2,E:IF E<>155 THEN GOTO 2Ø5
25Ø POSITION 2,9:PRINT "
      ":POSITION 5,9:PRINT
    "
26Ø POSITION 2,9:PRINT "... GRAFIK BET
```

```
RIEBSART  8";CHR$(3Ø);CHR$(3Ø);:INPUT
MODUS
27Ø POSITION 2,11:PRINT "... STIFTFARB
E  1";CHR$(3Ø);CHR$(3Ø);:INPUT SF
28Ø GRAPHICS MODUS
29Ø SETCOLOR 2,HF,HE:COLOR SF
3ØØ TRAP 1ØØØØ:REM FEHLER ABFANGEN
999 REM ***************** HAUPTPROGRAM
M
1ØØØ GET #2,E
1Ø1Ø REM CURSOR BEWEGEN
1Ø2Ø IF E=87 THEN Y=Y-C:GOTO 15ØØ
1Ø3Ø IF E=88 THEN Y=Y+C:GOTO 15ØØ
1Ø4Ø IF E=68 THEN X=X+C:GOTO 15ØØ
1Ø5Ø IF E=65 THEN X=X-C:GOTO 15ØØ
1Ø6Ø IF E=69 THEN X=X+C:Y=Y-C:GOTO 15Ø
Ø
1Ø7Ø IF E=67 THEN X=X+C:Y=Y+C:GOTO 15Ø
Ø
1Ø8Ø IF E=81 THEN X=X-C:Y=Y-C:GOTO 15Ø
Ø
1Ø9Ø IF E=9Ø THEN X=X-C:Y=Y+C:GOTO 15Ø
Ø
11ØØ REM BEFEHL FINDEN
111Ø IF E=76 THEN GOTO 3ØØØ
112Ø IF E=26 THEN GOTO 5ØØØ
113Ø IF E=16 THEN GOTO 4ØØØ
114Ø IF E=19 THEN GOTO 6ØØØ
115Ø IF E=12 THEN GOTO 7ØØØ
116Ø IF E=14 THEN CLOSE #2:CLR :RUN
117Ø IF E=8Ø THEN GOTO 31ØØ
118Ø IF E=85 THEN GOTO 311Ø
119Ø IF E=3 THEN PRINT "NEUE CURSOR-SC
HRITTWEITE ";:INPUT C:PRINT :PRINT :PR
INT:GOTO 1ØØØ
12ØØ IF E=17 THEN GOTO 2ØØØØ
1499 REM BEREICHSKONTROLLE
15ØØ IF X>XMAX THEN X=XMAX
151Ø IF X<Ø THEN X=Ø
152Ø IF Y<Ø THEN Y=Ø
153Ø IF Y>YMAX THEN Y=YMAX
```

```
154Ø GOSUB 2ØØØ
1545 PRINT "X,";X;" Y:";Y;" F:";F:IF E
=155 THEN GOTO 315Ø
155Ø IF F1=Ø THEN COLOR HF:PLOT X1,Y1
1559 REM PUNKT SETZEN
156Ø IF E=8Ø THEN F1=1:COLOR SF:PLOT X
,Y:PX(PZ)=X:PY(PZ)=Y:PC(PZ)=3:PZ=PZ+1
157Ø IF E=32 THEN F1=Ø:COLOR HF:PLOT X
,Y:PX(PZ)=X:PY(PZ)=Y:PC(PZ)=2:PZ=PZ+1
158Ø GOTO 1ØØØ
1999 REM ************** POSITION ZEIGE
N
2ØØØ LOCATE X,Y,F
2Ø1Ø FOR I=1 TO 2Ø:COLOR SF:PLOT X,Y:C
OLOR HF:PLOT X,Y:NEXT I
2Ø2Ø COLOR HF:PLOT X,Y
2Ø3Ø RETURN
2999 REM *************** EINGABE LINIE
3ØØØ IF NACH=Ø THEN PRINT "LINIE VON X
=";X;", Y=";Y:VX=X:VY=Y:NACH=-1:GOTO 1
ØØØ
3ØØ5 PRINT "NACH ";X;", ";Y:NX=X:NY=Y:
NACH=Ø
3Ø1Ø COLOR SF:PLOT VX,VY:DRAWTO NX,NY:
IF KONTROLLE=Ø THEN E=74:GOTO 3Ø3Ø
3Ø2Ø PRINT "OKAY ?":GET #2,E
3Ø3Ø IF E=74 THEN VX(LZ)=VX:VY(LZ)=VY:
NX(LZ)=NX:NY(LZ)=NY:PRINT :PRINT :LZ=L
Z+1:GOTO 1ØØØ
3Ø4Ø IF E<>74 THEN COLOR HF:PLOT VX,VY
:DRAWTO NX,NY:GOTO 1ØØØ
3Ø99 REM ************** EINGABE PUNKT
31ØØ PRINT "X,Y ";:INPUT X,Y
311Ø IF X>XMAX THEN PRINT "** X IST ZU
 GROSS":GOTO 31ØØ
312Ø IF Y>YMAX THEN PRINT "** Y IST ZU
 GROSS":GOTO 31ØØ
313Ø IF Y<Ø THEN PRINT "** Y IST UNZUL
AESSIG":GOTO 31ØØ
314Ø IF X<Ø THEN PRINT "** X IST UNZUL
AESSIG":GOTO 31ØØ
```

```
315Ø COLOR SF:PLOT X,Y:IF KONTROLLE=Ø
THEN E=74:GOTO 317Ø
316Ø PRINT "OKAY ?":GET #2,E
317Ø IF E=74 THEN PX(PZ)=X:PY(PZ)=Y:PR
INT :PRINT :PZ=PZ+1:GOTO 1ØØØ
318Ø IF E<>74 THEN COLOR HF:PLOT X,Y:G
OTO 1ØØØ
3999 REM ************ PROGRAMM ERZEUGE
N
4ØØØ PRINT "NAME DES BILDES ";:INPUT N
AME$:IF LEN(NAME$)>8 THEN PRINT " ** Z
U LANG ! **":GOTO 6ØØØ
4ØØ5 PRINT "ERSTE ZEILENNUMMER ";:INPU
T ZEILE
4Ø1Ø NAME$(LEN(NAME$)+1)=".BAS"
4Ø2Ø FILE$="D:":FILE$(LEN(FILE$)+1)=NA
ME$
4Ø3Ø OPEN #1,8,Ø,FILE$
4Ø5Ø Z$="GRAPHICS ":X=MODUS:GOSUB 8Ø:H
$=":SETCOLOR 2,":GOSUB 4Ø:X=HF:GOSUB 8
Ø:H$=",":GOSUB 4Ø
4Ø55 X=HE:GOSUB 8Ø:GOSUB 2Ø
4Ø6Ø Z$="COLOR ":X=SF:GOSUB 8Ø:GOSUB 2
Ø
41ØØ FOR P=Ø TO PZ-1
411Ø Z$="PLOT ":X=PX(P):GOSUB 8Ø:H$=",
":GOSUB 4Ø:X=PY(P):GOSUB 8Ø:GOSUB 2Ø
412Ø NEXT P
415Ø FOR L=Ø TO LZ-1
416Ø Z$="PLOT ":X=VX(L):GOSUB 8Ø:H$=",
":GOSUB 4Ø:X=VY(L):GOSUB 8Ø:H$=":DRAWT
O":GOSUB 4Ø
417Ø X=NX(L):GOSUB 8Ø:H$=",":GOSUB 4Ø:
X=NY(L):GOSUB 8Ø:GOSUB 2Ø
418Ø NEXT L
419Ø CLOSE #1:PRINT "BILD IST PROGRAMM
IERT !":PRINT :GOTO 1ØØØ
4999 REM ************ BILD NEU ZEICHNE
N
5ØØØ GRAPHICS MODUS:SETCOLOR 2,HF,HE:C
OLOR SF
```

