SIT 771

OBJECT-ORIENTED DEVELOPMENT

Learning Summary Report

YUPENG WEN
224212855

Self-Assessment Details

The following checklists provide an overview of my self-assessment for this unit.

	Pass (D)	Credit (C)	Distinction (B)	High Distinction (A)
Self-Assessment				√

Self-Assessment Statement

	Included
Learning Summary Report	√
Pass tasks complete	√

Minimum Pass Checklist

	Included
All Credit Tasks are Complete on OnTrack	√

Minimum Credit Checklist (in addition to Pass Checklist)

	Included
Distinction tasks (other than Custom Program) are Complete	√
Custom program meets Distinction criteria	√

Minimum Distinction Checklist (in addition to Credit Checklist)

	Included
Something Awesome included	√
Custom project meets HD requirements	√

Minimum High Distinction Checklist (in addition to Distinction Checklist)

Declaration

I declare that this portfolio is my individual work. I have not copied from any other student's work or from any other source except where due acknowledgment is made explicitly in the text, nor has any part of this submission been written for me by another person.

Signature: YUPENG WEN

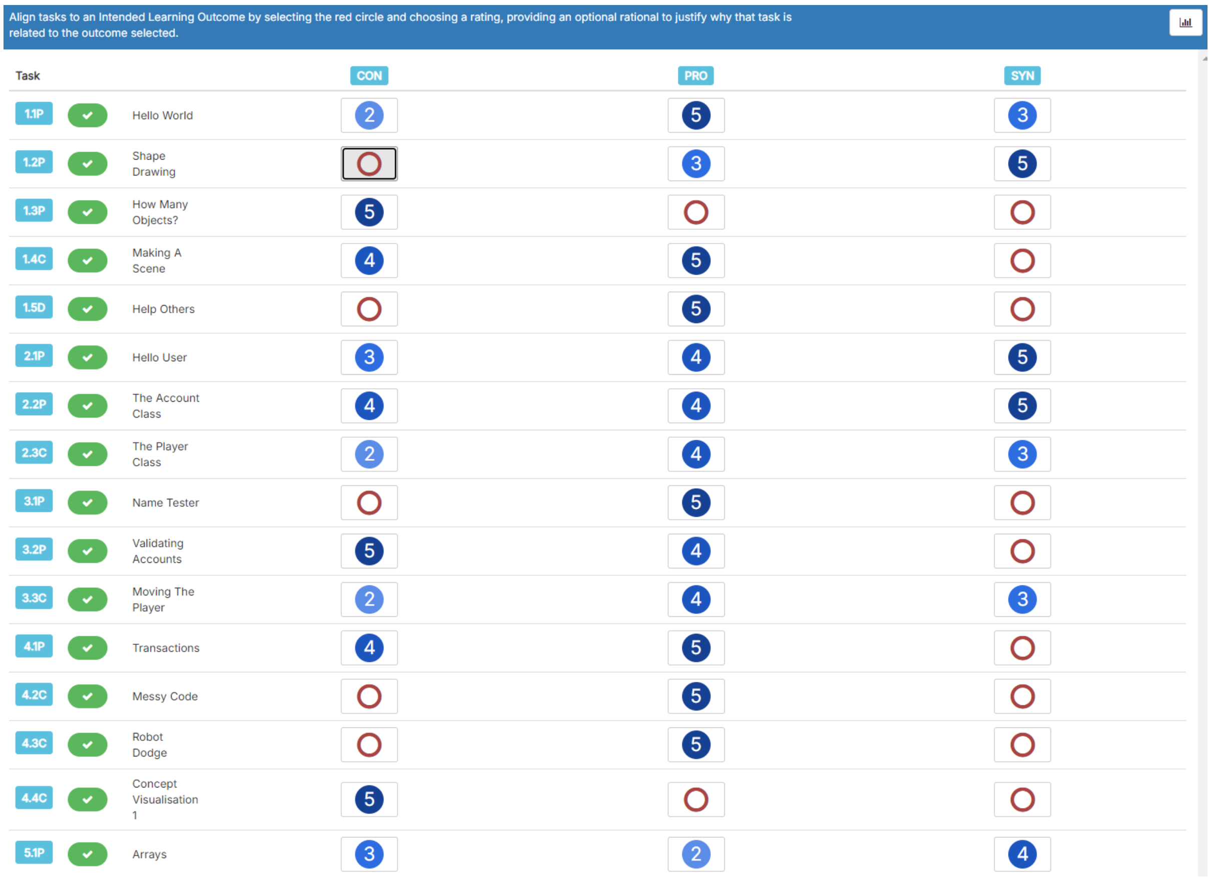
Portfolio Overview

This portfolio includes work that demonstrates that I have achieved all Unit Learning Outcomes for SIT771 to a **High Distinction** level.

I am writing to justify why I should receive a High Distinction grade for the SIT771 unit. Throughout this unit, I have applied both theoretical knowledge and practical skills to complete a series of complex tasks, consistently demonstrating a deep understanding of the key concepts, techniques, and methodologies.

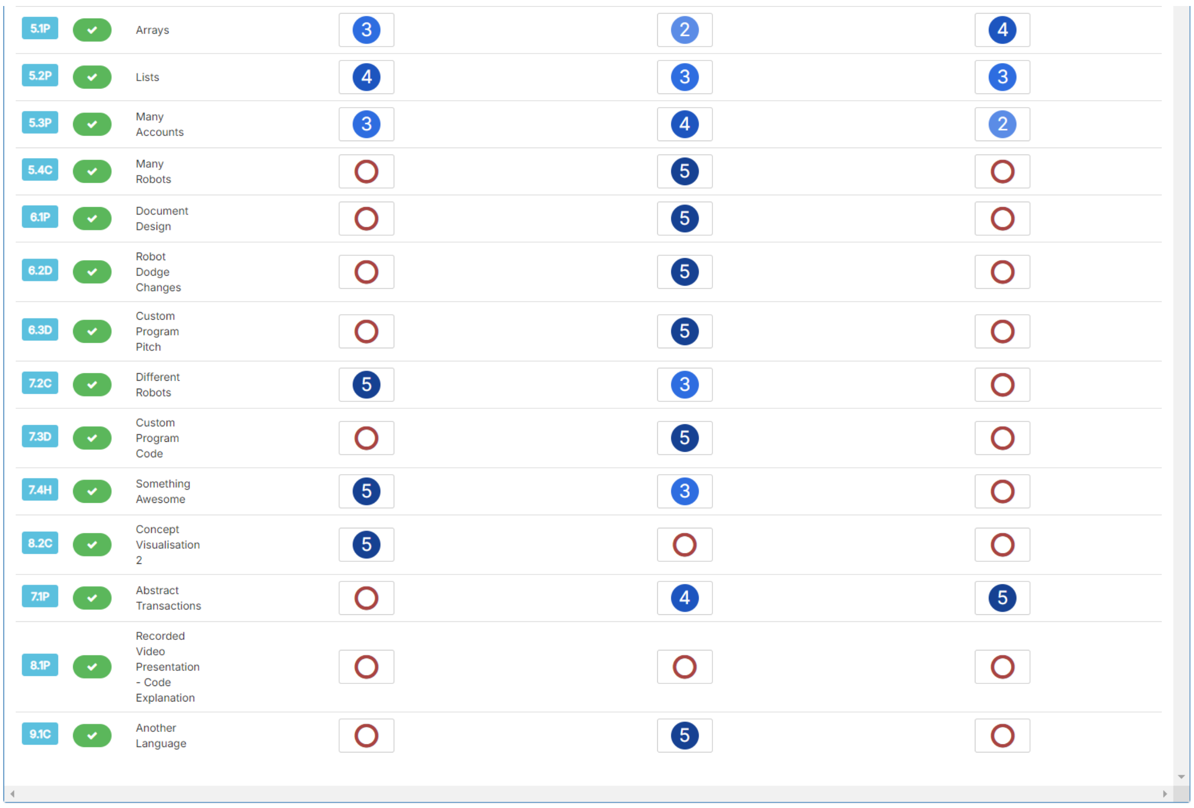
Each task I completed showcases my ability to master the core concepts of object orientation, as well as my flexibility in using C# to build robust programs. I have demonstrated proficiency in extending program functionality by incorporating suitable structures and precise syntax. My work reflects a solid grasp of object-oriented principles, such as encapsulation, inheritance, polymorphism, and abstraction, which I have applied to create efficient and scalable software solutions.

