ASSESSMENT BRIEF	
Subject Code and Title	ISE102 Introduction to Software Engineering
Assessment	Assessment 2 – Simple Program
Individual/Group	Individual
Length	Executable program and source code.
Learning Outcomes	c) Understand the relationship between software & hardware. d) Utilise software engineering practices to solve simple application design problems. e) Demonstrate a foundational understanding of object-oriented programming.
Submission	Due by 11:55pm AEST/AEDT Sunday End of Module 4
Weighting	35%
Total Marks	100 marks

Context:

This assessment will demonstrate that you understand the theory and implementation of C++ in the context of a small application development. You will use the fundamental building blocks of software engineering to construct a small game. This process will help you develop larger and more complex application as you progress in software engineering.

Instructions:

Design and implement a **virtual slot machine** game in C++ utilising the knowledge and conventions covered in your module content and by your facilitator.

Application Design Requirements:

The game should start the player off with $2000, and should display the following main menu:

Player's chips: $2000

1) Play Slots

2) Quit Slot Machine

3) Credits

The game should include (but not limited to) the following functionality.

- If the player chooses 1, the program should ask the user to enter their bet.
- The program should check that the bet is legal.
- The program should then generate three random numbers, in the range of [2, 7] and output them to the screen.
- If all three numbers are '7', then award the player 10x their bet.
- If all three numbers are the same, (but not '7's), then award the player 5x their bet.
- If two out of the three numbers are the same, then award the player 3x their bet.
- Otherwise, the player loses their bet.

The program then calculates the player's current chip count and displays it on the screen.

- Finally, the main menu is displayed.

Build Quality:

The source code is required to display the following features:

- **Free of:**
  - Build warnings at Warning Level 3 for all build targets.
  - Build errors for all build targets.
  - All intermediate files, (.obj, .pdb, .ilk, …, files etc).
- **Commenting, Naming, Structure & Documentation:**
  - Code formatting is consistent, with good use of whitespace, tabbing and alignment.
  - Consistent and clear naming conventions are used.
  - Where necessary, **comments** should be used to clarify the purpose and use of data and functions. **Comments in the context of this assessment will also demonstrate understanding and provide evidence of academic integrity.**
  - Any necessary documentation should be included as a separate Readme.txt file.
- **An electronic copy of the source code (.cpp), solution file (.sln) and project file (.vcxproj) is required.**
  - Name the source code folder as: Source – Student Name
  - Name the solution as: SlotMachine.sln

Runtime Quality:

The source code is required to display the following features:

- Free of:
  - Bugs.
  - Crashes.

Interface Features:

The executable is required to provide an intuitive interface with the following features:

- Provide clear instructions.
- Controls are clearly identifiable and intuitive while playing.
- Design and layout of the in-game interface make effective use of screen space.
- All game state information is clearly presented on screen.

Submission Instructions:

Zip files containing the following.

Release Build Zip: A release build executable must be zipped and included with the submission. Ensure that project settings are set to Release when creating this build.

Include a Readme.txt and any dependencies.

Source Code Zip: All relevant source code files and project files must be zipped and included with the submission.

Naming & file structure for the zip file.

- ISE102_Assessment2_LastName_FirstName.zip
    - Assessment2_Build_LastName_FirstName.zip
    - Assessment2_Source_LastName_FirstName.zip

Assessment 2: Small Program

Assessment Attributes	Fail (Unacceptable) 0-49%	Pass (Functional) 50-64%	Credit (Proficient) 65-74%	Distinction (Advanced) 75 -84%	High Distinction (Exceptional) 85-100%
Subject Learning Outcome : *c) Understand the relationship between software & hardware.* *d) Utilise software engineering practices to solve simple application design problems.* *e) Demonstrate a foundational understanding of object-oriented programming..* The game implements a minimum of the following core features & qualities. • Ability to display the main menu with all three options. ü • Ability to allow the user to choose a correct menu option. • Ability to move to play slots screen when option 1 is chosen. • Ability to quit slot machine when option 2 is chosen. • Ability to display the credits when option 3 is chosen. • Ability to generate and display three random numbers that simulate the roll of the slot machine. • There are no build errors in both Debug and Release builds.	Not all features have been sufficiently implemented or implemented features contain bugs or errors affecting the functionality of the build.	Program meets the minimum requirements & feature set and demonstrates a functional understanding of the underlying programming principles.	Program meets the minimum requirements & feature set and demonstrates a proficient understanding of the underlying programming principles.	Program meets the minimum requirements & feature set and demonstrates a proficient understanding of the underlying programming principles. Some additional features have been attempted and implemented.	Program meets the minimum requirements & feature set and demonstrates an advanced understanding of the underlying programming principles. Some additional features have been attempted and implemented.