```
5Ø1Ø PZ=PZ-1:FOR P=Ø TO PZ-1
5Ø2Ø X=PX(P):Y=PY(P):PLOT X,Y
5Ø3Ø NEXT P
5Ø4Ø LZ=LZ-1:FOR L=Ø TO LZ-1
5Ø5Ø PLOT VX(L),VY(L):NX=NX(L):NY=NY(L
):DRAWTO NX,NY
5Ø6Ø NEXT L
5Ø7Ø PRINT "ZEICHNUNG FERTIG !":GOTO 1
ØØØ
5999 REM ********** BILDDATEN SPEICHER
N
6ØØØ PRINT "SPEICHERN: NAME DES BILDES
 ";:INPUT NAME$:IF LEN(NAME$)>8 THEN P
RINT " ** ZU LANG ! **":GOTO 6ØØØ
6Ø1Ø NAME$(LEN(NAME$)+1)=".BDT"
6Ø2Ø FILE$="D:":FILE$(LEN(FILE$)+1)=NA
ME$
6Ø3Ø OPEN #1,B,Ø,FILE$
6Ø4Ø PRINT #1,MODUS:PRINT #1,HF:PRINT
#1,HE:PRINT #1,PZ:PRINT #1,LZ
6Ø5Ø FOR P=Ø TO PZ-1
6Ø6Ø PRINT #1,PX(P):PRINT #1,PY(P)
6Ø7Ø NEXT P
6Ø8Ø FOR L=Ø TO LZ-1
6Ø9Ø PRINT #1,VX(L):PRINT #1,VY(L):PRI
NT #1,NX(L),NY(L)
61ØØ NEXT L
611Ø PRINT "BILDDATEN GESPEICHERT !":C
LOSE #1:GOTO 1ØØØ
6999 REM ************** BILDDATEN LADE
N
7ØØØ PRINT "LADEN: NAME DES BILDES ";:
INPUT NAME$:IF LEN(NAME$)>8 THEN PRINT
 " ** ZU LANG ! **":GOTO 6ØØØ
7Ø1Ø NAME$(LEN(NAME$)+1)=".BDT"
7Ø2Ø FILE$="D:":FILE$(LEN(FILE$)+1)=NA
ME$
7Ø3Ø OPEN #1,4,Ø,FILE$
7Ø4Ø INPUT #1,MODUS:INPUT #1,HF:INPUT
#1,HE:INPUT #1:PZ:INPUT #1,LZ
7Ø5Ø FOR P=Ø TO PZ-1
```

```
7Ø6Ø INPUT #1,X:INPUT #1,Y:PX(P)=X:PY(
P)=Y
7Ø7Ø NEXT P
7Ø8Ø FOR L=Ø TO LZ-1
7Ø9Ø INPUT #1,X:INPUT #1,Y:VX(L)=X:VY(
L)=Y:INPUT #1,X:INPUT #1,Y:NX(L)=X:NY(
L)=Y
71ØØ NEXT L
711Ø PRINT "BILDDATEN GELADEN !":CLOSE
 #1:GOTO 1ØØØ
7999 REM ***************** KONTROLLE
?
8ØØØ IF KONTROLLE=-1 THEN KONTROLLE=Ø:
PRINT "ACHTUNG! KEINE RUECKFRAGEN !":G
OTO 1ØØØ
8Ø1Ø IF KONTROLLE=Ø THEN KONTROLLE=-1:
PRINT "BESTAETIGUNG ERFORDERLICH.":GOT
O 1ØØØ
1ØØØØ PRINT "**** ERROR ****":GOTO 1ØØ
Ø
2ØØØØ CLOSE #1:CLOSE #2:SRAPHICS Ø:END
```

Adventure Generator

ADVENTUREGENERATOR
©VENTUREFIX©

Dieses Programm wird Sie mit allen Fragen konfrontieren, die beantwortet sein müssen, um ein komplettes, funktionsfähiges Adventureprogramm zu erzeugen.

Vor dem Start von ©Venturefix© muß die Handlung stehen, Sie müssen eine Liste aller Räume, Gegenstände, Wörter und auch Mitteilungen erstellt haben, wie Sie auch sämtliche Aktionen bereits textuell formuliert haben sollten.

Die kreative Arbeit wird Ihnen also nicht abgenommen werden, wohl aber die Routinearbeit des Programmierens.
Sie müssen nicht einmal wissen, was ein PRINT-Befehl ist, dennoch können Sie fehlerlose BASIC-Programme erzeugen.

Venturefix wird im Dialog alle erforderlichen Daten der Reihe nach von Ihnen anfordern und Sie ständig über den Stand der Entwicklung informieren, bis schließlich ein fertiges Adventure im Stile der in diesem Buch vorgestellten Programme entstanden ist.

HINWEISE FÜR BESITZER DES ATARI 6ØØ XL

Erschrecken Sie bitte nicht, wenn Sie das folgende Listing auf 3Ø - 4Ø KiB schätzen, denn Venturefix kann ohne Probleme in mehrere Teile zerlegt werden, so daß auch Sie mit diesem Programmgenerator arbeiten können.

Teilen Sie das folgende Listing in gerade noch in den Speicher passende Blöcke auf (geben Sie des öfteren während der Eingabe PRINT FRE(O) ein, mehr als 1ØØØ Bytes brauchen auf keinen Fall frei zu bleiben).

Lassen Sie die Zeilennummern dabei unverändert, achten Sie nur darauf, daß Sie das Programm nicht inmitten einer Schleife trennen und daß Sie nicht auf die erforderlichen Unterprogramme mit den Zeilennummern 2Ø bis 1ØØ verzichten. Weiterhin muß die erste Programmzeile aller nach Teil 1 folgenden Teile lauten:

```
Ø OPEN #1,9,Ø,"D:ADVENTUR.TXT"
```

Beenden Sie jeden Teil mit:

```
6ØØØØ CLOSE #1:PRINT "NAECHSTER PRO-
GRAMMTEIL WIRD GELADEN":RUN"D:VENTUREx
```

wobei Sie x natürlich durch die Nummer des nächsten Programmteiles ergänzen.

```
1 REM 3.11.84
5 GRAPHICS Ø:SETCOLOR 2,Ø,Ø:POSITION 1
,5
6 PRINT "VENTUREFIX":PRINT "(c) 1984 b
y Joerg Walkowiak":PRINT :PRINT :PRINT
 "aus dem DATA BECKER Buch:"
7 PRINT :PRINT "ADVENTURES, und wie ma
n sie auf dem   ATARI programmiert":PR
INT :PRINT
8 PRINT "(C) 1984 BY DATA BECKER, DUES
SELDORF":FOR I=1 TO 3ØØØ:NEXT I
9 DIM ADVENTURE$(12),EINGABE$(2Ø),COPY
RIGHT$(35),DATUM$(1Ø),Z$(12Ø),H$(8Ø),V
ERSION$(5),VERBEN$(8Ø),MØ$(22),M1$(38)
1Ø DIM M$(4Ø):GOTO 1ØØ
2Ø IF DRUCK THEN LPRINT ZEILENNUMMER;"
 ";Z$
3Ø PRINT #1,ZEILENNUMMER,Z$:ZEILENNUMM
ER=ZEILENNUMMER+ZEILENABSTAND:RETURN
4Ø Z$(LEN(Z$)+1)=H$:RETURN
5Ø GOSUB 7Ø:GOSUB 4Ø:GOSUB 7Ø:H$=":RET
URN":GOSUB 4Ø:GOSUB 2Ø:RETURN
6Ø GOSUB 4Ø:GOSUB 7Ø:GOSUB 2Ø:RETURN
7Ø Z$(LEN(Z$)+1)=CHR$(34):RETURN
8Ø Z$(LEN(Z$)+1)=STR$(X):RETURN
9Ø OPEN #2,4,Ø,"K:":GET #2,EINGABE:CLO
SE #2:RETURN
95 GOSUB 4Ø:GOSUB 2Ø:RETURN
1ØØ GRAPHICS Ø:SETCOLOR 2,Ø,Ø
11Ø ADVENTURE$="D:":PRINT "WIE SOLL DA
S ADVENTURE HEISSEN:":PRINT :INPUT EIN
GABE$
12Ø IF LEN(EINGABE$)>8 THEN PRINT "BIT
TE NUR MAXIMAL 8 BUCHSTABEN !":GOTO 1Ø
Ø
13Ø ADVENTURE$(LEN(ADVENTURE$)+1)=EING
ABE$:ADVENTURE$(LEN(ADVENTURE$)+1)=".A
DV"
14Ø PRINT :PRINT "SOLL GLEICHZEITIG EI
N LISTING AUF":PRINT "EINEM DRUCKER ER
ZEUGT WERDEN ?"
```

