Computer Programming

C Programming Assignment #4 1999/2000

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 is no fancy graphics, as it emulates a type of game written when fancy graphics were not possible. There is little complicated program structure, as earlier systems were less powerful than modern ones.

Your Task:
Write a program that reads in a series of strings from a file on the keyboard. The strings are stored in a one-dimensional array. However, since a string can be thought of as a specialised array for characters, this effectively produces a two dimensional array.

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

A sample maze might look 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 cancelled 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 of the exit 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?

Useful links:
However, for anyone able to read English well, and to reason their way through computer documentation independently, you can gain a head start by read the manual by typing:
`man ncurses`
at the linux prompt (which includes cross references to many other man pages), or by reading up on the new curses library on the internet.
There is a document by Eric S. Raymond which might be too good, as he writes C like a native(!). The ncurses package is based upon the curses package, which has a page here but has some dodgy links.

Documentation:
Provide evidence of your planning process, the commented source code and sample data used. You should also include a hand-drawn as well as a SmartDraw formatted flow-chart. Any other appropriate documentation may be included if you wish. The submission should be preceeded with a signed copy of the 'my own work' form.

Due Date:

Thursday, March 29th 2000, 17:00