College of Engineering, Construction & Living Sciences
Bachelor of Information Technology
ID630151: Introduction to Algorithmic Problem Solving
Level 6, Credits 15

Portfolio

Assessment Overview

In this **individual** assessment, you will develop **four** games using **C#** in **Unity**. You will build the core mechanics for each game in class using the provided **lecture notes**. You will be given **assessment tasks** to implement independently. These tasks are at a **beginner** to **intermediate level**. In addition, marks will be allocated for code elegance, documentation & **Git** usage.

Learning Outcomes

At the successful completion of this course, learners will be able to:

1. Design & build usable, attractive games using various introductory algorithms following an appropriate software development methodology.

Assessment Table

Assessment Activity	**Weighting**	**Learning Outcomes**	**Assessment Grading Scheme**	**Completion Requirements**
Portfolio	100%	1	CRA	Cumulative

Conditions of Assessment

You will complete this assessment during your learner managed time, however, there will be availability during the weekly meetings to discuss the requirements & your progress of this assessment. This assessment will need to be completed by **Wednesday, 22 June 2022** at **5 PM**.

Pass Criteria

This assessment is criterion-referenced (CRA) with a cumulative pass mark of **50%** over all assessments in **ID630151: Introduction to Algorithmic Problem Solving**.

Authenticity

All parts of your submitted assessment **must** be completely your work & any references **must** be cited appropriately including, externally-sourced graphic elements using **APA 7th edition**. Provide your references in a **README.md** file. All media **must** be royalty free (or legally purchased) for educational use. Failure to do this will result in a mark of **zero** for this assessment.

Policy on Submissions, Extensions, Resubmissions & Resits

The school's process concerning submissions, extensions, resubmissions & resits complies with **Otago Polytechnic** policies. Learners can view policies on the **Otago Polytechnic** website located at https://www.op.ac.nz/about-us/governance-and-management/policies.

Submission

You **must** submit all program files via **GitHub**. The latest program files in the **master** or **main** branch will be used to mark against the **Functionality** criterion. Please test your **master** or **main** branch application before you submit. Partial marks **will not** be given for incomplete functionality. Late submissions will incur a **10% penalty per day**, rolling over at **5:00 PM**.

Extensions

Familiarise yourself with the assessment due date. If you need an extension, contact the course lecturer before the due date. If you require more than a week's extension, a medical certificate or support letter from your manager may be needed.

Resubmissions

Learners may be requested to resubmit an assessment following a rework of part/s of the original assessment. Resubmissions are to be completed within a negotiable short time frame & usually **must** be completed within the timing of the course to which the assessment relates. Resubmissions will be available to learners who have made a genuine attempt at the first assessment opportunity & achieved a **D grade (40-49%)**. The maximum grade awarded for resubmission will be **C-**.

Resits

Resits & reassessments **are not** applicable in **ID630151: Introduction to Algorithmic Problem Solving**.

Instructions

You will need to submit games & documentation that meet the following requirements:

Functionality - Learning Outcomes 1 (70%)

- Application **must** open without code or file structure modification in **Unity**.
- The four games you will create are:
  - Introduction to Unity scripting - Sheep Saving (15%)
  - Game mechanics - Tower Defence (15%)
  - Maze generation - 3D Dungeon Crawler (20%)
  - AI strategy - Chess (20%)
- In the course materials repository on **GitHub**, you will find the following directories:
  - 01-introduction-to-unity-scripting
  - 02-game-mechanics
  - 03-maze-generation
  - 04-ai-strategy
- In each of these directories, you will find additional directories - **lecture notes**, **assessment tasks** & **advanced assessment tasks**.
  - The **lecture notes** consist of detailed step-by-step tasks that will help you develop skills & knowledge in **Unity** while building a simple game. In addition, you will be introduced to commonly used algorithms in games. **Note:** If you do not complete these tasks, you will be able to successfully complete the **assessment** & **advanced assessment tasks**.
  - The **assessment tasks** consist of step-by-step tasks that will help you extend the functionality of your game. However, these tasks are not as detailed as the **lecture notes**.
  - The **advanced assessment tasks** consist of **independent research** & **problem-solving** tasks that will help you extend the functionality of your game to an **intermediate level**. You will complete these tasks in your own learner managed time.

Code Elegance - Learning Outcomes 1 (20%)

- Use of intermediate variables, i.e., no function calls as arguments.
- Idiomatic use of control flow, data structures & in-built functions.
- Efficient algorithmic approach.
- Sufficient modularity.
- Adhere to an **Object-Oriented** architecture.
- File header comments. You **need** to explain the purpose of each **script** file.
- In-line comments. You **need** to explain complex logic in each **script** file that is not obvious.
- **Script** files are formatted.
- No dead or unused code.

Documentation & Git/GitHub Usage - Learning Outcomes 1 (10%)

- Provide the following in your repository **README.md** file:
  - URL(s) to your games online.
  - URLs to resources used to build your games, i.e., **StackOverflow** posts, **Unity Forum** posts, etc.
- Commit messages **must** reflect the context of each functional requirement change.

Additional Information

- Attempt to commit at least **10** times per week.
- **Do not** rewrite your **Git** history. It is important that the course lecturer can see how you worked on your assessment over time.