```
15Ø GOSUB 9Ø
16Ø IF (EINGABE=74 OR EINGABE=2Ø2) THE
N DRUCK=-1
17Ø ZEILENNUMMER=1Ø
18Ø ZEILENABSTAND=5
2ØØ GRAPHICS Ø:SETCOLOR 2,Ø,Ø:PRINT "K
ENNDATEN DES ADVENTURES: ";ADVENTURE$(
1,7)
21Ø PRINT "----------------------------
----------"
22Ø PRINT "NAME DES AUTORS ":INPUT COP
YRIGHT$:IF LEN(COPYRIGHT$)>35 THEN PRI
NT "... LEIDER ZU LANG !":GOTO 22Ø
23Ø PRINT EINGABE$;", VERSION$: ":INPU
T VERSION$
24Ø PRINT "HEUTIGES DATUM ":INPUT DATU
M$
25Ø PRINT "WIE VIELE RAEUME ":INPUT AR
:DIM DU(AR,6)
26Ø PRINT "WIE VIELE OBJEKTE ":INPUT A
O:DIM OB(AO)
27Ø PRINT "WIE VIELE VERBEN ":INPUT AV
28Ø PRINT "SCHAETZWERT, WIE VIELE FLAG
S ":INPUT AF
29Ø PRINT :PRINT "IN WELCHEM RAUM SOLL
 ";ADVENTURE$(3,1Ø);" BEGINNEN ":INPUT
 SPIELER
5ØØ GRAPHICS Ø:PRINT "TEIL   I : INITI
ALISIERUNG":PRINT :PRINT
52Ø PRINT " - DATEI OEFFNEN":OPEN #1,8
,Ø,ADVENTURE$
53Ø PRINT " - PROGRAMMKOPF":Z$="REM ":
Z$(LEN(Z$)+1)=EINGABE$:GOSUB 2Ø
54Ø Z$="REM VERSION ":Z$(LEN(Z$)+1)=VE
RSION$:GOSUB 2Ø
55Ø Z$="REM (C) BY ":Z$(LEN(Z$)+1)=COP
YRIGHT$:GOSUB 2Ø
56Ø Z$="REM ERSTELLT AM ":Z$(LEN(Z$)+1
)=DATUM$:GOSUB 2Ø
57Ø Z$="REM UNTER ZUHILFENAHME VON:":G
OSUB 2Ø
```

```
58Ø Z$="REM  VENTUREFIX            ":G
OSUB 2Ø
59Ø Z$="REM  (C) 1984  BY WALKOWIAK":G
OSUB 2Ø
6ØØ PRINT " - TITEL":Z$="REM----------
----------------------":GOSUB 2Ø
61Ø Z$="GRAPHICS Ø:SETCOLOR 2,1,Ø:SETC
OLOR 4,1,Ø":GOSUB 2Ø
62Ø Z$="PRINT ":GOSUB 7Ø:Z$(LEN(Z$)+1)
=EINGABE$:GOSUB 2Ø
63Ø Z$="REM ***** HIER TITELBILD EINFU
EGEN":GOSUB 2Ø:ZEILENNUMMER=ZEILENNUMM
ER+5Ø
64Ø PRINT " - KENNDATEN":Z$="AR=":X=AR
:GOSUB 8Ø:GOSUB 2Ø
65Ø Z$="AO=":X=AO:GOSUB 8Ø:GOSUB 2Ø
66Ø Z$="AF=":X=AF:GOSUB 8Ø:GOSUB 2Ø
67Ø Z$="SP=":X=SPIELER:GOSUB 8Ø:GOSUB
2Ø
68Ø Z$="TRAP 49ØØ":GOSUB 2Ø
7ØØ PRINT " - FELDER DIMENSIONIEREN"
71Ø Z$="DIM RA$(8Ø),DU(AR,6),RI$(36),E
INGABE$(2Ø),T$(4Ø),OB(AO),LZ$(38),DZ$(
5Ø),ZE$(3),FL(AF),OK$(4)":GOSUB 2Ø
72Ø Z$="DIM VERBEN$(12Ø),EV$(2Ø),EO$(2
Ø),OBJEKTE$(3ØØ),MØ$(4Ø)":GOSUB 2Ø
73Ø Z$="OK$=":GOSUB 7Ø:H$="O.K.":GOSUB
 6Ø
8ØØ GRAPHICS Ø:SETCOLOR 2,Ø,Ø:PRINT "T
EIL  II : DATEN DES ADVENTURES":PRINT
: PRINT
8Ø5 Z$="VERBEN$=":GOSUB 7Ø
81Ø PRINT "WIE VIELE BUCHSTABEN SOLLEN
":PRINT "SIGNIFIKANT SEIN ";:INPUT WL
82Ø PRINT :PRINT "BENOETIGEN SIE MEHR
ALS WEITERE":PRINT "8Ø ZEICHEN ?":GOSU
B 9Ø:NOCHMAL=Ø
83Ø IF (EINGABE=74 OR EINGABE=2Ø2) THE
N NOCHMAL=-1
84Ø PRINT :PRINT "GEBEN SIE NUN IHREN
VERBEN$ EIN:"
```

```
85Ø INPUT H$
86Ø IF LEN(H$)>8Ø THEN GOTO 8ØØ
87Ø GOSUB 4Ø:GOSUB 7Ø:GOSUB 2Ø
88Ø IF  NOT NOCHMAL THEN GOTO 9ØØ
89Ø Z$="VERBEN$(LEN(VERBEN$)+1)=":GOSU
B 7Ø:GOTO 82Ø
9ØØ Z$="OBJEKTE$= ":GOSUB 7Ø
92Ø PRINT :PRINT "BENOETIGEN SIE FUER
DIE OBJEKTE":PRINT "MEHR ALS 8Ø ZEICHE
N ?":GOSUB 9Ø:NOCHMAL=Ø
93Ø IF (EINGABE=74 ØR EINGABE=2Ø2) THE
N NOCHMAL=-1
94Ø PRINT :PRINT "GEBEN SIE NUN IHREN
OBJEKTE$ EIN:"
95Ø INPUT H$
96Ø IF LEN(H$)>8Ø THEN GOTO 9ØØ
97Ø GOSUB 4Ø:GOSUB 7Ø:GOSUB 2Ø
98Ø IF  NOT NOCHMAL THEN GOTO 1ØØØ
99Ø Z$="VERBEN$(LEN(VERBEN$)+1)=":GOSU
B 7Ø:GOTO 92Ø
1ØØØ Z$="GOTO 7ØØ":GOSUB 2Ø
1Ø1Ø Z$="REM ********** RAUMBESCHREIB
UNGEN":ZEILENABSTAND=1:ZEILENNUMMER=2Ø
Ø:GOSUB 2Ø
1Ø2Ø GRAPHICS Ø:SETCOLOR 2,Ø,Ø:PRINT "
TEIL  II : RAEUME EINGEBEN":PRINT :PRI
NT
1Ø3Ø FOR I=1 TO AR
1Ø4Ø H$="":PRINT "ICH BIN ";:INPUT H$
1Ø45 PRINT "Dieser Raum fuehrt":PRINT
" im Norden nach Raum ";:INPUT D:DU(I,
1)=D
1Ø5Ø PRINT " im SUEDEN nach Raum ";:IN
PUT D:DU(I,2)=D
1Ø55 PRINT " im WESTEN nach Raum ";:IN
PUT D:DU(I,3)=D
1Ø6Ø PRINT " im OSTEN  nach Raum ";:IN
PUT D:DU(I,4)=D
1Ø65 PRINT " OBEN nach Raum      ";:IN
PUT D:DU(I,5)=D
1Ø7Ø PRINT " UNTEN nach Raum     ";:IN
```