In summary, my portfolio demonstrates a comprehensive understanding of the unit's learning outcomes, highlighting my capability to apply both theory and practice to complex programming challenges. This justifies my application for a High Distinction grade in SIT771.



Align tasks to an Intended Learning Outcome by selecting the red circle and choosing a rating, providing an optional rational to justify why that task is related to the outcome selected.

Task			CON	PRO	SYN
1.1P	✓	Hello World	2	5	3
1.2P	✓	Shape Drawing	○	3	5
1.3P	✓	How Many Objects?	5	○	○
1.4C	✓	Making A Scene	4	5	○
1.5D	✓	Help Others	○	5	○
2.1P	✓	Hello User	3	4	5
2.2P	✓	The Account Class	4	4	5
2.3C	✓	The Player Class	2	4	3
3.1P	✓	Name Tester	○	5	○
3.2P	✓	Validating Accounts	5	4	○
3.3C	✓	Moving The Player	2	4	3
4.1P	✓	Transactions	4	5	○
4.2C	✓	Messy Code	○	5	○
4.3C	✓	Robot Dodge	○	5	○
4.4C	✓	Concept Visualisation 1	5	○	○
5.1P	✓	Arrays	3	2	4



5.1P	✓	Arrays	3	2	4
5.2P	✓	Lists	4	3	3
5.3P	✓	Many Accounts	3	4	2
5.4C	✓	Many Robots	○	5	○
6.1P	✓	Document Design	○	5	○
6.2D	✓	Robot Dodge Changes	○	5	○
6.3D	✓	Custom Program Pitch	○	5	○
7.2C	✓	Different Robots	5	3	○
7.3D	✓	Custom Program Code	○	5	○
7.4H	✓	Something Awesome	5	3	○
8.2C	✓	Concept Visualisation 2	5	○	○
7.1P	✓	Abstract Transactions	○	4	5
8.1P	✓	Recorded Video Presentation - Code Explanation	○	○	○
9.1C	✓	Another Language	○	5	○

Unit Learning Focuses

Throughout the unit, we have been using focus to direct your attention to the key learning outcome of each task. Kindly justify your understanding for each focus below:

- SYN (Syntax): Focus on understanding and remembering the C# and Python syntax.
  [Briefly describe how your understanding of syntax was developed throughout the unit]

After completing tasks 1.1P, 1.2P, 2.1P, 2.2P, 2.3C, 3.3C, 5.1P, 5.2P, 5.3P, 7.1P, and 9.1C, I have thoroughly learned the syntax rules and conventions for both C# and Python. This included mastering key concepts such as class definitions, interface implementations, and exception handling. Additionally, I gained expertise in C#'s advanced features, like delegates and lambda expressions, which required precise syntax knowledge for effective use. This comprehensive understanding has greatly enhanced my ability to write correct and efficient code in both languages.

- PRO (Process): Focus on how you use the syntax and concepts to build software solutions.
  [Briefly describe how your understanding of programming processes was developed throughout the unit]

Throughout the unit, I developed a deep understanding of programming processes by using conditional statements and for loops to control program flow and implement critical logic. By integrating these control structures into my code, I managed to handle complex scenarios and dynamic inputs effectively. This hands-on practice with decisionmaking and iterative processes significantly enhanced my proficiency in building robust and efficient applications, ensuring that my solutions are both reliable and well-structured.

- CON (Concept): Focus on the underlying programming and Object Oriented Programming concepts.

  [Briefly describe how your understanding of Programming and OOP concepts was developed throughout the unit]

Throughout the unit, my understanding of programming and Object-Oriented Programming concepts was deepened through comprehensive exploration and application. I studied foundational principles such as encapsulation, inheritance, polymorphism, and abstraction, which are core to OOP. By implementing these concepts in C# and Python, I was able to design and build scalable and modular software solutions. The unit emphasized how to use classes and objects effectively, apply design patterns, and create reusable code, which enhanced my ability to solve complex problems and develop well-structured applications. This thorough engagement with OOP principles has provided me with a robust framework for designing efficient and maintainable software.

