Time Travel fixed*

* The trouble of up and down movement in the underground castle dungeon (Round 2) was corrected on December 10th, 2001,

VIC 1001 + 3K RAM
BASIC "TIME TRAVEL 1"
BASIC "TIME TRAVEL 2"

starting method:
add 3K RAM
LOAD
RUN

At last, the time machine was completed. My ten years effort has come to an end. With a robot called Max, I decided to go to Ancient Egypt. We arrived safely there, but while exploring, I was caught by a dragon. Apparently, Dragons also kidnapped two princesses. They are now guarded by a group of monsters.
I ask you to save the princesses. I cannot help you, but I will let you control my Max robot. Thank you, my friend.

About the game

This game is for I/O, August 1982. I borrowed the idea of "TINY DUNGEON".

Please, rescue the princesses who are trapped somewhere in the maze, using a remote-controlled robot for your exploration and fights.

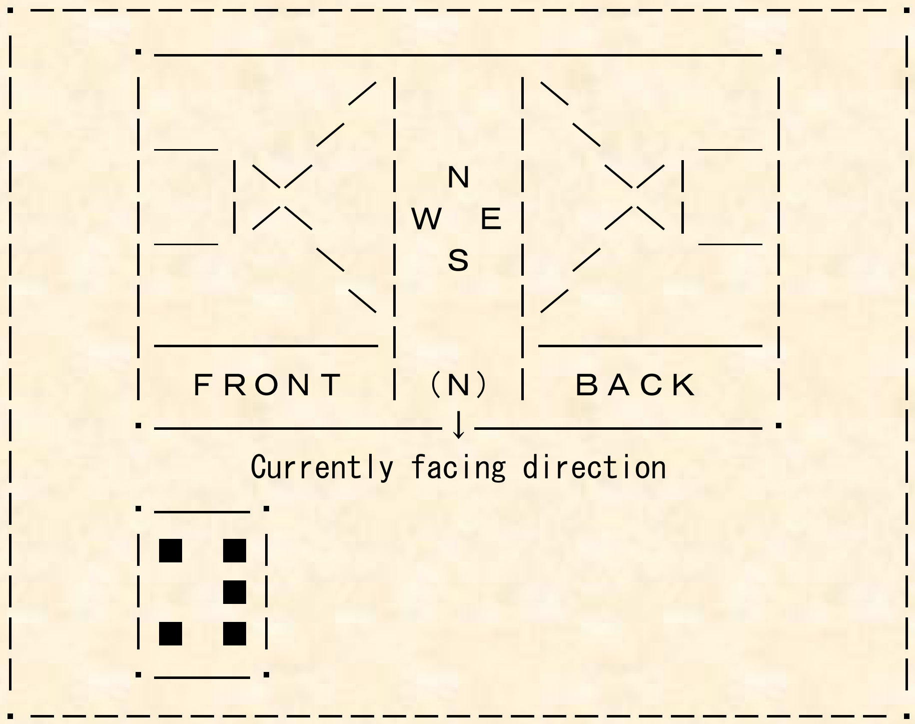
The first princess is in the pyramid. The pyramid has a 1st, 2nd and 3rd floor. They are 7×7, 5×5, 3×3 labyrinths, respectively, You can go Up and down only where there is a ladder.

When you bravely save the first princess, you move to the underground castle. This castle consists of three 7×7 mazes, ranging from the first to the third underground level, The game is getting more and more complicated…

The meaning of what appears on the screen is shown in Figure 1. The maze is displayed on the screen. The view represents your current position. You cannot tell how the maze will be going forward in your exploration.

The commands you can give to the robot are described under the screen display scheme.

1. Example of screen display



[G] – GO Go forward the given direction.
If there is a wall in this direction of travel you cannot move forward. Energy decreases by 1 to 3, depending on the current floor.

[L] -LEFT Rotates 90 degrees to the left.
Energy does not change.

[R] –RIGHT Rotates 90 degrees to the right.
Energy does not change.

[U] UP You can go up to the upper floor. It cannot be done in places without a ladder. Energy is reduced by 2.

[D] DOWN You can go downstairs. It cannot be done in places without a ladder. Energy is reduced by 2.

[F] –FIGHT when a monster or a dragon comes out. You use it when you want to attack.

[E] –ESCAPE when a monster or dragon comes out. You will use it to escape.

Monster Power

It shows the power of the current monster.

Fighting Or Escaping

The robot will show whether you are fighting or escaping a monster with [F] or [E], respectively.

Fighting Power

Shows the amount of energy used in the battle. If you have more power than the monster, then the probability of winning will be higher - but you will not always win! The highest energy you can have is shown as PROGRAM LEVEL .

Win

If you won the fight against a monster, the PROGRAM LEVEL is incremented by 1.

Lose

If you lost the fight with a monster, although the PROGRAM LEVEL does not change, the energy decreases by the power of the monster.

Trap

When you lose the fight with a monster, you will probably fall in a trap. You will drop down to the first floor directly from where you are now. However, it will not happen if you are on the bottom floor.

Magic Ring

Each princess was caught by a dragon. To defeat the dragon, 5 or more magic rings are necessary. You are awarded a magic ring when you defeat a monster.

I forgot to mention a couple of things, If you lose all your energy before you save the princess, it will be GAME OVER.

As the maze gets more and more complicated, I suggest to advance while writing down a map on paper.

About the program

One of the reasons why this program was put in about 5 KB is that I crammed a lot of information into 15 bits. Figure 2 shows how they are assigned.

Also, I tried as much as possible to use recurring subroutines, in order to save memory. Making unnecessary subroutines would have inflated the program considerably. Therefore, it may be difficult to decipher the program listing, Anyway, the main variables are shown in Table 1 and the program contents are shown in Table 2.

Figure 2. Description of each bit

```
.-------------------------------------------.
|  ○  ○  ○  ○  ○  ○  ○  ○  ○  ○  ○  ○  ○  ○  ○  |
|  ___________________                          |
|  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 |
.-------------------------------------------.
```

1-7.	Monster power
8.	Is there a magic ring?
9.	Does a monster exist?
10.	Upward ladder
11.	Downward ladder
12.	North
13.	East
14.	South
15.	West

Table 1. Variables

X	Horizontal
Y	portrait
Z	floor number
M	Shape of the current point
MU	Orientation
A (X, Y)	Maze on the first floor
B (X, Y)	Maze on the second floor
C (X, Y)	Maze on the third floor
RU	Round
EN	Energy
PL	Program level
ML	Number of magic rings [Magic Lings ☺]

Table 2. Description of the program

Line number	Description
1 - 140	Initial explanation
200 - 550	Main routine
600 - 710	Process when winning
750 - 800	Process when losing
2000 - 2030	Character display routine
2500 -	Clear both sides
3000 - 3520	Maze display routine
3600 - 3650	Direction display routine
4000 -	Demo display

Program input

First, enter List 1 and save it on tape. Next, enter List 2 and save it after List 1. When executing, read List 1 from the tape and run it, The game will start automatically by loading List 2.

A final word

Although this game is a bit different from "real time" games, I think that it can be considered as a proper game. Actually, I was planning to add music and animations, but eventually the game took this shape because of memory limitations.

There are not only "real time" games for the VIC-1001...
How about trying to create programs such as adventures and simulations?