TAS Faculty
Assessment Task 2 Project
Year 11 Software Engineering, 2025

Nature of Task	Date of Task	Weighting
App Project	Term 2 - Week 8	40%

Description of the Task:

You are to design, code and document a software solution for an entertainment product. The software is to be developed in a high-level general-purpose programming language (Python).

Outcomes:

- describes methods used to plan, develop and engineer software solutions **SE-11-01**
- explains how structural elements are used to develop programming code **SE-11-02**
- applies tools and resources to design, develop, manage and evaluate software **SE-11-06**
- implements safe and secure programming solutions **SE-11-07**

You will be assessed on how well you:

The entertainment piece of software is to be an interactive solution. It will include an intuitive interface that displays directions and responses to the user's requests.

1. Download and Install Python 3.13 , Pygame and Appjar
2. Create a folder with necessary assets for your program
3. Create the planning documents as outlined in the instructions
4. Code your program and how well your program executes
5. Evaluation of your game self and peers

-

Instructions:

Steps	What I need to do
Software Development Explore the fundamental development steps in relation to their project.	Write a requirements definition for the given real-world problem. Identify the user specifications for the chosen software solution.
Designing Algorithms	Develop a structured algorithm using pseudocode or flowcharts including the use of subprograms and passing parameters. Algorithms should include sequence, selection, iteration and subprograms, and be described using a structure chart.
Data for Software Engineering	Define and discuss the use of the following data types. Select a minimum of 3 to use in your project: ● char and string ● Boolean ● real ● single precision floating point ● integer ● date and time. Create a data dictionary for use with your project. Define and discuss the use of the following data structures. Select a minimum of 2 to use in your project: ● arrays ● records ● trees ● sequential files.
Developing Solutions with Code	Convert your algorithm into code using: ● control structures ● data structures ● standard modules ● subprograms (including parameter passing). Define and discuss the following debugging tools used

	in your project: ● breakpoints ● single line stepping ● watches ● interfaces between functions ● debugging output statements. Document and implement at least one appropriate data structure that supports data storage. Describe the errors you experienced in the coding of your solution including: ● syntax ● logic ● runtime.

Submission Instructions/Time Allowed:
While class time will be allocated to the development of the task, you may be required to work on component at home to complete the task.

Please review the College policy on plagiarism located in the Assessment Booklet.

AI Traffic light indicator for this task. RED AI use is NOT acceptable

Planning Documents

Criteria	Exceptionally well executed (5)	Good, with room for improvement (3-4)	Meets minimum requirement (1-2)
Requirements definition	Explains the requirements	Describes a requirement	Identifies a requirement
User specifications	Explains the specifications	Outlines specifications	Identifies a specification
Structured algorithm using pseudocode or flowcharts	Clear Algorithm showing all processes	Most processes shown	Attempts an algorithm
Structure chart	Clearly shows all modules and flow of data	Shows some modules and flow of data	Attempts a structure chart
. Data types	All 6 explained with examples	Most explained	One explained
Data dictionary	All columns completed	Most completed	Attempted
Data structures	4 explained	Most explained	Attempted
Debugging tools	5 explained how used in your project	Most explained	Attempted
Data storage.	One explained	One described	One identified
Describe the errors	Describes syntax,logic and runtime	Identifies features of syntax, logic , runtime	Identifies one of syntax / logic / runtime
		TOTAL (out of 50)	

Python Code

Criteria		Exceptionally well executed (5)	Good, with room for improvement (3-4)	Meets minimum requirement (1-2)	Pts	
	Python, Pygame and Appjar Download	All successfully done: Python 3.13, Pygame, appjar	Most done	attempted		
	Modularization & Generalization	Program broken into well thought out elements that are of an appropriate length, scope and independence. Individual elements are written in a way that actively invites reuse in other projects.	Code elements are generally well planned and executed. Some code is repeated that should be encapsulated. Individual elements are often, but not always, written in a way that invites code reuse.	Code elements exist, but are not well thought out, are used in a somewhat arbitrary fashion, or do not improve program clarity. Elements are seldom written in a way that invites code reuse.		
	Design, Structure & Efficiency	Program is designed in a clear and logical manner. Control structures are used correctly. The most appropriate algorithms are implemented.	Program is mostly clear and logical. Control structures are used correctly. Reasonable algorithms are implemented.	Program isn't as clear or logical as it should be. Control structures are occasionally used incorrectly. Steps that are clearly inefficient are used.		
	Readability, Consistency & Naming	Coding style guidelines are followed correctly, code is exceptionally easy to read and maintain. All names are consistent with regard to style and are expressive without being verbose.	Coding style guidelines are almost always followed correctly. Code is easy to read. Names are consistent in style and expressive. Isolated cases may be verbose, overly terse or ambiguous.	Coding style guidelines are not followed and/or code is less readable than it should be. Names are nearly always consistent, but occasionally verbose, overly terse, ambiguous or misleading.		
	Initial Comments	Initial comments are complete. Internal documentation is complete and well suited to the program	Initial comments are complete but internal documentation is in some small fashion inadequate.	Initial comments are incomplete or internal documentation is inadequate.		
	User Interface	Screen based instructions and final output are clear, correct and attractive. Program is "user friendly" with informative and consistent prompts and messages.	Screen based instructions and final output are mostly clear, correct and attractive. Program is "user friendly" with informative and consistent prompts and messages.	Screen based instructions and final output are not clear, are not correct or are not attractive. And/or Program is not "user friendly.		
	Splash screen	Well designed and connected	Basic design and connected to program	Good design but doesn't connect to program		
	Correctness	Program correctly solves problem in all cases, exceeds problem specifications, meets language feature requirement	Program correctly solves problem in all or nearly all cases, but may have minor problems in some instances. All language feature requirements are used	Program solves problem in some cases, but has one or more problems. It meets all language feature requirements, but doesn't provide a solution to the problem		
				Total Points (out of 40)		
				Final Score		