Program Design (I)

Personal Final Project

Demo Period: 2022/1/10 to 2022/1/12

Please use C language to complete the project.

Introduction

The final project is based on program exercises 4, 5, 7, and 8. You will need to complete your adventure and add some new functions. Moreover, the map size is up to 1000*1000. Also, you will need to write a README file to introduce the game you developed (please see the example of a README file below for the details). TAs will grade your final project according to two parts during the demonstration: (1) the basic part and (2) the extra part.

Basic Part

Apart from completing the program exercise 4, 5, 7, and 8, you need to add the following new functions for this adventure game:

- Monster dodge probability
- Get guns on the map (Attack+)
- Map trap (Blood-) (please see the example code below for more detail).

Please see other rules of grading in the Grading part below.

Extra Part

You can make any other extra functions to make this adventure game even more enjoyable! For example, you can add different kinds of monsters or even add another NPC in the game. The goal is to make a fun game to play. Please see the rules of grading in the Grading part below.

Submission and Demonstration

You will need to introduce and demo your game to TA within 15 minutes. We encourage you to make a slide to present your final game. Please reserve a time slot for the demo previously using the google sheet link below. **If you don't demo, you won't get any score for your final project.** All final project files need to be uploaded to eCourse2 before the demonstration. The rules of file naming will be updated at the announcement soon. Please remember to upload your code, README, and slide (if you have).

Demo time

- 19:00 - 21:00 during 1/10 (Mon.), 1/11 (Tue.), or 1/12 (Wed.)
- One student will have 15 mins for a demo
- Please fill in your student ID and name on this sheet (**only one student is allowed in one time slot**).
  https://docs.google.com/spreadsheets/d/1L_FrWVFA97OLsZbasPv3_QWEq-l7aQs6rV5Js6iL0zw/edit?usp=sharing
- Do not delete or modify other students' data. If you find your data has been modified or deleted, please contact TA.

Grading

- (80%) Basic Parts
  a. (20%) Program Exercise 4, 5, 7, 8, and new functions
     - The map size is up to 1000*1000
     - Monster dodge probability
     - Get guns on the map (Attack+)
     - Map trap (Blood-)

  b. (15%) Pointers
     - You can not use external/global variables

  c. (15%) Pointers and Array
     - Use `malloc` for the map

  d. (15%) README
     - See README example

  e. (10%) Functions
     - All in Exercise 5 change to use function
     - Clear `main` function (You need to distribute most of the statements to different functions and only keep the function calls in the `main` function)
     - Don’t place the `main` function in strange places

  f. (5%) Code Review During Demonstration

- (20%) Extra Parts
  a. (10%) Extra functions to make the game more interesting
  b. (5%) Clean code architecture
  c. (5%) Detailed comments

Example for README (Please saved in pdf format)

- Project Description
  - < Describe the mechanism of this game and the new functions you make >
- Playing Method
  - < How to play it? How does the game start? Using what command to play? How does the game end? >
- Function Description
  - < Describe the purpose and usage of every function in your game >
- Variable Description
  - < Describe purpose and usage of every variable in your game >
- Version History
  - <0.1 - Initial Release>

Example Code for Basic Part

Functions

Monster dodge probability

```
//monsterLevel: 怪物等級 1 ~ 4
//playerHealth: 玩家血量，使用 pointer 傳遞數值 1 ~ 10
//playerAttack: 玩家攻擊力，至少為 1
//回傳戰鬥結果
// === English Version ===
//monsterLevel: monster level ranging from 1 to 4
//playerHealth: the amount of player's blood, use pointer to pass a
integer value ranging from 1 to 10
//playerAttack: the value of player's attack with minimal value of 1
// Return a integer value to represent the result of battle

int battle_result(int monsterLevel, int *playerHealth, int playAttack){
    //設定亂數種子碼，給等等的 evasion 函式使用
    //setup the seed for generate random number for evasion function

    //如果一直怪物閃避成功且血量大於 0，繼續判斷怪物是否閃避成功
    //if monster dodge the attack with blood value greater than zero,
    //keep judge whether the monster dodge the attack successfully

    //攻擊力低於怪物等級則輸，血量被扣到 0 代表怪物閃避後的攻擊把玩家打死
    //player with attack value lower than the monster
    //will lose the battle. The player's blood will reach zero when the
    //monster skip successfully and defeat the player
}
//evasionRate: 閃避機率 0 ~ 100
//monsterLevel: 怪物等級 1 ~ 4
//playerHealth: 玩家血量，使用 pointer 傳遞數值 1 ~ 10
//回傳閃避是否成功
// === English Version ===
//evasionRate: probability of skipping ranging from 0 - 100
//monsterLevel: monster level 1 - 4
//playerHealth: the amount of player's blood, use pointer to pass a
integer value ranging from 1 to 10
//return whether skip successfully
int evasion(int evasionRate, int monsterLevel, int *playerHealth){
    //檢查是否閃避成功
    //check if the monster dodge successfully

    //閃避成功則回擊
    //if so, the monster will attack back
}
```

Get guns on the map (Attack+)

```
//xAxis x 座標      yAxis y 座標
//mapRow 地圖的 row       mapColumn 地圖的 column
//回傳是否在地圖邊界內
// === English Version ===
//xAxis: x coordinate      yAxis: y coordinate
//mapRow: row of the map  mapColumn: column of the map
//return if still inside the map
int check_boundary(int xAxis, int yAxis, int mapRow, int mapColumn){}

//map 地圖
//xAxis x 座標     yAxis y 座標
//回傳地圖位置是否已被空白
// === English Version ===
//map
//xAxis: x coordinate      yAxis: y coordinate
//return if the location is available
int check_availability(char **map, int xAxis, int yAxis){}

//map 地圖
//mapRow 地圖的 row       mapColumn 地圖的 column
//xAxis x 座標     yAxis y 座標
//gunNumber 槍的數量
// === English Version ===
//map
//mapRow: row of the map  mapColumn: column of the map
//xAxis: x coordinate      yAxis: y coordinate
//gunNumber: number of gun in the map

void setup_gun(char **map, int mapRow, int mapColumn){

    //輸入槍的位置
    //enter the location of the gun
    for (){
        //檢查位置是否能用
       //check if the location is available
    }
}

//playerAttack 玩家攻擊力
void encounter_gun(int *playerAttack){}
```

Map trap (Blood-)

```
//map 地圖
//mapRow 地圖的 row       mapColumn 地圖的 column
//xAxis x 座標     yAxis y 座標
//trapNumber 陷阱的數量
// === English Version ===
//map
//mapRow: row of the map  mapColumn: column of the map
//xAxis: x coordinate      yAxis: y coordinate
//trapNumber: number of trap in the map
void setup_trap(char **map, int mapRow, int mapColumn){

  //輸入陷阱位置
    //enter the location of the trap
    for (){
       //檢查位置是否能用
      //check if the location is available
    }
}

//playerHealth 玩家血量 (the value of player's blood)
void encounter_trap(int *playerHealth){}
```