Lab 5 [Overworld]

In this workshop, we will continue our development of a text based game "Code Quest!". This workshop focuses on the use of pointers to change variables within functions.

LEARNING OUTCOMES

Upon successful completion of this workshop, you will be able to

- use pointers to assign to variables from functions

Part 1:

SPECIFICATIONS

Write a program which will act as an overworld for CodeQuest! It will track the number of days remaining (before the destruction of the world) and the player's HP.

This program will require the following 3 functions:

`void rest_at_inn(float* days_remaining, int* hp_remaining, int max):` This function takes three arguments a float address (days remaining), an int address (HP remaining), and an int (max HP). This function refills the player's HP to its maximum value and decreases the number of days remaining by one. Print a message indicating what has been done (check output below).

`void train(float* days_remaining, int* hp_remaining, int* experience):` This function takes three arguments a float address (days remaining), an int address (HP remaining), and, an int address (Player Experience). This function increases the player's experience by 10 but reduces HP by 2 and reduces the number of days remaining by 0.5. This function prints a message indicating what has been done (check output below).

`void battle_demon_lord(int* current_hp):` This function takes one arguments an int address (current HP). This function simply outputs "He's too strong!" and sets the player's current HP to zero. This function will be more correctly filled in later in the semester, it will act as the final battle in the game.

Once the above three functions are implemented, write a program that displays the player's stats and days remaining followed by a menu to the user and prompts them for input. This program will track the number of days remaining (float, initialize to 8), the player's current HP (int,

initialize to 10), the player's maximum HP (int, initialize to 10), and the player's experience (int, initialize to 0). The menu will allow the user to select either "1 - Rest at Inn", "2 - Train", "3 - Fight the Demon Lord", "4 - Quit Game". Based on the user's input, call the associated function passing in the required arguments. This program should not exit until either 4 is selected, or if days or the user's HP is less than or equal to zero. If either are zero or below output "game over" before shutting down.

The output of a typical run-through of your program should look like this (user's input highlighted in green). This does not cover all possibilities, ensure you're program follows all paths correctly:

```
Days remaining: 8.0 HP: 10 EXP: 0

 1 - Rest at Inn
 2 - Train
 3 - Fight the Demon Lord
 4 - Quit Game

 Select: 2
 You did some training!

 Days remaining: 7.5 HP: 8 EXP: 10

 1 - Rest at Inn
 2 - Train
 3 - Fight the Demon Lord
 4 - Quit Game

 Select: 2
 You did some training!

 Days remaining: 7.0 HP: 6 EXP: 20
 1 - Rest at Inn
 2 - Train
 3 - Fight the Demon Lord
 4 - Quit Game

 Select: 1
 You rested up at the inn

 Days remaining: 6.0 HP: 10 EXP: 20

 1 - Rest at Inn
 2 - Train
 3 - Fight the Demon Lord
 4 - Quit Game

 Select: 3
 He's too strong!

 Game Over!
```

Be sure to test the quit condition and the day limit condition to ensure they work as well. If your program's output exactly matches the output shown above, given the provided inputs, and meets the day and quit conditions then your lab is complete and ready to be submitted (read below).

PART1

BUILDING THE GAME (bonus)

If you have completed all previous game labs, you may complete this section in order to merge your current lab with you previous labs.

Copy the functions (and prototypes) into your codeQuest source code file. Make a new function with the following signature:

```
void battleSequence(float* days, int* curHP, int* exp, int maxHP, int str, int def, int intel, int luck)
```

move all source code for the battle sequence to this function, adjust it as needed to make use of the parameter values. Enemy stats are not passed in, they should remain hard coded for now. For each round of the battle, reduce the days' value by 0.1. If the player wins, increase their experience by the original health of the enemy (10).

Move the main code from this lab into the newGame function where the battle code used to be, just after character creation. A new variable will need to be added to differentiate currentHP and maxHP. Add another option called "Fight Monster" which when selected calls the battle scene function, passing in the required variables. Use the input verification function from the previous lab to replace the menu input in this lab. Clear the screen before the overworld menu displays each time, add pauses (wait for the user to press enter) where appropriate so the messages don't dissappear immediately.

If you've followed all instructions up to this point, you should be half done CodeQuest! Check attached to download a binary version of CodeQuest v0.5 to help you ensure your game works correctly!

CodeQuestV0.5.exe for Windows

CodeQuestV0.5 for Linux

Prepare a Typescript

Create a typescript on the remote Linux host using the following commands:

```
+ At the prompt, type: script w5.txt
+ At the prompt, type: whoami
+ At the prompt, type: cat w5.c
+ At the prompt, type: gcc w5.c –o w5.out
+ At the prompt, type: w5.out
+ At the prompt type: exit
```

This will produce a typescript named **"w5.txt"** in your current directory. Transfer a copy of this file to your local computer.

SUBMISSION

Upload your typescript file to BlackBoard:

- Login to BlackBoard
- Select your course code
- Select Workshop 5 under Workshops
- Upload **`w5.txt`**
- Write a short note to your instructor
    - Under “Add comments”, add a sentence or two regarding what you think you learned in this workshop in the notes textbox
    - press "Save Changes"
- When ready to submit, press "Submit". Note you can save a draft until you are ready to submit.