COMP 2710: Project 1 – Phase 2: Design

Dragons – A Simple Text-Based Game

Design (100 points) – turned in via Canvas

No collaboration between students. Students should NOT share any project code with each other. Collaborations in any form will be treated as a serious violation of the University's academic integrity code.

Rating

- Design difficulty: 4/5
- Implementation difficulty: 2/5
- Time required: 4/5
- Fun: 3/5

<u>Goals of Project 1:</u>

- To perform function-oriented analysis, design, and testing
- To develop some proficiency with basic C++ syntax and semantics.
- To learn data flow diagrams.
- To learn use cases and use case diagram.
- To gain experience with unit and system testing.
- To perform separate compilation.
- To use singly linked lists
- To develop a reasonably user-friendly application
- To write a fun application!

<u>Goals of Project 1 - Phase 2:</u>

- To design a singly linked list.
- To learn the function-oriented design approach.
- To design function prototypes.

- To learn how to design algorithms using pseudo-code.
- To design unit and system testing cases.

1. Overview

From your analysis performed in Step 1, you can design a class diagram, a system sequence diagram, and system-level test cases in this phase. Please do **NOT** intend to implement the entire program in a single main function.

1.1 Data Structures (15 points)

You must design important data structures. **Note:** You must use **singly linked list** as one of your data structures. You will lose points if you do not use singly linked lists**.**

Note: Suggested structures

You will use structures in this assignment. You will identify which data structures to use in your design. Obvious basic structures will be Menu, System, Character, Scoreboard, Scorefile, Encounter, and Puzzle. You are free to other data structures as you see fit.

1.2. Function Prototypes (15 points)

1.2.1 Requirements

You must design a list of function prototypes for your program to be implemented in the next phase (Phase 3 – Implementation) of this project. Please refer to

http://en.wikipedia.org/wiki/Function_prototype

for details on function prototypes.

Each function prototype must contains the following items:

- Function name
- A list of parameters, including parameter names and their data types
- Return data type
- Purpose of the function

1.2.2 Suggested Functions

A well-done implementation will produce a number of robust functions, many of which may be useful for future programs in this course and beyond.

Remember good design practices include:

- A function should do one thing, and do it well
- Functions should NOT be highly coupled

Some potential functions:

- A few functions to deal with the Menu issues, handling basic user screw-ups (choosing an option out of bounds).
- A few functions for the system, which instantiates the other functions and runs encounters. Receives input from the Menu functions.

- A few function for Encounters, providing the basic framework for a generic encounter (with additional information feed in from system as the encounters occur).
- A Puzzle function which operates with Encounter.
- A HighScores data type which loads, sorts, and collects the high scores.
- A few functions for Characters, keeping track of all the data relating to the character.

1.3 Algorithms (20 points):
You must design important algorithms for the above function prototypes. You may use pseudo code to present your algorithms. Please refer to
http://en.wikipedia.org/wiki/Pseudo_code
for details on pseudo-code.

1.4 Testing Cases (50 points):
1.4.1 (20 points) The System at Large
System-level tests are designed to fully exercise a program as a whole and check that all software components of your integrated system function properly.

For the system at large. In other words, describe inputs for "nominal" usage. You may need several scenarios. Also, suggest scenarios for abnormal usage and show what your program should do (for example, entering a negative number for a menu might ask the user to try again). Please follow the format of the following table below to prepare your test cases at the system level.

Pitfalls:
- No purpose.
- Only offer testing ideas. No expected input and output.
- Expected inputs are not specific.
- Expected outputs are not specific.

Test Cases:			
What	**Input**	**Expected Output**	**Actual Output**
Entering negative number for a menu.	-1	"The selected option does not exist. Please try again."	
Entering a letter for a menu.	G	"The selected option does not exist. Please try again."	
Leaving the menu option blank.		"Please select an option."	
Selecting a menu item that does not exist.	*	"The selected option does not exist. Please try again."	
If user neglects to enter a valid username & password.	Username: aaaaaa Password: 111111	"Error – No user exists. Please enter a valid username and password."	