```
PUT D:DU(I,6)=D
1Ø8Ø Z$="PRINT ":GOSUB 5Ø
1Ø9Ø NEXT I
11ØØ ZEILENNUMMER=ZEILENNUMMER+5Ø
111Ø Z$="REM ****************** GEGENS
TAENDE":ZEILENNUMMER=3ØØ:GOSUB 2Ø
112Ø GRAPHICS Ø:SETCOLOR 2,Ø,Ø:PRINT "
TEIL  II : OBJEKTE EINGEBEN":PRINT :PR
INT
113Ø FOR I=1 TO AO
114Ø PRINT "ICH SEHE ";:INPUT H$
115Ø PRINT :PRINT "STARTRAUM ";:INPUT
SR:OB(I)=SR
116Ø Z$="T$=":GOSUB 5Ø
117Ø NEXT I
119Ø ZEILENNUMMER=ZEILENNUMMER+5Ø
12ØØ GRAPHICS Ø:PRINT "TEIL  II : DATE
N DES ADVENTURES":PRINT :PRINT
121Ø PRINT " - OBJEKTE PLAZIEREN"
122Ø I=1
123Ø Z$="DATA "
124Ø H$=STR$(OB(I))
125Ø I=I+1
126Ø GOSUB 4Ø
127Ø IF LEN(H$)>1ØØ THEN 129Ø
128Ø IF I<AO+1 THEN H$=",":GOSUB 4Ø:GO
TO 124Ø
129Ø GOSUB 2Ø:IF I<AO+1 THEN 123Ø
13ØØ PRINT " - RAEUME VERBINDEN"
131Ø Z$="DATA ":KOMMA=Ø
132Ø FOR RA=1 TO AR
133Ø FOR RI=1 TO 6
134Ø IF KOMMA THEN H$=",":GOSUB 4Ø
135Ø H$=STR$(DU(RA,RI)):GOSUB 4Ø:KOMMA
=-1
136Ø IF LEN(Z$)>1ØØ THEN GOSUB 2Ø:Z$="
DATA ":KOMMA=Ø
137Ø NEXT RI
138Ø NEXT RA
139Ø IF LEN(Z$)>1 THEN GOSUB 2Ø
14ØØ PRINT " - TITEL: INSTRUKTIONEN":Z
```

```
$="REM ******* HIER 2. TITEL EINBAUEN"
:ZEILENNUMMER=699:GOSUB 2Ø
141Ø Z$="PRINT :PRINT :PRINT ":GOSUB 7
Ø:H$=" WUENSCHEN SIE RATSCHLAEGE FUER
IHR            WEITERES VORGEHEN"
142Ø GOSUB 4Ø:GOSUB 7Ø:H$=";:INPUT EIN
GABE$":GOSUB 4Ø:ZEILENNUMMER=7ØØ:GOSUB
 2Ø
143Ø Z$="IF EINGABE$(1,1)=":GOSUB 7Ø:H
$="J":GOSUB 4Ø:GOSUB 7Ø:H$="THEN GOSUB
 8ØØ":GOSUB 4Ø
1431 ZEILENNUMMER=798:GOSUB 2Ø
144Ø Z$="GOTO 9ØØ":ZEILENNUMMER=799:GO
SUB 2Ø:ZEILENABSTAND=5
145Ø Z$="GRAPHICS Ø:SETCOLOR 2,1,Ø:SET
COLOR 4,1,Ø":GOSUB 2Ø
146Ø Z$="PRINT ":GOSUB 7Ø:H$="
   ATARI - VENTURES         ":GOSUB 6Ø
147Ø Z$="PRINT ":GOSUB 7Ø:H$="     (C)
 1984 BY JOERG WALKOWIAK    ":GOSUB 4Ø
:GOSUB 2Ø
148Ø Z$="PRINT :PRINT ":GOSUB 7Ø:H$="S
tellen Sie Sich einen Roboter vor,   d
en Sie mit zahlreichen Kommandos"
149Ø GOSUB 6Ø
15ØØ Z$="PRINT ":GOSUB 7Ø:H$="steuern
koennen. Ich bin dieser Ro-   boter, u
nd ich werde mich fuer Sie":GOSUB 6Ø
151Ø Z$="PRINT ":GOSUB 7Ø:H$="den Gefa
hren der verwegensten Aben-   teuer au
ssetzen.":GOSUB 6Ø
152Ø Z$="PRINT ":GOSUB 7Ø:H$="Damit Si
e mich sinnvoll agieren las-  sen koen
nen, werde ich Ihnen die": GOSUB 6Ø
153Ø Z$="PRINT ":GOSUB 7Ø:H$="Situatio
n, in der ich mich gerade":GOSUB 6Ø
154Ø Z$="PRINT ":GOSUB 7Ø:H$="befinde,
 jeweils genau beschreiben.":GOSUB 6Ø
155Ø Z$="PRINT ":GOSUB 7Ø:H$="Anschlie
ssend sagen Sie mir mit zwei":GOSUB 6Ø
156Ø Z$="PRINT ":GOSUB 7Ø:H$="Worten w
```

```
ie zum Beispiel UNTERSUCHE":GOSUB 6Ø
157Ø Z$="PRINT ":GOSUB 7Ø:H$="TUER, NI
MM MESSER, was ich tun sOll.":GOSUB 6Ø
158Ø Z$="PRINT :PRINT ":GOSUB 7Ø:H$="D
arueber hinaus verstehe ich die Be-  f
ehle:":GOSUB 6Ø
159Ø Z$="PRINT ":GOSUB 7Ø:H$="
    SAVE, LOAD,":GOSUB 6Ø
16ØØ Z$="PRINT ":GOSUB 7Ø :H$="
 INVENTUR und ENDE.":GOSUB 6Ø
161Ø Z$="PRINT :PRINT ":GOSUB 7Ø:H$="B
ITTE DRUECKEN SIE (RETURN) UND ...":GO
SUB 4Ø:GOSUB 7Ø
162Ø H$=";:INPUT EINGABE$:RETURN":GOSU
B 4Ø:GOSUB 2Ø
17ØØ PRINT " - INITIALISIERUNGSTEIL":Z
EILENNUMMER=9ØØ:ZEILENABSTAND=1Ø
171Ø Z$="FOR I=1 TO AO:READ LAGEORT:OB
(I)=LAGEORT:NEXT I":GOSUB 2Ø
172Ø Z$="FOR I=1 TO AF:FL(I)=Ø:NEXT I"
:GOSUB 2Ø
173Ø Z$="FOR RA=1 TO AR:FOR RI=1 TO 6"
:GOSUB 2Ø
174Ø Z$="READ ZIEL:DU(RA,RI)=ZIEL":GOS
UB 2Ø
175Ø Z$="NEXT RI:NEXT RA":GOSUB 2Ø
18ØØ GRAPHICS Ø:PRINT "TEIL III : ADVE
NTURE TREIBER SCHREIBEN":PRINT :PRINT
181Ø ZEILENNUMMER=1ØØØ
182Ø Z$="GRAPHICS Ø:SETCOLOR 2,1,Ø:SET
COLOR 4,1,Ø:GOTO 1Ø3Ø":GOSUB 2Ø
183Ø Z$="REM ******** KLARTEXT: RICHTU
NGEN":ZEILENABSTAND=1:ZEILENNUMMER=1Ø2
Ø:PRINT " - RICHTUNGEN":GOSUB 2Ø
184Ø Z$="T$=":GOSUB 7Ø:H$="NORDEN, ":G
OSUB 4Ø:GOSUB 7Ø:H$=":RETURN":GOSUB 4Ø
:GOSUB 2Ø
185Ø Z$="T$=":GOSUB 7Ø:H$="SUEDEN. ":G
OSUB 4Ø:GOSUB 7Ø:H$=":RETURN":GOSUB 4Ø
:GOSUB 2Ø
186Ø Z$="T$=":GOSUB 7Ø:H$="WESTEN. ":G
```

