WESTERN UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

SE2250b – Software Construction

Project: Role Playing Game

Objective

This project provides hands-on experience on various aspects of software construction including practical experience with software implementation and testing. It will give you an opportunity to improve your software development practices and to try the iterative model of the software development life cycle through three phases of the project.

Phase 3:

Deadlines:

Section 002: 8:00 am, Tuesday, March 31st, 2020
Section 003: 8:00 am, Monday, March 30th, 2020

This phase will continue the game you described and started in phases 1 and 2. From the previous phases, you should have working Level 1.

NOTE: Proper coding practices must be used. Examples include but are not limited to the following: naming conventions, project organization, proper code formatting, meaningful comments, clean code (no old commented out code or unnecessary code), and any other topics discussed in lectures or labs. The object-oriented design must be used throughout the project.

Requirements:

Implementation requirements for this phase:

- Character progression. The character needs to evolve/change throughout the game. Examples include the character getting more resilient with experience or the character getting a boost when not performing well.
- Progress measure. This could, for example, be points the player accumulates when completing tasks, combats, and similar. Points should reflect the difficulty of the task and there should be at least two different activities resulting in two different point gains or different progressions.
- At least one animation. What exactly you are going to animate is up to you, but you must use Unity Animator for this. For the animation, you need to create your own objects (cannot use downloaded assets but can use downloaded images). Make animations non-trivial: an

example of a trivial animation is the 3D animation from “Unity – Sprites and Animations” tutorial.

- At least one additional level. The new level must introduce a different feature and there must be a visible indication of the current level on the screen. The new features cannot be simple extensions of the previous layer such as an enemy moving faster. Examples of the new features include new types of enemies, new tasks or quests.
- Two more features of your choosing. Those features should not be very similar to other features implemented as a part of another requirement.

Watch for:

- One feature cannot be counted for more than one requirement. For example, if a specific feature is counted towards character progression, it cannot be counted as an additional level feature.
- At the end of this phase, you need to have a playable game. During the demo, you will need to show and explain your game. If needed, you can slow down movements, actions and similar (by changing parameters) so it gives you time to demonstrate features.

Select **three** software features you are implementing in **this phase**. The three must be significantly different: for example, two enemies, one moving straight and one zig-zag, involved in the same combat type are overly similar features. In contract, an exploding enemy and a shooting enemy are sufficiently different. For each of the three selected features, do the following tasks:

1. Identify a new software feature. Describe how your feature will behave (requirements).
2. Define the key components (objects) of the new software feature using the object-oriented approach.
3. Create UML diagrams such as class diagrams, flowcharts, and sequence diagrams to clearly define the strategy to solve the problem. The choice of the UML diagrams is up to you as long as your UML diagrams convey your design (including dynamic behaviour).
4. Create a test plan and test the new feature. Use the provided Test Plan template.

Marking:

Part	**Mark**
Implementation (including demo)	20
Question 1. Requirements	5
Question 2. Components	3
Question 3. Design (UMLs)	12
Question 4. Testing	10
Total:	50

WESTERN UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Deliverables for Stage 3:

- Functionality demonstration. All group members must be present for the demo.
- Only one group member should submit the project. Names of all group members should be included with the OWL submission.
- OWL submission:
  - A single Pdf file including tasks 1, 2, and 3 for each of the three selected features. This must be uploaded as a separate file and not a part of the game zip file.
  - The test plan spreadsheeted (task 4). This must be uploaded as a separate file and not a part of the game zip file.
  - All game files uploaded to OWL. This should contain all the files needed to run your game from UNITY.

- Code submitted to GIT.
- In OWL submission provide a reference to the GIT submission corresponding to Phase 3: Project, Phase 3: git submission xyz

For no demo, 50% of the available mark will be deducted. Each group member must be present during the demo. Not being present for the demo without justifiable reason (providing documentation to Undergraduate Office or self-reported absence) will result in a deduction of 50% of the available mark for that student.