Unit Learning Outcomes (ULOs)

Every unit at Deakin is designed to meet certain learning outcomes that the students can attain by the end. Below mentioned are the ULOs for SIT 771. Justify your learnings below.

> *For Pass: you need to indicate how you have demonstrated all Unit Learning Outcomes to a minimal level.*
>
> *For Credit: you need to indicate how you have demonstrated all Unit Learning Outcomes to a good level.*
>
> *For Distinction: you need to indicate how you have been able to apply all of the Unit Learning Outcomes in achieving the distinction tasks.*
>
> *For High Distinction: you need to indicate how you have been able to extend beyond the material presented in the unit.*

- ULO 1: Evaluate simple program code for correct use of coding conventions and use code tracing and debugging techniques to identify and correct issues.

By implementing tasks 4.3C, 5.4C, and 6.2D, I have demonstrated the correct use of coding conventions and applied code tracing and debugging techniques to identify and correct issues. The evidence in my portfolio includes reviewed and refined code that adheres to best practices for both C# and Python. I used debugging tools and techniques to trace code execution, identify logical errors, and resolve issues, which is evident in the enhanced functionality and reliability of my software solutions.

- ULO 2: Apply and explain the principles of object-oriented programming including abstraction, encapsulation inheritance, and polymorphism.

By implementing tasks 1.1P, 1.2P, 2.1P, 3.3C, 4.4C, and 5.3P, I have effectively demonstrated an understanding of object-oriented programming (OOP) principles, including abstraction, encapsulation, inheritance, and polymorphism. Through abstraction, I designed classes to represent abstract concepts and defined abstract methods, simplifying complex systems. Encapsulation was achieved by using private fields and public methods to control access to data, ensuring proper management of object states. I applied inheritance by creating base classes with shared functionality that derived classes could extend and customize, illustrating code reuse and extension. Polymorphism was demonstrated through method overriding and interface implementation, allowing objects to be treated as instances of their base class while providing specific implementations. These tasks reflect a thorough application of OOP principles in my software development work.

- ULO 3: Implement, and test small object-oriented programs that conform to planned system structures and requirements.

I have the ability to implement and test small object-oriented programs, as demonstrated in my portfolio. In task 5.3P, I successfully built an account transfer program that adheres to object-oriented principles and meets the required system structure. Additionally, in task 7.2C, I developed a graphic program featuring dynamic robots, further showcasing my ability to design, implement, and test programs that conform to planned system requirements and functionality. These tasks highlight my capability to apply object-oriented programming techniques in real-world applications.

- ULO 4: Design, communicate, and evaluate solution structures using appropriate diagrams and textual descriptions.

I have achieved ULO 4 by designing, communicating, and evaluating solution structures through the use of appropriate diagrams and textual descriptions. In task 7.3D, I designed a game called Tank Battle, which involved complex class structures and interactions. To construct the program, I utilized class diagrams that visually represented the relationships and communication between classes, helping to organize and plan the program effectively. Additionally, I documented the program details, further showcasing my ability to clearly communicate and evaluate the solution structure.

- ULO 5: Justify meeting specified outcomes by providing relevant evidence and critiquing the quality of that evidence against given criteria.

In task 9.1C, I redesigned a transaction program using Python, adhering to the required criteria for functionality and efficiency. Additionally, in task 7.4HD, I employed C#'s advanced features and server communication, ensuring that the program met its specified outcomes. These tasks provide relevant evidence that I have successfully achieved the desired results, and I have critiqued the quality of this evidence against given criteria, confirming that the outcomes were met to a high standard.

Reflection

The most important things I learnt:

The most important things I learned during this unit include understanding program structure, data structure, and how to effectively design a program. I gained valuable insights into organizing code through object-oriented principles, such as abstraction, encapsulation, inheritance, and polymorphism, which allowed me to create modular and maintainable software solutions. Additionally, I developed a deeper understanding of data structures and their role in efficiently managing and processing information within programs.

