CS1580 - Introduction to Programming Lab (FS2024)
Lab 7

Lab Objectives

In this lab, you will be implementing the following topics:

- Structs
- Multiple files
- Random Number Generation
- Function Documentation

Lab Task: Simple Adventure Game Simulation

In this assignment, you will build a program that simulates a simple adventure game. The player will traverse a series of random events and encounter outcomes like gaining health, losing health, or finding treasure. The simulation will involve a basic **game loop** that runs until the player's health reaches zero.

Implement the assignment in multiple files:

- game.h
- game.cpp
- main.cpp

Game Mechanics:

1. In your `game.h`, define a struct `Player` with the following member variables:
    a. Health (int)
    b. Score (int)

    Also, define the prototypes of all the functions.

    ```
    #ifndef GAME_H
    #define GAME_H

    #include <iostream>
    using namespace std;

    // Your code here.

    #endif
    ```

    Do not forget to include the header file in all other cpp files.
    `#include “game.h”`

    You don’t have to include the packages again in cpp files.

2. In the implementation file, `game.cpp`, write the game logic including random event generation and handling outcomes based on random values.

    a. Define the game simulation function as following:

```
void simulateTurn(Player& p) {
      // Write your logic here.
}
```

    In this function,

        i. Generate a random number `int event` in the range 0 – 2. Use seed value as `srand(time(0))` and do not forget to include the following packages

```
#include<cstlib>
#include<ctime>
```

        ii. If `event == 0`, Player finds a **health potion** and gains between 5 and 20 health points, i.e., generate a random number between 5 and 20.
        If `event == 1`, Player was **attacked** and loses between 10 and 30 health points.
        If `event == 2`, Player finds a **treasure** and gains 50 score points.

    b. Write a function to display Player's health and score at each game run. This function **overloads the cout operator.**

```
void displayStatus(const Player& p) {
      // Print status here.
}
```

3. In your `main.cpp`,

    a. The Player struct will store the player's health and score. The game will start with a player having an **initial health of 50 and a score of 0**.
    b. Main game loop:
        i. Simulate for one turn by calling `simulateTurn()` function
        ii. Print the current Player status by calling `displayStatus()` function
        iii. Pause between turns.

```
cout << "Press Enter to continue."
cin.ingore()
```

        iv. **Continue this process until Player health <= 0**
    c. Finally, print the Player's score

Function Documentation (in your game.cpp)

```
//Description: a short description about the function
//Pre: what are parameters and their data types
//Post: what is the function returning and its type
void your_function(){

}
```

Sample Input/Output

```
You found a health potion! Gained 15 health points.
Player Status: Health = 65, Score = 0
Press Enter to continue...

You were attacked! Lost 20 health points.
Player Status: Health = 55, Score = 0
Press Enter to continue...

You found treasure! Gained 50 score points.
Player Status: Health = 55, Score = 50
Press Enter to continue...

...
Game Over! Final Score: 200
```

Gitlab Cloning Instructions

- Open the browser and go to https://git-classes.mst.edu/. Click on the Lab7 repository named *2024-FS-303-lab7-<your_username>*
- Click on 'Clone' button and copy the HTTPS link.
- Open Putty and
    - Change the directory to SDRIVE: `cd SDRIVE`
    - Clone the repository: `git clone <copy_the_HTTPS_link_here>`
    - Change the directory to cloned repository: `cd 2024-FS-303-lab7<your_username>`
- Start coding by opening a new file in nano: `nano main.cpp` and `nano game.cpp`

Compiling Instructions

- To run your code, `fg++ *.cpp`
- To get the output, `./a.out`

Submission Instructions

Push your code to your gitlab account.

- Add all your files to the repository, `git add .`
- Commit your changes, `git commit -m "<your_message_goes_here>"`
- Push the changes, `git push`