To: Dungeon Players
From: "The Translator"
Subj: Game Information
Date: 20-AUG-78

I. Installation Instructions

Dungeon is provided as a ready to run executable image and two data files:

-DUNGEO.SAV (or DUNGEON.TSK)
-DTEXT.DAT
-DINDX.DAT

The executable image may be installed in any directory on any disk. The data files must be installed in the user's default directory on physical or logical disk SY:.

Because of program size, the RT-11 version requires a 28K system and the baseline or single job monitor, while the RSX-11M version requires a 32K partition.

II. Startup Procedure

Once installed, Dungeon is started with the command:

RUN DUNGEON(cr)

Dungeon takes 3-5 seconds to initialize itself and then prompts for user input.

III. Warnings and Restrictions

Dungeon includes a debugging tool, the use of which is protected by a challenge/response validation procedure. Should you accidentally invoke the debugging tool and be challenged to provide the proper response, you may escape without penalty by typing (cr). Providing an improper response is fatal.

For those familiar with the MDL version of the game on the ARPAnet, the following is a list of the major incompatabilties:

-There is no endgame.
-The first six letters of a word are considered significant, instead of the first five.
-AGAIN, NOOBJ, ROOM, OBJECTS, INCANT, ANSWER, BUG, and FEATURE are not currently implemented.
-The syntax for TELL is different.
-Compound objects are not recognized.
-Compound commands must be delimited with comma rather than the word AND.

IV. Abstract of Informational Printouts

SUMMARY

Welcome to Dungeon!

Dungeon is a game of adventure, danger, and low cunning. In it you will explore some of the most amazing territory ever seen by mortal man. Hardened adventurers have run screaming from the terrors contained within.

In Dungeon, the intrepid explorer delves into the forgotten secrets of a lost labyrinth deep in the bowels of the earth, searching for vast treasures long hidden from prying eyes, treasures guarded by fearsome monsters and diabolical traps!

No DECsystem should be without one!

Dungeon was created at the Programming Technology Division of the MIT Laboratory for Computer Science by Tim Anderson, Marc Blank, Bruce Daniels, and Dave Lebling. It was inspired by the Adventure game of Crowther and Woods, and the Dungeons and Dragons game of Gygax and Arneson. The original version was written in MDL (alias MUDDLE). The current version was translated from MDL into FORTRAN IV by a somewhat paranoid DEC engineer who prefers to remain anonymous.

INFO

Welcome to Dungeon!

You are near a large dungeon, which is reputed to contain vast quantities of treasure. Naturally, you wish to acquire some of it. In order to do so, you must of course remove it from the dungeon. To receive full credit for it, you must deposit it safely in the trophy case in the living room of the house.

In addition to valuables, the dungeon contains various objects which may or may not be useful in your attempt to get rich. You may need sources of light, since dungeons are often dark, and weapons, since dungeons often have unfriendly things wandering about. Reading material is scattered around the dungeon as well; some of it is rumored to be useful.

To determine how successful you have been, a score is kept. When you find a valuable object and pick it up, you receive a certain number of points, which depends on the difficulty of finding the object. You receive extra points for transporting the treasure safely to the living room and placing it in the trophy case. In addition, some particularly interesting rooms have a value associated with visiting them. The only penalty is for getting yourself killed, which you may do only twice.

Of special note is a thief (always carrying a large bag) who likes to wander around in the dungeon (he has never been seen by the light of day). He likes to take things. Since he steals for pleasure rather than profit and is somewhat sadistic, he only takes things which you have seen. Although he prefers valuables, sometimes in his haste he may take something which is worthless. From time to time, he examines his take and discards objects which he doesn't like. He may occasionally stop in a room you are visiting, but more often he just wanders through and rips you off (he is a skilled pickpocket).

HELP

Useful commands:

The 'BRIEF' command suppresses printing of long room descriptions for rooms which have been visited. The 'SUPERBRIEF' command suppresses printing of long room descriptions for all rooms. The 'VERBOSE' command restores long descriptions.

The 'INFO' command prints information which might give some idea of what the game is bout.

The 'QUIT' command prints your score and asks whether you wish to continue playing.

The 'SAVE' command saves the state of the game for later continuation.

The 'RESTORE' command restores a saved game.

The 'INVENTORY' command lists the objects in your possession.

The 'LOOK' command prints a description of your surroundings.

The 'SCORE' command prints your current score and rankings.

The 'TIME' command tells you how long you have been playing.

The 'DIAGNOSE' command reports on your injuries, if any.

Containment:

Some objects can contain other objects. Many such containers can be opened and closed. The rest are always open. They may or may not be transparent. For you to access (e.g., take) an object which is in a container, the container must be open. For you to see such an object, the container must be either open or transparent. Containers have a capacity, and objects have sizes; the number of objects which will fit therefore depends on their sizes. You may PUT any object you have access to (it need not be in your hands) into any other object. At some point, the program will attempt to pick it up if you don't already have it, which process may fail if you're carrying too much. Although containers can contain other containers, the program doesn't access more than one level down.

Fighting:

Occupants of the dungeon will, as a rule, fight back when attacked. In some cases, they may attack even if unprovoked. Useful verbs here are 'ATTACK <villain> WITH <weapon>', 'KILL', etc. Knife-throwing may or may not be useful. You have a fighting strength which varies with time. Being in a fight, getting killed, and being injured all lower this strength. Your carrying capacity may also be reduced after a fight. Strength is regained with time. Thus, it is not a good idea to fight someone immediately after being killed. Other details should become apparent after a few melees or deaths.

Command parser:

A command is one line of text terminated by a carriage return. For reasons of simplicity, all words are distinguished by their first six letters. All others are ignored. For example, typing 'DISASSEMBLE THE ENCYCLOPEDIA' is not only meaningless, it also creates excess effort for your fingers. Note that this truncation may produce ambiguities in the interpretation of longer words.

You are dealing with a fairly stupid parser, which understands the following types of things--

Actions:

Among the more obvious of these, such as TAKE, PUT, DROP, etc. Fairly general forms of these may be used, such as PICK UP, PUT DOWN, etc.

Directions:

NORTH, SOUTH, UP, DOWN, etc. and their various abbreviations. Other more obscure directions (LAND, CROSS) are appropriate in only certain situations.

Objects:
Most objects have names and can be referenced by them.

Adjectives:
Some adjectives are understood and required when there are two objects which can be referenced with the same 'name' (e.g., DOORs, BUTTONs).

Prepositions:
It may be necessary in some cases to include prepositions, but the parser attempts to handle cases which aren't ambiguous without. Thus 'GIVE CAR TO DEMON' will work, as will 'GIVE DEMON CAR'. 'GIVE CAR DEMON' probably won't do anything interesting. When a preposition is used, it should be appropriate; 'GIVE CAR WITH DEMON' won't parse.

Sentences:
The parser understands a reasonable number of syntactic constructions. In particular, multiple commands (separated by commas) can be placed on the same line.

Ambiguity:
The parser tries to be clever about what do do in the case of actions which require objects that are not explicitly specified. If there is only one possible object, the parser will assume that is should be used. Otherwise, the parser will ask. Most questions asked by the parser can be answered.