The things that helped me most were:

The things that helped me most throughout this unit were developing logical thinking, mastering process control, and understanding data structures for efficient data storage. Logical thinking allowed me to break down complex problems and devise clear, structured solutions. Process control, through the use of loops, conditionals, and methods, helped me manage the flow of the program and ensure that it responded correctly to different scenarios. Additionally, learning about data structures enabled me to store and access data efficiently, which is crucial for optimizing performance and scalability in software design. These skills were essential in building robust and effective programs.

I found the following topics particularly challenging:

Sometimes, I found the documentation for SplashKit to be confusing, as some of the APIs were not well-defined and it was difficult to find accurate or clear examples. This occasionally made it challenging to fully understand how to implement certain functionalities, leading to extra time spent troubleshooting and experimenting with the code.

I found the following topics particularly interesting:

I found algorithms to be the most interesting topic, algorithms are a wide topic, but when I learned how to implement a list, it gave me a great deal of understanding and practical help. Lists are fundamental data structures, and by learning how to efficiently implement and manipulate them, I gained key insights into how data can be stored, accessed, and processed in a program.

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

I feel I learned the concepts of encapsulation, inheritance, polymorphism, and abstraction really well. These core principles of object-oriented programming became second nature to me as I applied them across various tasks. Encapsulation helped me manage data securely within classes, inheritance allowed me to reuse and extend functionality efficiently, polymorphism enabled flexible and dynamic code, and abstraction simplified complex systems by focusing on essential features. Mastering these concepts has strengthened my ability to design clean, scalable, and maintainable software solutions.

Describe how focusing on syntax, concepts, and programming processes affected your learning in this unit:

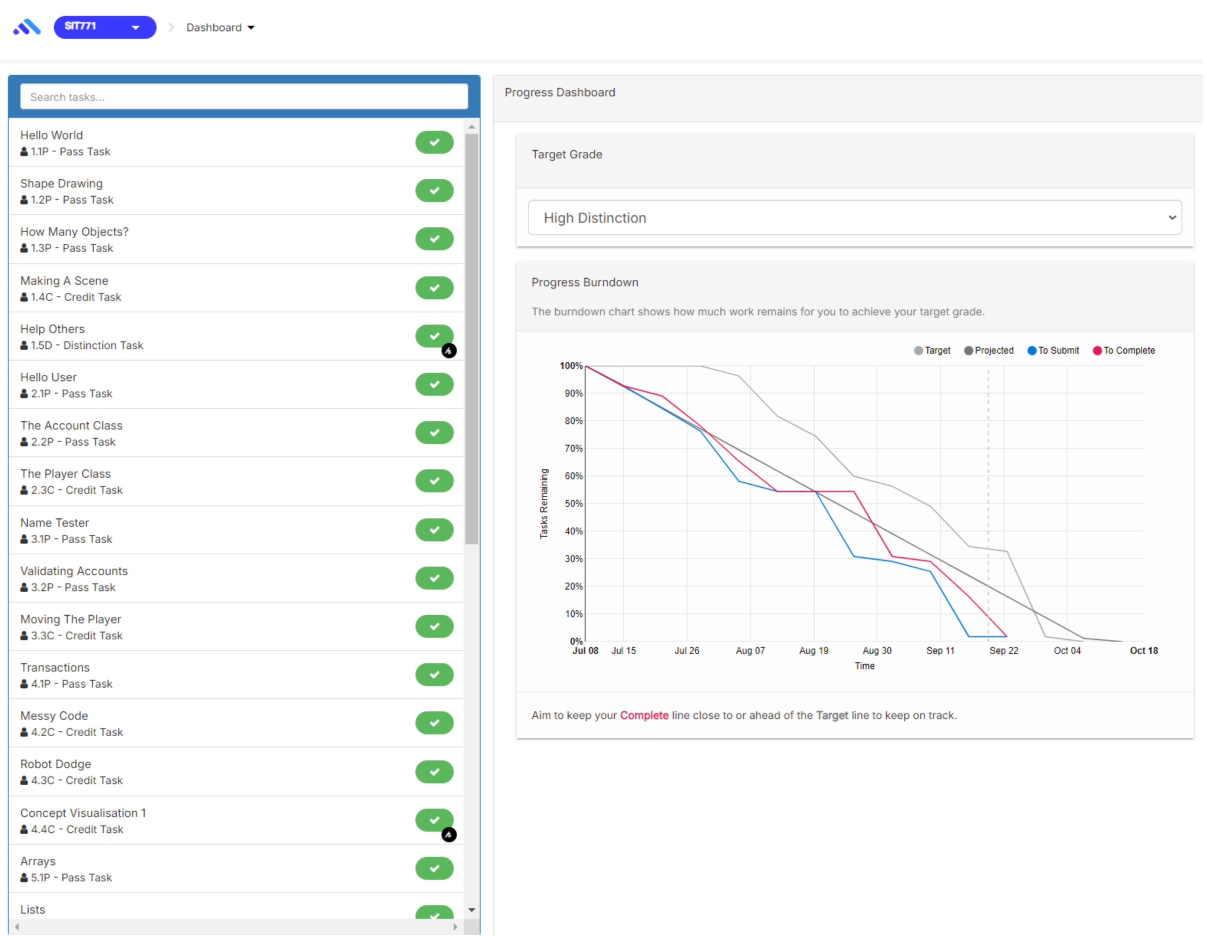
Focusing on syntax, concepts, and programming processes greatly enhanced my learning in this unit. Mastering syntax for C# and Python enabled me to write precise and error-free code, while understanding object-oriented principles like encapsulation, inheritance, polymorphism, and abstraction provided a solid foundation for designing modular and maintainable software. Additionally, grasping programming processes such as control flow and data management allowed me to develop effective solutions and debug issues systematically. Integrating these focuses into my tasks ensured that I completed them accurately and efficiently, ultimately improving my programming skills and understanding.