```
OSUB 4Ø:GOSUB 7Ø:H$=":RETURN":GOSUB 4Ø
:GOSUB 2Ø
187Ø Z$="T$=":GOSUB 7Ø:H$="OSTEN, ":GO
SUB 4Ø:GOSUB 7Ø:H$=":RETURN":GOSUB 4Ø:
GOSUB 2Ø
188Ø Z$="T$=":GOSUB 7Ø:H$="OBEN, ":GOS
UB 4Ø:GOSUB 7Ø:H$=":RETURN":GOSUB 4Ø:G
OSUB 2Ø
189Ø Z$="T$=":GOSUB 7Ø:H$="UNTEN, ":GO
SUB 4Ø:GOSUB 7Ø:H$=":RETURN":GOSUB 4Ø:
GOSUB 2Ø
19ØØ ZEILENNUMMER=1Ø3Ø:ZEILENABSTAND=1
Ø:Z$="LZ$=":GOSUB 7Ø:H$="
                        ":GOSUB 6Ø
191Ø Z$="T$=":GOSUB 7Ø:GOSUB 7Ø:GOSUB
2Ø
192Ø Z$="ZE$=CHR$(3Ø):ZE$(LEN(ZE$)+1)=
CHR$(3Ø):ZE$(LEN(ZE$)+1)=":GOSUB 7Ø:H$
=".":GOSUB 6Ø
193Ø ZEILENNUMMER=1Ø8Ø:PRINT " - BILDS
CHIRMAUSGABE"
194Ø Z$="PRINT":GOSUB 2Ø
195Ø Z$="FOR Z=Ø TO 1Ø:POSITION 2,Z:PR
INT LZ$:NEXT Z":GOSUB 2Ø
196Ø Z$="POSITION 2,Ø:PRINT ":GOSUB 7Ø
:H$="ICH BIN ":GOSUB 4Ø:GOSUB 7Ø:H$=";
:GOSUB 2ØØ+SP":GOSUB 4Ø:GOSUB 2Ø
197Ø Z$="DZ$=":GOSUB 7Ø:H$="ICH SEHE "
:GOSUB 6Ø
198Ø Z$="FOR I=1 TO AO":GOSUB 2Ø
199Ø Z$="IF OB(I)<>SP THEN GOTO 117Ø":
GOSUB 2Ø
2ØØØ Z$="GD=-1:GOSUB 3ØØ+I:T$(LEN(T$)+
1)=":GOSUB 7Ø:H$=", ":GOSUB 6Ø
2Ø1Ø Z$="IF LEN(DZ$)+LEN(T$)>=38 THEN
PRINT DZ$:DZ$=T$(1,LEN(T$)):T$=":GOSUB
 7Ø:GOSUB 7Ø:H$=":GOTO 117Ø"
2Ø2Ø GOSUB 4Ø:GOSUB 2Ø
2Ø3Ø Z$="DZ$(LEN(DZ$)+1)=T$:T$=":GOSUB
 7Ø:GOSUB 7Ø:GOSUB 2Ø
2Ø4Ø Z$="NEXT I":GOSUB 2Ø
```

```
2Ø5Ø Z$="IF  NOT GD THEN DZ$(LEN(DZ$)+
1)=":GOSUB 7Ø:H$="NICHTS BESONDERES. "
:GOSUB 6Ø
2Ø6Ø Z$="DZ$(LEN(DZ$)+1)=ZE$":GOSUB 2Ø
2Ø7Ø Z$="PRINT DZ$":GOSUB 2Ø
2Ø8Ø Z$="PRINT LZ$":GOSUB 2Ø
2Ø9Ø Z$="DZ$=":GOSUB 7Ø:H$="ICH KANN N
ACH ":GOSUB 4Ø:GOSUB 7Ø:H$=":GD=Ø":GOS
UB 4Ø:GOSUB 2Ø
21ØØ Z$="FOR RI=1 TO 6":GOSUB 2Ø
211Ø Z$="IF DU(SP,RI)=Ø THEN GOTO 128Ø
":GOSUB 2Ø
212Ø Z$="GOSUB 1Ø2Ø+RI:GD=-1":GOSUB 2Ø
213Ø Z$="IF LEN(DZ$)+LEN(T$)>=38 THEN
PRINT DZ$:DZ$=T$(1,LEN(T$)):T$=":GOSUB
 7Ø:GOSUB 7Ø:H$=":GOTO 128Ø"
2135 GOSUB 4Ø:GOSUB 2Ø
214Ø Z$="DZ$(LEN(DZ$)+1)=T$:T$=":GOSUB
 7Ø:GOSUB 7Ø:GOSUB 2Ø
2142 Z$="NEXT RI":GOSUB 2Ø
2144.Z$="IF  NOT GD THEN DZ$(LEN(DZ$)+
1)=":GOSUB 7Ø:H$="NIRGENDWO":GOSUB 6Ø
2146 Z$="DZ$(LEN(DZ$)+1)=ZENDE$
215Ø Z$="PRINT DZ$":GOSUB 2Ø
216Ø Z$="PRINT ":GOSUB 7Ø:H$="
":GOSUB 6Ø
217Ø Z$="REM *** HIER TEST AUF SPIELEN
DE ****":GOSUB 2Ø
218Ø Z$= "REM *** ODER FALLEN U.A. ***
*******":GOSUB 2Ø
219Ø Z$="REM *** FALLEN AUCH IN 1391 B
IS 1399":GOSUB 2Ø
22ØØ PRINT " - SPIELER BEWEGEN":ZEILEN
NUMMER=139Ø
221Ø Z$="POSITION 2,23:PRINT ":GOSUB 7
Ø:H$="WAS SOLL ICH TUN ":GOSUB 4Ø:GOSU
B 7Ø:H$=";:INPUT EINGABE$":GOSUB 4Ø
2215 GOSUB 2Ø
222Ø Z$="IF LEN(EINGABE$)>2 THEN 148Ø"
223Ø Z$="IF EINGABE$=":GOSUB 7Ø:H$="N"
```

```
:GOSUB 4Ø:GOSUB 7Ø:H$="AND DU(SP,1)<>Ø
 THEN SP=DU(SP,1):PRINT OK$:GOTO 1Ø8Ø"
224Ø GOSUB 95
225Ø Z$="IF EINGABE$=":GOSUB 7Ø:H$="S"
:GOSUB 4Ø:GOSUB 7Ø:H$="AND DU(SP,2)<>Ø
 THEN SP=DU(SP,2):PRINT OK$:GOTO 1Ø8Ø"
226Ø GOSUB 95
227Ø Z$="IF EINGABE$=":GOSUB 7Ø:H$="W"
:GOSUB 4Ø:GOSUB 7Ø:H$="AND DU(SP,3)<>Ø
 THEN SP=DU(SP,3):PRINT OK$:GOTO 1Ø8Ø"
228Ø GOSUB 95
229Ø Z$="IF EINGABE$=":GOSUB 7Ø:H$="O"
:GOSUB 4Ø:GOSUB 7Ø:H$="AND DU(SP,4)<>Ø
 THEN SP=DU(SP,4):PRINT OK$:GOTO 1Ø8Ø"
23ØØ GOSUB 95
231Ø Z$="IF EINGABE$=":GOSUB 7Ø:H$="OB
":GOSUB 4Ø:GOSUB 7Ø:H$="AND DU(SP,5)<>
Ø THEN SP=DU(SP,5):PRINT OK$:GOTO 1Ø8Ø
"
232Ø GOSUB 95
233Ø Z$="IF EINGABE$=":GOSUB 7Ø:H$="U"
:GOSUB 4Ø:GOSUB 7Ø:H$="AND DU(SP,6)<>Ø
 THEN SP=DU(SP,6):PRINT OK$:GOTO 1Ø8Ø"
234Ø GOSUB 95
2345 Z$="IF LEN(EINGABE$)<3 THEN PRINT
":GOSUB 7Ø:H$="DAHIN FUEHRT KEIN WEG !
":GOSUB 4Ø:GOSUB 7Ø:H$=":GOTØ1Ø8Ø"
2346 GOSUB 95
235Ø Z$="IF LEN(EINGABE$)>8 THEN GOTO
2ØØØ":GOSUB 2Ø
24ØØ PRINT " - INVENTUR":ZEILENNUMMER=
15ØØ:ZEILENABSTAND=5
241Ø Z$="IF EINGABE$(1,3)<>":GOSUB 7Ø:
H$="INV":GOSUB 4Ø:GOSUB 7Ø:H$=" THEN G
OTO 16ØØ":GOSUB 95
242Ø Z$="PRINT ":GOSUB 7Ø:H$="ICH TRAG
E FOLGENDES MIT MIR:":GOSUB 6Ø
243Ø Z$="FOR I=1 TO AO":GOSUB 2Ø
244Ø Z$="IF OB(I)=-1 THEN GOSUB 3ØØ+I:
PRINT T$":GOSUB 2Ø
245Ø Z$="NEXT I":GOSUB 2Ø
```

