Learning Summary Report

	Pass	Credit	Distinction	High Distinction
Self-Assessment (please tick)		✓		

Self-assessment Statement

	Included (please tick)
Learning Summary Report (This document)	✓
Semester Test with all corrections	✓
C# or C++ Programming Reference Sheet	✓
Principles of OOP Report	✓
Reflection	✓
At least 50% of Pass Tasks signed off as Complete	✓

Minimum Pass Checklist

	Included (please tick)
All of the Pass requirements	✓
100% of Pass Tasks signed off as Complete	✓
Concept Map	✓
Evidence of credit tasks that have been completed	✓

Minimum Credit Checklist, in addition to Pass Checklist

	Included (please tick)
All of the Credit requirements	
100% of Credit Tasks signed off as Complete	
Code for your program that demonstrates good OO design *including the use of polymorphism, implementing an interface and managing collections of objects*.	
UML class diagrams and UML sequence diagrams that communicate the design your custom program	
Description of what your program does and screenshots of your program in action	

Minimum Distinction Checklist, in addition to Credit Checklist

	Included (please tick)
All of the Distinction requirements	
Research report and associated pieces	

Minimum High Distinction Checklist, in addition to Distinction Checklist

Introduction

This section summarises what I learned about Object Oriented Programming. It includes a justification of the assessment pieces included in the portfolio, details of the coverage of the unit learning outcomes, and a reflection on my learning.

Overview of Pieces Included

Briefly describe what you have included in each section of the report.

- C# reference sheet
- Shape Drawer Program:
- Swinburne School Of Magic Program
- SwinAdventure Iteration 1-8
- Concept map
- Planet Rover/Robust Rover
- Document on good object-oriented programming practices.
- Document on Principles of Object-oriented Programming.
- Semester test result

Coverage of the Intended Learning Outcomes

This section outlines how the pieces I have included demonstrate the depth of my understanding in relation to each of the unit's intended learning outcomes.

ILO 1: Object Oriented Principles

Explain the principles of the object oriented programming paradigm specifically including abstraction, encapsulation, inheritance and polymorphism.

The following pieces demonstrate my ability in relation to this ILO:

- **Semester Test**: Demonstrates my basic understanding of C# programming language and the principles of Object-oriented Programming.
- **Concept Map**: Demonstrates my deeper understanding of the principles of Object-oriented Programming, how they are related to each other, and how they are related to other programming artefacts.
- **Document on principles of Object-oriented Programming**: Demonstrates my understanding of the four principles: Encapsulation, Inheritance, Polymorphism, and Abstraction.

[Describe the you have included in your portfolio that demonstrates you ability in relation to each outcome.]

ILO 2: Language Syntax

Use an object oriented programming language, and associated class libraries, to develop object oriented programs.

- **C# reference sheet:** Demonstrates my knowledge of the C# programming language syntax.
- **Shape Drawer, Swinburne School of Magic, SwinAdventure iteration**s: Demonstrates my ability to program using the C# OOP language.
- **Planet Rover/ Robust Rove**: Demonstrates my ability to program using the C++ OOP language.

ILO 3: Writing Programs

Design, develop, test, and debug programs using object oriented principles in conjunction with an integrated development environment.

- **SwinAdventure iterations**: Demonstrates my ability to design and develop a program in C#, as well as Unit testing using Xamarin IDE.
- **Planet Rover/ Robust Rove**: Demonstrates my ability to design and develop a program in C++, as well as Unit testing using the Visual Studio IDE.

ILO 4: Object Oriented Design

Construct appropriate diagrams and textual descriptions to communicate the static structure and dynamic behaviour of an object oriented solution.

- **Planet Rover's UML Diagram**: Demonstrates my ability to design a program based on what I have learnt in this semester and communicate it using a UML Class diagram.
- **SwinAdventure Iterations 6-8**: Demonstrates my ability to design a program based on what I have learnt in this semester and communicate it using UML Class diagrams and UML Sequence Diagrams.

ILO 5: Program Quality

Describe and explain the factors that contribute to a good object oriented solution, reflecting on your own experiences and drawing upon accepted good practices.

- **SwinAdventure iterations**: Demonstrates how to create unity and uniformity between all the classes in a program
- **Robust Rover program**: Demonstrates how to manage memory usage when programming using the C++ language.
- **Good programming practice documen**t: Demonstrates good practices in Object-oriented programming I have come up with from other tasks I have completed during the semester.

Reflection

The most important things I learned:

- Unit Testing and Test-driven Development
- Code documenting
- Use of UML Class and Sequence diagrams to communicate program designs
- Memory management in C++ programming.

The things that helped me most were:

- My lab tutor, who helped me clarify ambiguities in the tasks specifications.

I found the following topics particularly challenging:

- Interfaces in C#: this is mostly due to a shortage of practice exercises for this particular topic.

I found the following topics particularly interesting:

- Memory management in C++: This is because C++ is my preferred programming language.

