Environment : ANSI C, CentOS 7.4.1708, GCC 4.8.5 20150623 (Red Hat 4.8.5-16)

The Main Goal and Points Need To Be Understood

There are many purposes for giving this assignment . More than one topics are included. I use structures , dynamic and static arrays, file input and output functions and processes , parsing and manipulating strings (char arrays), pointers and more. They are all combined with each other. The assignment makes me use the dynamic memory allocation and pointers that make C special and fast . In this assignment this two important topics and structures need to be understood .Understanding dynamic memory allocation ,file input / output process and pointers with structures is the main goal.

The Problem and My Solution to Approach

Loading Characters: There will be created a basic text based adventure game. Two input files will be given and first one contains the characters of games. In this file the line format will be <TYPE>,<NAME>,<HP>,<DAMAGE> . There can be any number of characters in file and the first step is loading this characters.

My Approach to loading characters: Firstly I create two structures. One for monster and the other for hero.The reason I create two structs is “monster” struct does not have “XP” .The structs also have “x” and “y” coordinates to settle characters in the map. I read file line by line and then I parse it. I created a blank object for appropriate

type of the line (hero/monster) and I store them in structs array dynamicly. (hero and monster struct array separately).

Playing the Game:

The Second input file contains commands to play the game. There are 5 commands named “LOADMAP” , “PUT” ,”SHOW”,”ATTACK” and ”MOVE”. The commands are stored dynamicly as first file.I loaded the map with “LOADMAP” command. The map a created is 2 dimentional dynamic array. And I created a function to printing the map in case the need. Secondly the “PUT” command is implemented and the characters settled the map. To implement “SHOW HERO” command , used a for loop and data processed from hero structures array . This is the same as “SHOW MONSTER” . To implement “SHOW MAP” command I use the before created function to show the map .

To implement “ATTACK <type>” command it is written 2 functions. One for heros and another is for monsters. When this command implement , This functions also check if hero or monster dead or alive and if dead delete from the map and write “,” instead the name’s first letter. There are two more functions “getTargety” and “getTargetx” to control adjancent enemies and get target them.

“HP “ ‘s will be decreased as “Damage” value of characters also. To get monster and hero coordinates. , I created four functions also. And so I could get index from name and from coordinates.

To implement “MOVE” command , I parse the command string and get locations separately. Then get index of hero and check negative situations . (“Out of map” , “place is occupied” ,”hero is dead”) and then print the appropriate messages.

Finishing Game : My program check after “ATTACK” command if all monster dead or alive. If one of this cases is true , then print “ALL <type>S DEAD” .

Used Data Structures and Functions

Functions used in assignment:

1- **strSubstring Function:** This function returns a substring of the given string between start and stop indexes.
   char* strSubstring(int startIndex, int stopIndex, char *str);
2- **compareString Function :** This function compares two string, if they are equal, it returns 1, otherwise 0

3 – getHeroIndexByName Function : This function returns the hero's index which has the given name.It gets the name, heroes array, heroes count,loops over the array, and compares the heroes with strcmp.

4 – getHeroIndexByCoordinate Function: Same idea with getByName. This gets by coordinate (x, y).

5-getMonsterIndexByName Function: Nothing different with hero functions. Its only for monsters.

6) getMonsterIndexByCoordinate Function : Do same job with getHeroIndexByCoordinate function this is just for monsters.

7) printMap Function : This function prints the 2D World. It prints it to the output file.

8) heroAttack Function: This function processes attack.The hero h attacks the monster m.m's health decreases by amount of h's damage power.It also checks if m is dead or not. If the monster m is dead, it increases h's XP point by one.

9) monsterAttack Function: Almost nothing different with heroAttack function. Just for attack to heros.

Used another data structures:

- Static arrays to store file's lines in.
- "monster" and "hero " structures to create and fill the features of characters.
- Monster and Hero Structures arrays dynamicly created to store structures.
- Malloc and free functions used frequently to create dynamic arrays.

Algorithm of the program

- First open and read files line by line .
- Count the hero and monster number.
- Create structure arrays on efor hero and the other for monster
- Close first file that include characters.
- Read the argv[1] file again (contains characters)
- Check first letter of line and if it is hero , create a blank hero struct. Use strSubstring (written from myself) function to parse string and use it. Count commas and adjust program to work .
- Fill the blank new blank hero's other informations like xp,hp,damage,x,y. (All them set to 0 first).
- If it is a monster do same things like hero.
- Create map dynamicly with "char **map".
- Read another line and parse it with strSubstring again .

- Read the file by “sscanf” and use “malloc” to create it dynamicly.
- If line starts with ‘P’ then parse it . Pay attention to commas and if neither “firstSpaceIndex” not “secondSpaceIndex” not zero , it means we are at third one and it is time to parse our “NAME X Y” .We are right after the “Y” part.
- Parse and change the characters’s x and y positions.
- Use strSubstring again.
- If there is a hero in command , then find index of hero by “getHeroIndexByName” function and assign to our hero .
- If there is monster do that again for our monster. Use “getMonsterIndexByName” and assign “x” and “y” values to our monster via access it.
- Put the first character of the name to the map .
- Reset flags to read new characters and positions.
- After every reset , free the allocated memory to prevent memory leaks.
- If command’s first character is ‘S’ than check fifth character for “SHOW HERO” command , check sixth character for “SHOW MAP” command , else implement “SHOW MONSTER” command.
- If there is “SHOW HERO” command , loop our heros array and print the information of them.
- If there is “SHOW MAP” command , call “printMap” function to print the map.
- If it is “SHOW MONSTER “ command , loop monsters array and print information about them.
- Else if first character of line is ‘A’ than it is a “ATTACK “ command
- If seventh character of line is ‘H’ than it is “ATTACK HERO” command.
- Make a loop itera the number of hero’s times and check whether hero’s health more than zero . If it is ; for each direction of 1 to 8 , we will get the index of monster at that index if there is .
- Get enemy coordiates “x” and “y” by “getTargetX” and “getTargety” functions.
- Than give this coordinates as parameters of “getMonsterIndexByCoordinate” function.
- Check if it is adjacent and if it is use “heroAttack” to attack the monsters.
- Count the number of dead monster and heros by the way.
- If it is “ATTACK MONSTER” command. Than do same things as “HERO ATTACK” .
- Check after every attack whether if all monsters or heros dead or alive.
- If it is true for one situation , finish the game and print message.
- If not , continue untill commands ended.