```
246Ø Z$="GOTO 1Ø8Ø":GOSUB 2Ø
25ØØ PRINT " - SAVE GAME":ZEILENNUMMER
=16ØØ
251Ø Z$="IF EINGABE$(1,3)<>":GOSUB 7Ø:
H$="SAV":GOSUB 4Ø:GOSUB 7Ø:H$=" THEN G
OTO 17ØØ":GOSUB 4Ø:GOSUB 2Ø
252Ø Z$="PRINT ":GOSUB 7Ø:H$="UNTER WE
LCHEM NAMEN ":GOSUB 4Ø:GOSUB 7Ø
2525 H$=";:INPUT EINGABE$:IF LEN(EINGA
BE$)>8 THEN PRINT"
253Ø GOSUB 4Ø:GOSUB 7Ø:H$="BITTE ETWAS
 KUERZER !":GOSUB 4Ø:GOSUB 7Ø:H$=":GOT
O 16Ø5":GOSUB 95
254Ø Z$="T$=":GOSUB 7Ø:H$="D:":GOSUB 4
Ø:GOSUB 7Ø:H$=":EINGABE$(LEN(EINGABE$)
+1)=":GOSUB 4Ø:GOSUB 7Ø
2545 H$=".DAT":GOSUB 6Ø
255Ø Z$="T$(LEN(T$)+1)=EINGABE$":GOSUB
 2Ø
256Ø Z$="OPEN #1,8,Ø,T$":GOSUB 2Ø
257Ø Z$="PRINT #1,SP":GOSUB 2Ø
258Ø Z$="FOR I=1 TO AO:PRINT #1,ØB(I):
NEXT I":GOSUB 2Ø
259Ø Z$="FOR RA=1 TO AR":GOSUB 2Ø
26ØØ Z$="FOR RI=1 TO 6":GOSUB 2Ø
261Ø Z$="PRINT #1,DU(RA,RI)"
262Ø Z$="NEXT RI":GOSUB 2Ø
263Ø Z$="NEXT RA":GOSUB 2Ø
264Ø Z$="FOR I=1 TO AF:PRINT #1,FL(I):
NEXT I":GOSUB 2Ø
265Ø Z$="CLOSE #1:PRINT OK$:GOTO 1Ø8Ø"
:GOSUB 2Ø
27ØØ PRINT " - LOAD GAME":ZEILENNUMMER
=17ØØ
271Ø Z$="IF EINGABE$(1,3)<>":GOSUB 7Ø:
H$="LOA":GOSUB 4Ø:GOSUB 7Ø:H$=" THEN G
OTO 19ØØ":GOSUB 95
272Ø Z$="PRINT ":GOSUB 7Ø:H$="WELCHES
SPIEL ":GOSUB 4Ø:GOSUB 7Ø:H$=";:INPUT
EINGABE$:IF LEN(EINGABE$)>8 THEN PRINT
"
```

```
273Ø GOSUB 4Ø:GOSUB 7Ø:H$="DAS KANN ES
 NICHT GEBEN !":GOSUB 4Ø:GOSUB 7Ø:H$="
:GOTO 17Ø5":GOSUB 95
274Ø Z$="T$=":GOSUB 7Ø:H$="D:":GOSUB 4
Ø:GOSUB 7Ø:H$=":EINGABE$(LEN(EINGABE$)
+1)=":GOSUB 4Ø:GOSUB 7Ø
2745 H$=".DAT":GOSUB 6Ø
275Ø Z$="T$(LEN(T$)+1)=EINGABE$":GOSUB
 2Ø
276Ø Z$="OPEN #1,4,Ø,T$":GOSUB 2Ø
277Ø Z$="INPUT #1,SP":GOSUB 2Ø
278Ø Z$="FOR I=1 TO AO:INPUT #1,LO:OB(
I)=LO:NEXT I":GOSUB 2Ø
279Ø Z$="FOR RA=1 TO AR":GOSUB 2Ø
28ØØ Z$="FOR RI=1 TO 6":GOSUB 2Ø
281Ø Z$="INPUT #1,ZIEL:DU(RA,RI)=ZIEL"
:GOSUB 2Ø
282Ø Z$="NEXT RI":GOSUB 2Ø
283Ø Z$="NEXT RA":GOSUB 2Ø
284Ø Z$="FOR I=1 TO AF:INPUT #1,ZIEL:F
L(I)=ZIEL:NEXT I":GOSUB 2Ø
285Ø Z$="CLOSE #1:PRINT OK$:GOTO 1Ø8Ø"
:GOSUB 2Ø
29ØØ PRINT " - INS und ENDE":ZEILENNUM
MER=19ØØ
291Ø Z$="IF EINGABE$(1,3)<>":GOSUB 7Ø:
H$="INS":GOSUB 4Ø:GOSUB 7Ø:H$=" THEN G
OTO 191Ø" :GOSUB 95
292Ø Z$="GOSUB 8ØØ:GOTO 1Ø8Ø":GOSUB 2Ø
293Ø Z$="IF EINGABE$(1,3)<>":GOSUB 7Ø:
H$="END":GOSUB 4Ø:GOSUB 7Ø:H$=" THEN G
OTO 2ØØØ":GOSUB 95
294Ø Z$="GRAPHICS Ø:PRINT ":GOSUB 7Ø:H
$="DER AUTOR WUENSCHT IHNEN MEHR GLUEC
K":GOSUB 4Ø:GOSUB 7Ø
2941 H$=":PRINT ":GOSUB 4Ø
3ØØØ GOSUB 7Ø:H$="FUERS NAECHSTE MAL !
":GOSUB 4Ø:GOSUB 7Ø:H$=":PRINT :PRINT
:PRINT :END":GOSUB 95
3Ø1Ø PRINT " - EINGABEANALYSE":ZEILENN
UMMER=2ØØØ:ZEILENABSTAND=1Ø
```

```
3Ø2Ø Z$="LN=LEN(EINGABE$)":GOSUB 2Ø
3Ø3Ø Z$="FOR I=1 TO LN":GOSUB 2Ø
3Ø4Ø Z$="IF EINGABE$(I,I)<>":GOSUB 7Ø:
H$=" ":GOSUB 4Ø:GOSUB 7Ø:H$=" THEN NEX
T I":GOSUB 95
3Ø5Ø Z$="EV$=EINGABE$(1,I)":GOSUB 2Ø
3Ø6Ø Z$="IF LEN(EV$)=LN-1 THEN GOTO 2Ø
9Ø":GOSUB 2Ø
3Ø7Ø Z$="EO$=EINGABE$(I+1,LN)":GOSUB 2
Ø
3Ø8Ø Z$="VN=Ø:N=Ø":GOSUB 2Ø
3Ø9Ø Z$="EV$=EV$(1,":X=WL:GOSUB 8Ø:H$=
")":GOSUB 4Ø:GOSUB 2Ø
31ØØ Z$="EO$=EO$(1,":X=WL:GOSUB
8Ø:H$=")":GOSUB 4Ø:GOSUB 2Ø
311Ø Z$="FOR I=1 TO LEN(VERBEN$) STEP
":GOSUB 8Ø:GOSUB 2Ø
312Ø Z$="VN=VN+1":GOSUB 2Ø
313Ø X=WL-1:Z$="IF VERBEN$(I,I+":GOSUB
 8Ø:H$=:")=EV$ THEN 214Ø":GOSUB 95
314Ø Z$="NEXT I":GOSUB 2Ø
315Ø Z$="PRINT ":GOSUB 7Ø:H$="ICH VERS
TEHE DAS VERB NICHT !":GOSUB 4Ø:GOSUB
7Ø:H$=":GOTO 1Ø8Ø":GOSUB 95
316Ø X=WL:Z$="FOR I=1 TO LEN(OBJEKTE$)
 STEP ":GOSUB 8Ø:GOSUB 2Ø
317Ø Z$="N=N+1":GOSUB 2Ø
318Ø X=WL-1:Z$="IF OBJEKTE$(I,I+":GOSU
B 8Ø:H$=")=EO$ THEN 22ØØ":GOSUB 95
319Ø Z$="NEXT I":GOSUB 2Ø
32ØØ Z$="PRINT ":GOSUB 7Ø:H$="ICH VERS
TEHE DAS OBJEKT NICHT !":GOSUB 4Ø:GOSU
B 7Ø:H$=":GOTO 1Ø8Ø":GOSUB 95
321Ø PRINT " - SPRUNGTABELLE":ZEILENNU
MMER=22ØØ:X=4ØØØ:Z$="ON VN GOTO"
322Ø FOR I=1 TO AV
323Ø X=X+1ØØØ:GOSUB 8Ø
323Ø IF I<>AV THEN H$=",":GOSUB 4Ø
324Ø NEXT I
3245 GOSUB 2Ø
325Ø PRINT " - TITEL: SPIELER TOD":ZEI
```