I feel I learned these topics, concepts, and/or tools really well:

- Unit Testing: I am confident in my ability to create Unit Test to help check most, if not all of the possible results of each function. This is proven by the unit tests I created in the Planet Rover program and the SwinAdventure iterations.
- Object-oriented programming using C++ language as well as memory management in C++: I do not have much trouble programming in C++ language. This can be proven with my Planet Rover/Robust Rover codes.

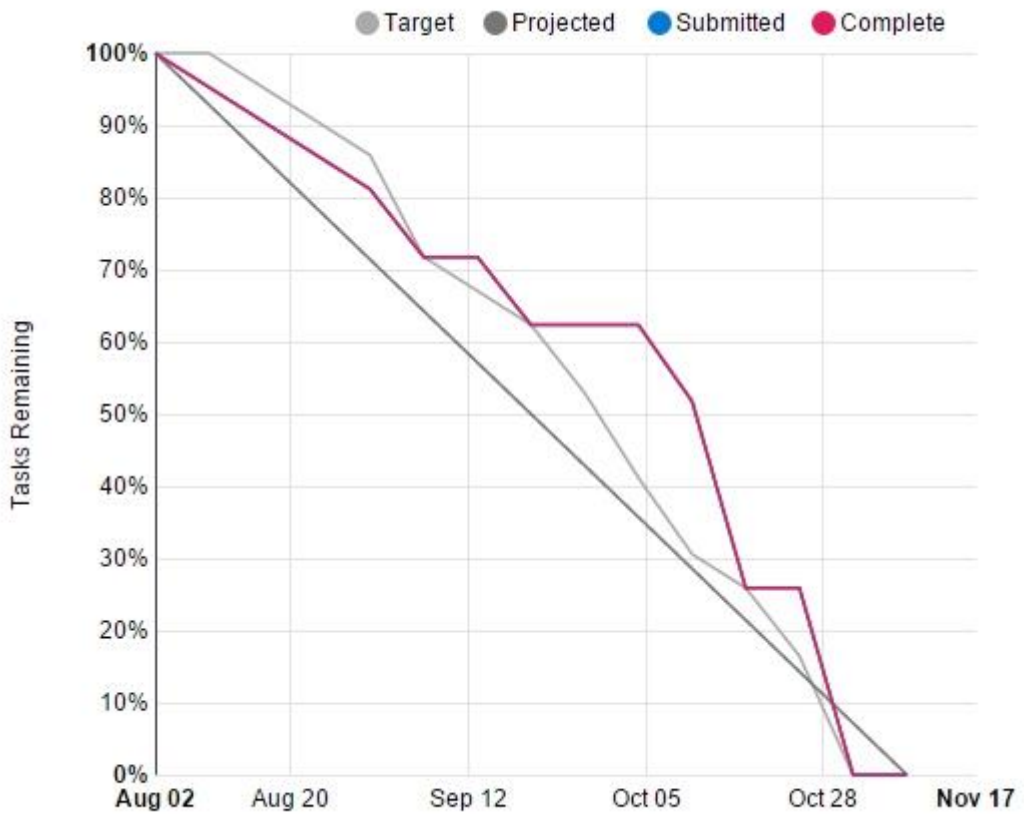
I still need to work on the following areas:

- UML diagrams: Most of the time I could only draw them after having completed the program, which means I have not been able to use it for designing programs yet, and can only use them to communicate ideas currently.
- Test-driven Development: Although I can use Unit Tests to test functions I have already written without any problems, I still have not been able to familiarize myself with the Test-driven Development approach.
- Using Xamarin IDE: I still have not been able to use Xamarin to its full capabilities yet.

My progress in this unit was …:

My progress was a bit slow and behind the targeted progress. However, it was more because of my weakness in writing formal submission rather than my skill in coding or my understanding of Object-oriented programming. As a result, after the first period, which was about the first four topics (with tasks mostly deal with programming), my progress was slowed down, due to the fact that I did not submit the first few concept tasks, the Principles of Object-oriented programming, and Concept map early and left until much later to finish.

On the other hand, because I had decent skills in programming, I managed to get ahead with programming tasks, such as the SwinAdventure iteration tasks and the Planet Rover/Robust Rover tasks (still late submission for the tasks that also involved UML diagram drawing), and managed to catch up with the targeted progress at several occasions.



This unit will help me in the future:

Almost everything I have learned in this unit will benefit me greatly in the future, both for my study progress in programming-related units, as well as my career. Since my aim is to become a game developer and designer, the knowledge and application of Object-oriented programming is indispensable to me.

If I did this unit again I would do the following things differently:

- Working on later assignments as soon as possible to avoid having a large workload at the end of the semester.
- Submit assignments as I finish them, instead of piling them up then submit them all together at a later point

Concluding Statement

In summary, I believe that I have clearly demonstrate that my portfolio is sufficient to be awarded a Credit grade.