1.4.2 (30 points) Unit Testing
Prepare a test driver for each function. (Later, these tests can be automated easily using a simple driver function in the function). Remember, test cases are specific, repeatable, and should have a clearly defined result. Testing functions with randomization may require specialized drivers.

2. Background

A customer wants a cool new video game, similar in style to popular video games like *World of Warcraft* or *Bioshock*. However, the customer recognizes the lack of significant funds and only has a resource poor computer to play this game on. Hence, the game will be a simple text-based adventure concerning a graduate student trying to navigate his way down Shelby Center. In this project, you will help the customer to design and implement a simple text-based game.

3. Requirement Details

3.1. Player
The “player” is represented by *at least* three attributes: *intelligence*, *time*, and *money*. If the player runs out of intelligence, time or money, the player dies. The goal of the player is to survive to the end of the “hall” with the highest combined total of the attributes as possible. “Score” is determined as the three attributes multiplied together.

3.2. The Hall
The player starts the game at the beginning of a hall, which is linear.

The “Hall” is a path that is at least twenty (20) steps long. After a move, the user should be told how far away from the goal they are (in steps). If the player survives to the goal square without any of the attributes falling to 0, they win. Their score should be displayed with a simple ASCII victory message. If the player dies, a “You Lose” message should appear indicating the cause of death (for example, if money falls to zero, you can say that the player starved to death because of poverty). All the attributes should start in some random range (e.g. 8-25).

3.3. Turns
Every turn, the player has (at least) 5 options to choose from:

- **Move**: The player moves one step in the grid, but risks an Encounter or a Puzzle. Moving also takes time.
- **Read technical papers** (boost intelligence): The player loses a fixed amount of time, but increases intelligence by a random amount
- **Search for loose change** (boost money): The player loses a fixed amount of time, but increases money by a random amount.

- **View character**: A simple display should show the character attributes and current position in the hall (ASCII is fine)
- **Quit the game** (shows the “You Lose” screen – optional mockery- and exits the program)

3.4. Encounters

Encounters: Every time the character steps, there is a random change of various events happening. You are free to change the probabilities as you see fit for “game balance,” but here are some suggestions:

- 25% chance: nothing happens, you just move forward.
- 30% chance: You encounter a Puzzle (see Puzzle below)
- 10% chance: Encounter a professor. This loses a random extra amount of time, but may slightly increase intelligence.
- 10% chance: Encounter another graduate student. This loses a random amount of time.
- 15% chance: Attacked by grunt work! Lose both time and intelligence.
- 10% chance: Grade papers. Lose time, but gain money.
- 0% chance: Get a huge raise, gain lots of money! (This never happens).

3.5. Puzzles

Puzzles: Puzzles are different from normal encounters since they require interaction from the user. These don’t necessarily have to be brilliant, but riddles or even edutainment light puzzles are fine. Examples:

- “What is 2 + 2:” For a correct response, Money + 1. For an incorrect response, Money -20 (you idiot).
- “What can you put in a barrel to make it lighter?” For a correct response, int+2. For an incorrect response, int-2.

3.6. Other options

You are free to add more details and rules to your game, but you must have at least the above specifications. Feel free to be creative – there are many opportunities to do so.

4. Deliverables

Please submit your project design through the Canvas system (e-mail submission will **<u>not</u>** be accepted). You need to submit your design document as (1) a C++ source code file named project1-design.cpp and (2) a doc or pdf file named project1-design.doc or project1-design.pdf.

- **project1-design.cpp:** this file contains data structures, function prototypes, and test drivers.
- **project1-design.pdf or project1-design.doc:** your pdf or doc file should contain the algorithms (i.e., pseudo-code) and the design of system-level

tests.