```
LENNUMMER=45ØØ
326Ø Z$="GRAPHICS Ø":GOSUB 2Ø
327Ø Z$="PRINT ":GOSUB 7Ø:H$="AUCH DAS
 NOCH !":GOSUB 7Ø:H$=":PRINT :PRINT MØ
$":GOSUB 95
328Ø Z$="PRINT :PRINT ":GOSUB 7Ø:H$="I
CH BIN TOT !":GOSUB 4Ø:GOSUB 7Ø:H$=":P
RINT":GOSUB 95
329Ø Z$="PRINT ":GOSUB 7Ø:H$="SOLL ICH
 ES NOCH EINMAL VERSUCHEN ":GOSUB 4Ø:G
OSUB 7Ø:H$=";:INPUT EINGABE$":GOSUB 95
33ØØ Z$="IF EINGABE$(I,1)=":GOSUB 7Ø:H
$="J":GOSUB 4Ø:GOSUB 7Ø:H$=" THEN CLR
:GOTO 1ØØ":GOSUB 95
331Ø Z$="GOTO 196Ø":GOSUB 2Ø
332Ø PRINT " - TITEL: SIEG":ZEILENNUMM
ER=48ØØ
333Ø Z$="GRAPHICS Ø":GOSUB 2Ø
334Ø Z$="PRINT ":GOSUB 7Ø:H$="HERZLICH
EN GLUECKWUNSCH !":GOSUB 6Ø
335Ø Z$="PRINT :PRINT: PRINT ":GOSUB 7
Ø:H$="SIE HABEN DIE IHNEN GESTELLTE AU
FGABE":GOSUB 4Ø:GOSUB 2Ø
336Ø Z$="PRINT :PRINT ":GOSUB 7Ø:H$="G
ELOEST UND DUERFEN SICH AN EINEM":GOSU
B 4Ø:GOSU8 2Ø
337Ø Z$="PRINT :PRINT":GOSUB 7Ø:H$="AN
DEREM ADVENTURE VERSUCHEN.":GOSUB 4Ø:G
OSUB 2Ø
338Ø Z$="PRINT :PRINT :PRINT :END":GOS
UB 2Ø
339Ø ZEILENNUMMER=49ØØ:Z$="PRINT ":GOS
UB 7Ø:H$="ACHTUNG FEHLER !":GOSUB 4Ø:G
OSUB 7Ø:H$=":GOTO 1Ø8Ø":GOSUB 95
4ØØØ M$="TEIL  IV : BEDINGUNGEN & AKTI
ONEN":ZEILENABSTAND=1:Z1=5ØØØ:ZEILENNU
MMER=Z1:O=Ø:V=Ø
4ØØ5 V=V+1
4Ø1Ø O=O+1
4Ø15 X=O:Z$="IF N=":GOSUB 8Ø:H$=" AND
":GOSUB 4Ø
```

```
4Ø2Ø MØ$="TEIL  IV : BEDINGUNGEN":M1$=
"
"
4Ø3Ø GRAPHICS Ø:SETCOLOR 2,Ø,Ø:PRINT M
Ø$:PRINT M1$
4Ø4Ø PRINT " 1 - OBJEKT IST IM RAUM"
4Ø5Ø PRINT " 2 - OBJEKT IST NICHT IM R
AUM"
4Ø6Ø PRINT " 3 - SPIELER HAT OBJEKT"
4Ø7Ø PRINT " 4 - FLAG IST GESETZT"
4Ø8Ø PRINT " 5 - FLAG IST NICHT GESETZ
T"
4Ø9Ø PRINT " 6 - SPIELER muß SEIN IN
RAUM:"
4Ø94 PRINT " 7 - UND WEITERE BEDINGUNG
"
4Ø97 PRINT " 8 - ODER WEITERE BEDINGUN
G"
41ØØ PRINT :PRINT " Ø - BEDINGUNGEN O.
K."
411Ø PRINT M1$
412Ø PRINT "WAEHLEN SIE FUER VERB ";V;
" OBJEKT ";Ø
413Ø INPUT E
414Ø IF E=1 THEN H$="OB(N)=SP ":GOSUB
4Ø
415Ø IF E=2 THEN H$="OB(N)<>SP ":GOSUB
 4Ø
416Ø IF E=3 THEN PRINT "OBJEKT NUMMER
":INPUT X:H$="OB(":GOSUB 4Ø:GOSUB 8Ø:H
$=")=SP":GOSUB 4Ø
417Ø IF E=4 THEN PRINT "FLAG NUMMER ":
INPUT X:H$="FL(":GOSUB 4Ø:GOSUB 8Ø:H$=
")<>-1":GOSUB 4Ø
418Ø IF E=5 THEN PRINT "FLAG NUMMER ":
INPUT X:H$="FL(":GOSUB 4Ø:GOSUB 8Ø:H$=
")<>-1":GOSUB 4Ø
419Ø IF E=6 THEN PRINT "WELCHEM RAUM "
:INPUT X:H$="SP":GOSUB 4Ø
42ØØ IF E=7 THEN H$=" AND ":GOSUB 4Ø
421Ø IF E=8 THEN H$=" OR ":GOSUB 4Ø
```

```
422Ø IF E<>Ø THEN 4Ø3Ø
4225 H$="THEN":GOSUB 4Ø
43ØØ MØ$="TEIL  IV : AKTIONEN"
431Ø GRAPHICS Ø:SETCOLOR 2,Ø,Ø:PRINT M
Ø$:PRINT M1$
432Ø PRINT " 1 - OBJEKT VERSCHWINDET"
433Ø PRINT " 2 - OBJEKT INS INVENTORV"
434Ø PRINT " 3 - OBJEKT ERSCHEINT NEU"
435Ø PRINT " 4 - FLAG WI RD GESETZT"
436Ø PRINT " 5 - FLAG WIRD GELOESCHT"
437Ø PRINT " 6 - DURCHGANG OEFFNEN"
438Ø PRINT " 7 - MITTEILUNG AUSGEBEN"
439Ø PRINT " 8 - SPIELER STIRBT"
44ØØ PRINT " 9 - SPIELER SIEGT"
441Ø PRINT :PRINT " Ø - AKTIONEN O.K."
442Ø PRINT M1$:PRINT "WAEHLEN SIE FUER
 VERB ";V;" OBJEKT ";O:INPUT E
443Ø IF E=1 THEN PRINT "OBJEKT NUMMER
";:INPUT X:H$="OB(":GOSUB 4Ø:GOSUB 8Ø:
H$=")=Ø":GOSUB 4Ø
444Ø IF E=2 THEN PRINT "OBJEKT NUMMER
";:INPUT X:H$="OB(":GOSUB 4Ø:GOSUB 8Ø:
H$=")=-1":GOSUB 4Ø
445Ø IF E=3 THEN PRINT "OBJEKT NUMMER
";:INPUT X:H$="OB(":GOSUB 4Ø:GOSUB 8Ø:
H$="):=SP":GOSUB 4Ø
446Ø IF E=4 THEN PRINT "FLAG NUMMER ";
:INPUT X:H$="FL(":GOSUB 4Ø:GOSUB 8Ø:H$
=")=-1":GOSUB 4Ø
447Ø IF E=5 THEN PRINT "FLAG NUMMER ";
:INPUT X:H$="FL(":GOSUB 4Ø:GOSUB 8Ø:H$
=")=Ø":GOSUB 4Ø
448Ø IF E=6 THEN H$="DU(":GOSUB 4Ø:PRI
NT "VON RAUM ";:INPUT X:GOSUB 8Ø:H$=",
":GOSUB 4Ø
4481 PRINT "IN RICHTUNG ";:INPUT X
4485 IF E=6 THEN GOSUB 8Ø:H$=")=":GOSU
B 4Ø:PRINT "NACH RAUM ";:INPUT X:GOSUB
 8Ø
449Ø IF E=7 THEN H$="PRINT":GOSUB 4Ø:G
OSUB 7Ø:PRINT "TEXT ";:INPUT H$:GOSUB
```