I still need to work on the following areas:

I still need to work on server management, microservices, and protocols. Enhancing my understanding of server operations and deployment will be crucial for building scalable applications. I aim to delve into microservice architecture to learn how to design modular and scalable systems effectively. Additionally, improving my grasp of communication protocols will enable me to better design and implement systems that interact across different platforms, ensuring data integrity and security. Addressing these areas will help me develop more comprehensive and robust software solutions.

My progress in this unit was …:

Everything is ok.



Hello World
1.1P - Pass Task ✓

Shape Drawing
1.2P - Pass Task ✓

How Many Objects?
1.3P - Pass Task ✓

Making A Scene
1.4C - Credit Task ✓

Help Others
1.5D - Distinction Task ✓

Hello User
2.1P - Pass Task ✓

The Account Class
2.2P - Pass Task ✓

The Player Class
2.3C - Credit Task ✓

Name Tester
3.1P - Pass Task ✓

Validating Accounts
3.2P - Pass Task ✓

Moving The Player
3.3C - Credit Task ✓

Transactions
4.1P - Pass Task ✓

Messy Code
4.2C - Credit Task ✓

Robot Dodge
4.3C - Credit Task ✓

Concept Visualisation 1
4.4C - Credit Task ✓

Arrays
5.1P - Pass Task ✓

Lists

Target Grade: High Distinction

The burndown chart shows how much work remains for you to achieve your target grade.

Aim to keep your Complete line close to or ahead of the Target line to keep on track.

This unit will help me in the future:

This unit will be highly valuable for my future studies and career. The deep understanding of object-oriented programming principles, such as encapsulation, inheritance, polymorphism, and abstraction, will serve as a strong foundation for developing complex software systems. Mastery of syntax, programming concepts, and processes will enhance my ability to write efficient, maintainable code and tackle various programming challenges. Knowledge of data structures and algorithms will aid in solving real-world problems effectively. Additionally, skills in debugging and refining code will be crucial for maintaining high-quality software. These competencies will be applicable across various programming languages and frameworks, supporting my growth in both academic and professional settings.

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

Practice More
I would dedicate additional time to practicing with real-world examples and coding exercises to reinforce my understanding of complex topics like algorithms and data structures.

Explore Advanced Topics

I would start exploring advanced topics such as server management, microservices, and protocols earlier to build a more comprehensive knowledge base and better prepare for future projects.