5. Grading Criteria

5.1 (15 points) Data Structures

- (5 points) structures
- (4 points) other data types
- (6 points) singly linked list

5.2 (15 points) Function Prototypes

- (3 points) Function name
- (3 points) A list of parameters, including parameter names and their data types
- (3 points) Return data type
- (6 points) Purpose of the function

5.3 (20 points) Algorithms

5.4 (20 points) System-Level Testing

- (15 points) nominal usages
- (5 points) abnormal usage

5.5 (30 points) Test Drivers

6. Late Submission Penalty

- A ten-point penalty per day for late submission. For example, an assignment submitted after the deadline but up to 1 day (24 hours) late can achieve a maximum of 90% of points allocated for the assignment. An assignment submitted after the deadline but up to 2 days (48 hours) late can achieve a maximum of 80% of points allocated for the assignment.
- Assignment submitted more than 3 days (72 hours) after the deadline will not be graded.

7. Rebuttal period

- You will be given a period of 72 hours to read and respond to the comments and grades of your homework or project assignment. The TA may use this opportunity to address any concern and question you have. The TA also may ask for additional information from you regarding your homework or project.

8. Hints

- Start early, you have a good deal of time but you may need it to start implementing your design. Although the following timeline is not mandated, it is

a suggestion of milestone:
  - 1/4 time: Design data structures and function prototypes.
  - 2/4 time: Design algorithms.
  - 3/4 time: Prepare system-level test cases.
  - 4/4 time: Write test drivers.
- If you bring your design documents by early, I will give you comments and help point you in the right direction on this project.

9. Sample Usage

```
        What’s your name? Bob


        ===========================================================
        |                    Welcome, Bob!                        |
        ===========================================================

   1) Start a New Game of Dunstan and Dragons!
   2) View top 10 High Scores
   3) Quit

        Please choose an option: 2

   The top 5 High Scores are:

   Win 1337
   CaseyZZZ 625
   JonnieKill 400
   Bob 75
   Daisy 33
   -no more scores to show-


   1) Start a New Game of Dunstan and Dragons!
   2) View top 10 High Scores
   3) Quit


        Please choose an option: 1

Entering the Dungeon…

You have:

intelligence: 20
time: 25
money: $11.00

You are 20 steps from the goal.  Time left: 25.
```

```
          1) Move forward(takes time, could be risky…)
          2) Read technical papers (boost intelligence, takes time)
          3) Search for loose change (boost money, takes time)
          4) View character
          5) Quit the game


      Please choose an action: 4



You have:

intelligence: 20
time: 25
money: $11.00

You are 20 steps from the goal.  Time left: 25.

          1) Move forward(takes time, could be risky…)
          2) Read technical papers (boost intelligence, takes time)
          3) Search for loose change (boost money, takes time)
          4) View character
          5) Quit the game


      Please choose an action: 2

You read through some technical papers.  You gain 3 intelligence, but
lose 2 units of time.

You are 20 steps from the goal.  Time left: 23.

          1) Move forward (takes time, could be risky…)
          2) Read technical papers (boost intelligence, takes time)
          3) Search for loose change (boost money, takes time)
          4) View character
          5) Quit the game


      Please choose an action: 1

You move forward one step, and…

NOTHING HAPPENS!

You spent one unit of time.

You are 19 steps from the goal.  Time left: 22.

          1) Move forward (takes time, could be risky…)
          2) Read technical papers (boost intelligence, takes time)
          3) Search for loose change (boost money, takes time)
          4) View character
          5) Quit the game


      Please choose an action: 1

You move forward one step, and…

YOU FIND SOME PAPERS TO GRADE.

You spent two units of time, but gained $3.00!

You are 18 steps from the goal.  Time left: 20.  You can move forward
or backward.

          1) Move forward(takes time, could be risky…)
          2) Read technical papers (boost intelligence, takes time)
          3) Search for loose change (boost money, takes time)
          4) View character
          5) Quit the game


      Please choose an action: 1

You move forward one step, and…

PUZZLE: It’s a riddling imp.  I hate riddling imps.  But fine, he asks:
“Find the product of 8 and 8!”

   1) 16
   2) 64
   3) 256
   4) Uh…uh… no?

Choose wisely: 4

The imp cackles “Oh yes.  Yes indeed.  Now you die.”

TIME HAS FALLEN TO ZERO.  YOU DIE.

<Print Score, adjust high scores>
```