```
4Ø:GOSUB 7Ø
45ØØ IF E=8 THEN H$="MØ$=":GOSUB 7Ø:PR
INT "TEXT ";:INPUT H$:GOSUB 4Ø:GOSUB 7
Ø:H$="GOTO 45ØØ":GOSUB 4Ø
451Ø IF E=9 THEN H$="GOTO 48ØØ":GOSUB
4Ø
452Ø IF E<>Ø THEN H$=":":GOSUB 4Ø:GOTO
 431Ø
4525 H$="GOTO 1Ø8Ø":GOSUB 95
4527 PRINT "MEHR ZUM GLEICHEN OBJEKT ?
":GOSUB 9Ø:IF EINGABE=74 THEN GOTO 4Ø1
5
453Ø IF O<>AO THEN GOTO 4Ø1Ø
454Ø O=Ø:Z1=Z1+1ØØØ:ZEILENNUMMER=Z1
455Ø IF V<>AV THEN GOTO 4ØØ5
1ØØØØ CLOSE #1:GRAPHICS Ø:SETCOLOR 2,Ø
,Ø:PRINT "LADEN SIE DAS FERTIGE ADVENT
URE NUN":PRINT "MIT:"
1ØØØ1 PRINT "ENTER";CHR$(34);ADVENTURE
$:PRINT :PRINT "VIEL VERGNUEGEN !":NEW
```

6. ANHANG

PROGRAMMVERBESSERUNGEN

Für Besitzer des ATARI 6ØØ XL wird es unvermeidlich sein, die in diesem Buch veröffentlichten Programme in Hinblick auf eine Speicherplatzoptimierung zu überarbeiten. Aber auch Besitzer eines ATARI 8ØØ XL können Nutzen aus den folgenden Vorschlägen ziehen.

Natürlich gilt das hier Gesagte nicht nur für unsere Adventures, sondern für alle BASIC-Programme !

SPEICHERPLATZOPTIMIERUNG

Achten Sie darauf, die einzelnen Programmzeilen möglichst zu füllen; verwenden sie somit die geringstmögliche Anzahl von Zeilen.
Denn jede einzelne Anweisungszeile belegt mehrere Bytes für die eigene Zeilennummer, zusätzlich müssen Daten zur Auffindung der nächsten Programmzeile abgelegt werden.

Wenn Sie die erlaubten Abkürzungen verwenden, können Sie sogar im Listing überlang erscheinende Zeilen eingeben !

Verzichten Sie auf alle REMarks !

Benutzen Sie statt der von uns zur Erhöhung der Lesbarkeit verwandten langen Variablennamen nur ein oder zwei Buchstaben lange Bezeichnungen !

ABLAUFOPTIMIERUNG

Folgende Vorschläge helfen dem BASIC Interpreter, Ihre Programme möglichst schnell ablaufen zu lassen. Natürlich tragen auch die zuvor gemachten Vorschläge zu einem Speed-Up bei.

Der einzige Weg zu diesem Ziel zwingt uns dazu, auf die Belange des Interpreters Rücksicht zu nehmen.

Dieser erstellt während seiner Arbeit eine Reihe von Listen, wobei die Reihenfolge der einzelnen Elemente von der Reihenfolge ihres Eintragens in die jeweilige Liste abhängig ist.
So wird jeder Variablen in der Reihenfolge ihres Auftretens Platz in dem dafür vorgesehenem Abschnitt des Programmspeichers zugewiesen.
Greift der Rechner nun zum Lesen oder Schreiben auf eine bestimmte Variable zu, so muß er alle Elemente dieser Liste durchgehen, bis die geforderte Variable erreicht worden ist.
Deshalb ist es zweckmäßig, die Variablen in der Reihenfolge ihrer Häufigkeit eintragen zu lassen, was durch eine Programmzeile, in der diese Variablen beispielsweise auf null gesetzt werden, geschehen kann.

Außerdem sollten Sie auch häufig benutzte Konstanten zuvor als Variablen definieren, denn diese festen Zahlen müssen bei jeder einzelnen Operation auf das Verarbeitungsformat des ATARI umgerechnet werden, was ansonsten nur einmal bei der Definition geschieht.
Achten Sie insbesondere auf Schleifen !

Unser zweiter Ansatzpunkt betrifft die Sprünge, die ein Programm ausführt. Denn woher weiß der Interpreter, an welcher Stelle im Speicher sich das Sprungziel befindet ?

Ganz einfach, er weiß es nicht, sondern er geht alle Zeilen durch, bis er die vorgegebene Zeile erreicht hat.
Legen Sie daher alle häufig benutzten Subroutinen an den Beginn Ihrer Programme, um die Suchzeit, auch wenn es sich nur um winzige Bruchteile von Sekunden handelt, zu verkürzen !

STICHWORTVERZEICHNIS

A

ABENTEUER.....9, 11
ADAMS, SCOTT.....16, 17
ADVENTURE.....
  FUNKTIONEN.....33
  GRAFIK.....2Ø, 157
  LABYRINTH.....2Ø
  QUEST.....2Ø
ADVENTURELAND.....16, 17
ADVENTURES.....12
ADVENTUREWELT.....29
AKTIONEN.....84, 94
ARCADE.....12
ARRAY.....48
AUSGABEFORMATIERUNG.....55
AVAILABLE LIGHT.....15Ø

B

BEDINGUNG.....82, 94
BENUTZERFREUNDLICHKEIT...98, 1Ø9

C

COLOSSAL CAVE.....17

D

DATA.....49
DATEI.....
  RELATIVE.....112
  SEQUENTIELLE.....112
DATENSPEICHERUNG.....112
DURCHGÄNGE.....47, 144

E

EINGABEANALYSE.....70
EINGABEN.....31, 166
ELIZA.....17

F

FELD.....48, 82
FILE.....114
FLAG.....88
FLOPPY.....112

G

GOLDRAUSCH.....195
GRAFIK PROGRAMMER.....241
GRAFIK-ADVENTURE.....16
GRAFIKADVENTURE.....157
GRAFIKINTERPRETER.....157

H

HELP.....126
HILFE.....33

I

INTELLIGENZ, KÜNSTLICHE.....17
INTERACTIVE FICTION.....18
INVENTUR.....33, 95
IRRGARTEN.....137

K

KARTE.....41

L

LABYRINTH.....137
LAGERRAUM.....66
LICHT, VERFÜGBARES.....150
LICHTSCHALTER.....152
LIMIT.....147
LOAD GAME.....111, 115
LÖSUNGSHINWEISE.....170

M

MICROSOFT ADVENTURE.....17
MITTEILUNGEN.....32

O

OBJEKT.....3Ø, 63
  BESCHREIBUNG.....65
  KURZNAMEN.....65
OPERATOR, LOGISCHER.....84
ORTSWECHSEL.....142

P

PRINT AT POSITION.....56
PUNKTE.....129

R

RAUMBESCHREIBUNG.....45
RÄUME.....29
READ.....49
RECORDER.....112
RICHTUNGSTABELLE.....51
RUFNAME.....65

S

SAMMELRAUM.....139
SAVE GAME.....111, 113
SCHLOSS-ADVENTURE.....173
SCORE.....13Ø
SPIELIDEE.....39
STEUERCODES.....56
STORE ROOM.....66
STRING.....59, 7Ø
STRUKTOGRAMM.....57
SYNONYME.....119

T

TIPPFEHLER.....37
TITEL.....96
TOKEN.....56
TRAVEL-TABLE.....48
TREIBER.....73

V

VARIABLEN.....47
VERBANALYSE.....71
VOKABULAR.....122

W

WAGENRÜCKLAUF.....57
WEIZENBAUM, JOSEPH.....17
WERTUNG.....129
WORTSCHATZ.....118

Z

ZÄHLER.....147
ZUFÄLLE.....156
ZUGÄNGE.....144

DAS STEHT DRIN:

Ein faszinierender Führer in die phantastische Welt der Abenteur-Spiele. Hier wird gezeigt wie Adventures funktionieren, wie man sie erfolgreich spielt und wie man eigene Adventures auf ATARI Computern der Serie XL programmiert. Dabei wird das gesamte Spektrum bis hin zum trickreichen Grafikadventure abgehandelt und mit vielen Programmbeispielen belegt.

Der Clou des Buches ist, neben vielen fertigen Adventures zum Abtippen, ein kompletter ADVENTURE-GENERATOR. Mit dem das Selberprogrammieren packender Adventures zum Kinderspiel wird.

Zusätzlich wird ein Grafik-Editor und -Programmierer vorgestellt, mit dessen Hilfe Sie im Handumdrehen die für Ihre Grafikadventures notwendigen Unterprogramme erzeugen können.

UND GESCHRIEBEN HAT DIESES BUCH:

Jörg Walkowiak, Informatik-Student, der auf jahrelange Erfahrung mit den gängisten Heim- und Personalcomputern zurückblicken kann und erfolgreicher Autor professioneller Adventures ist.

ISBN 3-89011-059-2

Walkowiak/Adventures – und wie man sie auf dem ATARI ® programmiert