40%					
Subject Learning Outcome : *c) Understand the relationship between software & hardware.* *d) Utilise software engineering practices to solve simple application design problems.* *e) Demonstrate a foundational understanding of object-oriented programming.* The game implements the following core features & qualities. • Ability to allow the user to input a bet amount and check that the betted amount is valid. • Ability to calculate correctly and report the award on the betted amount. • Ability to calculate and report the correct new credits based on the award. 20%	Not all features have been sufficiently implemented or implemented features contain bugs or errors affecting the functionality of the build.	Program meets the minimum requirements & feature set and demonstrates a functional understanding of the underlying programming principles.	Program meets the minimum requirements & feature set and demonstrates a proficient understanding of the underlying programming principles.	Program meets the minimum requirements & feature set and demonstrates a proficient understanding of the underlying programming principles. Some additional features have been attempted and implemented.	Program meets the minimum requirements and feature set and demonstrates an advanced understanding of the underlying programming principles. Some additional features have been attempted and implemented.
d) Utilise software engineering practices to solve simple application design problems. The software project is organised and correctly submitted. • The submission is clean with no unnecessary files. • Folder structure and filenames conform to given specifications.	Project files and build not correctly separated. Unnecessary files included in project. Project, folder or filenames do not consistently conform to given specification.	All necessary project files and executables files have been separated and submitted. Project contains some unnecessary files but Build	Project contains some unnecessary files but Build and executable files are appropriated separated. Folders and filenames conform to given specifications.		No unnecessary file submitted as per submission requirements. Folders and filenames conform to given specifications. Build and executable files appropriated separated.

• A Release build executable has been submitted separately. 10%		and executable files are appropriated separated. Folders and filenames conform to given specifications.			
Subject Learning Outcome : *c) Understand the relationship between software & hardware.* *d) Utilise software engineering practices to solve simple application design problems.* *e) Demonstrate a foundational understanding of object-oriented programming.* The software project displays the following qualities. • The game is polished and feels smooth to play, going beyond the minimal requirements of the brief. • Data validity checks have been done. • The interface makes excellent use of screen space and shows consideration towards user experience. 15%	Little to no effort made in refining the user experience through testing or iterative application design. No Data validity checks have been implemented.	An attempt has been made to polish the user experience and aesthetic design. Some Data validity checks have been attempted.	The application represents some user testing. This is demonstrated through more considered and planned application design. Data validity checks are present in key blocks of code.	The application represents adequate testing. This is demonstrated through an effective user interface and iterative application design choices. Data validity checks are present throughout the code.	The application represents considerable testing. This is demonstrated through the polished user interface and iterative application design choices. Data validity checks are consistent throughout the code.
Subject Learning Outcome : *c) Understand the relationship between software & hardware.*	Little to no commenting. Code formatting is inconsistent, or no attempt made towards readability.	*Code formatting is consistent, with appropriate use of whitespace, tabbing and alignment.*	*Code formatting is consistent, with appropriate use of whitespace, tabbing and alignment.*	*Code formatting is consistent, with appropriate use of whitespace, tabbing and alignment.*	*Code formatting is consistent, with appropriate use of whitespace, tabbing and alignment.*

d) Utilise software engineering practices to solve simple application design problems. *e) Demonstrate a foundational understanding of object-oriented programming.* The software project displays the following qualities. • Code formatting i • s consistent, with good use of whitespace, tabbing and alignment. • Consistent and clear naming conventions is used. • Where necessary, comments are used to clarify the purpose and use of data and functions. Comments demonstrate clear understanding of the related code. • 15%		*Clear naming conventions are used.* *Comments are used to describe the purpose and use of key data and functions.*	*Consistent and clear naming conventions are used.* *Comments are used to clarify the purpose and use of data and functions.* *Comments demonstrate an understanding of the key areas of related code.*	*Consistent and clear naming conventions are used.* *Comments are used to clarify the purpose and use of data and functions.* *Comments demonstrate a proficient understanding of the related code throughout the project.*	*Consistent and clear naming conventions are used.* *Comments are used to clarify the purpose and use of data and functions.* *Comments demonstrate an advanced understanding of the related code.*

Fail (Unacceptable) **0-49%**	**Fail** grade will be awarded if a student is unable to demonstrate satisfactory academic performance in the subject or has failed to complete required assessment points in accordance with the subject's required assessment points.
Pass **(Functional)** **50-64%**	**Pass** is awarded for work showing a satisfactory achievement of all learning outcomes and an adequate understanding of theory and application of skills. A consistent academic referencing system is used and sources are appropriately acknowledged.
Credit **(Proficient)** **65-74%**	**Credit** is awarded for work showing a more than satisfactory achievement of all learning outcomes and a more than adequate understanding of theory and application of skills. A consistent academic referencing system is used and sources are appropriately acknowledged.
Distinction **(Advanced)** **75 -84%**	**Distinction** is awarded for work of superior quality in achieving all learning outcomes and a superior integration and understanding of theory and application of skills. Evidence of in-depth research, reading, analysis and evaluation is demonstrated. A consistent academic referencing system is used and sources are appropriately acknowledged.
High Distinction **(Exceptional)** **85-100%**	**High Distinction** is awarded for work of outstanding quality in achieving all learning outcomes together with outstanding integration and understanding of theory and application of skills. Evidence of in-depth research, reading, analysis, original and creative thought is demonstrated. A consistent academic referencing system is used and sources are appropriately acknowledged.