Computer Programming

Perl Programming Assignment #2, 2007-8

Brief:

Your target is to create a simple text-based, arcade-style game. The object of the game is to guide your character from his initial starting point to the exit of a maze. There are no 'bad guys' required, though some objects in the maze will kill.

Informative:

This a task based on a classic game style. Though the implementation sounds rather difficult, this is actually a very manageable program.

There are no fancy graphics, as it emulates a type of game written when fancy graphics were not possible. There is little complicated program structure, as the gaming paradigm is relatively simple.

Process:

Your program will read a series of strings from a file on disk. The strings are stored in a one per line as a one dimensional array. However, since a string is a specialised array of characters, this effectively produces a two dimensional array, or a map grid to navigate.

This series of strings will be displayed on the screen. The contents of the strings are just ASCII characters, but different characters represent different objects. The list of objects is given here:

Symbol	Name	Represents
#	cardinal number, pound or hash symbol	Wall
@	at symbol	your character
P	letter P	Poison
T	letter T	Tonic
[]	space between two square brackets	The exit

The sample maze looks like this:

```
###[ ]###############################
##        #     P          #   # # #T#
##P#P#P################### ### # # # #
## # #     #     ###        P     # # # #
## # # ## #     ######### ### ### # # # #
## # # ## ########   P# ###  #        #
#      #### #    ## #### #### ###### # #
# ######T   # ## ## ####      ###### ###
# ## ## ### # ## ## #### ########## ###
#    ## ##    ## ## ##               ###
# ##     # ## ##   ## ######### ## ###
# ###### # ## ######## ####   #  ## ###
#    ###    # ## ##    ##    ######    #
# ###### #### ## ##### ## ####    ### #
#### ## # # #        ## #    ####### #
# ## ## # ## # ## ###### # ####    # #
# #    ## ## # ## ###### # ###### # # #
# ## ## ## #     ###         ## # # #
#     # #####  ######### #####   #@# #
#####################################
```

Once the maze is displayed, the program should enter a loop in which program awaits keyboard input and then acts upon it.

The list of expected inputs and actions is given here:

Key	Name	Expected action
←	Left arrow	If possible, character moves left one cell
→	Right arrow	If possible, character moves right one cell
↑	Up arrow	If possible, character moves up one cell
↓	Down arrow	If possible, character moves down one cell
Q	Upper or lower case letter q	Quit game

The character should be moved around until it either reaches the exit or comes into contact with a poison cell.

If the character comes into contact with a poison cell the character will die if there is not an unused tonic in its possession. The tonic so used is then considered canceled out and no longer available. The character obtains tonic by coming into contact with a tonic cell.

The character comes into contact with poison and tonic cells by moving onto them. Merely being on the adjacent cell does not count as coming into contact with those cells. Once a tonic cell is moved onto it is taken by the character and removed from the maze.

When the character is in the space between the square brackets at the exit of the maze the game is considered to be over.

Navigation of the maze is based on the current position of the character, and a request to move that is received. If the cell in the direction in which the character seeks to move is empty, the character is permitted to move. If not, the character is not permitted to move. Sounds simple, huh?

Presentation:
Marks are awarded for attractive presentation both of the screen output and the source code. Code indentation is *vital,* as is commenting of the code. An authorship section should be included at the top of the program.

Submission Mechanism:
By paper via the submission box in Room 15, and by e-mail (**the perl script only**). Don't forget the 'My Own Work' form as a cover page. Ensure you have at least:
- Cover sheet
- Flow chart (Read the programming website for information).
- Source code (The perl program, printed)
- Screen capture(s) as required

Any other relevant supporting materials

Due Date:
20080407, 15:15, Monday
(subject to on-line Calendar changes)

NOTE: The perl Term::Screen library is installed on the server and is required for this project.