# Knowledge assessment

## Criteria

### Unit code and name

ICTPRG441 | Apply skills in object-oriented design

ICTPRG439 | Use pre-existing components

ICTPRG434 | Automate processes

ICTGAM423 | Apply artificial intelligence in game development

### Qualification/Course code and name

ICT40120 | Certificate IV in Information Technology

## Student details

### Student number

880624021

### Student name

Yu Cheng

## Assessment declaration

*Note: If you are an online student, you will be required to complete this declaration on the TAFE NSW online learning platform when you upload your assessment.*

This assessment is my original work and has not been:

* plagiarised or copied from any source without providing due acknowledgement.
* written for me by any other person except where such collaboration has been authorised by the Teacher/Assessor concerned.

### Student signature and date

Yu Cheng

Version: 20220728

Date created: 28 July 2022

© 2022 TAFE NSW  
RTO Provider Number 90003 | CRICOS Provider Code: 00591E

This assessment can be found in the: [Learning Bank](https://share.tafensw.edu.au/share/items/d36df03f-9651-4d43-8c9d-a299699e8585/0/?attachment.uuid=fed43101-626e-4112-8bc8-3d5534d1194d)

The content in this document is copyright © TAFE NSW 2022 and should not be reproduced without the permission of TAFE NSW. Information contained in this document is correct at the time of printing: 8 March 2023. For current information please refer to our website or your Teacher/Assessor as appropriate.

## Assessment instructions

Table 1 – Assessment instructions

| Assessment details | Instructions |
| --- | --- |
| **Assessment overview** | The objective of this assessment is to assess your knowledge and performance in:   * researching, developing and implementing artificial intelligence (AI) solutions in games * writing scripts to automate solutions by using basic scripting processes and application-specific scripting options * identifying, evaluating and incorporating pre-existing (re-use) components from a library, or other source, as part of a software project * producing an object-oriented design from specifications, applying the cyclic process of iteration from identification of class, instance, role and type to the final object-oriented model of the application. |
| **Assessment event number** | 1 of 3 |
| **Instructions for this assessment** | This is a written assessment that assesses your knowledge of the unit.  This assessment is in 13 parts:   1. Derive high-level design from specification 2. Refine design 3. Document design 4. Identify potential re-use components 5. Evaluate and select re-use components 6. Incorporate re-use components into parent projects 7. Conduct research on AI strategies 8. Design, implement and test AI game strategy 9. Evaluate game and confirm with required personnel 10. Establish problem and applicable guidelines 11. Develop algorithms and solutions to given problem 12. Design and write script 13. Verify and review and document script and code.   And is supported by:   * Assessment feedback.   **Note**: This assessment may contain links to external resources. If a link does not work, copy and paste the URL directly into your browser. |
| **Submission instructions** | On completion of this assessment, you are required to submit it to your Teacher/Assessor for marking. If you are completing this as an online quiz, ensure you click the submit button when complete. You can save as you progress to avoid losing your work.  It is important that you keep a copy of all electronic and hardcopy assessments submitted to TAFE and complete the assessment declaration when submitting the assessment. |
| **What do I need to do to achieve a satisfactory result?** | To achieve a satisfactory result for this assessment all questions must be answered correctly.  If a resit is required to achieve a satisfactory result it will be conducted at an agreed time after a suitable revision period. |
| **What do I need to provide?** | * TAFE NSW student account username and password. If you do not know your username and password, contact your campus or service centre on 131601. * Computer or other device with word processing software and internet access. * Writing materials, if required. * USB drive or other storage method with enough free space to save work to. |
| **What the Teacher/Assessor will provide** | Access to this assessment and learning resources, including the student workbook and any supporting documents or links. |
| **Due date**  **Time allowed**  **Delivery location** | Refer to training plan  4 hours (indicative only)  TAFE NSW campus/ TAFE Digital Campus/ TAFE NSW Moodle/ a location determined by your Teacher/Assessor |
| **Assessment feedback, review or appeals** | In accordance with the TAFE NSW policy *Manage Assessment Appeals,* all students have the right to appeal an assessment decision in relation to how the assessment was conducted and the outcome of the assessment. Appeals must be lodged within **14 working days** of the formal notification of the result of the assessment.  If you would like to request a review of your results or if you have any concerns about your results, contact your Teacher/Assessor or Head Teacher. If they are unavailable, contact the Student Administration Officer.  Contact your Head Teacher for the assessment appeals procedures at your college/campus. |

## Part 1: Derive high-level design from specification

1. Read the statement carefully and indicate **true** or **false**.

Table 2 – True or false

| Statement | True or False |
| --- | --- |
| Object-oriented decomposition is a technique for decomposing a complex arrangement into the primary objects visible in the system. | T |

1. What are 3 of the main tools used in object-oriented documentation and development techniques?

Table 3 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Class diagrams/templates | x |
| 1. Object diagrams | x |
| 1. Object state diagrams | x |
| 1. Transition diagrams |  |

1. Match each principle of object-oriented design to its description.

Table 4 -– Matching

|  |  |  |
| --- | --- | --- |
| Description | Answer | Principle |
| Avoid duplication of code. You should not have the same block of code in more than one place. Never have a value hardcoded more than once, assign it to a constant. |  | 1. Inheritance |
| Hides unnecessary details from the user. The user doesn’t need to see or know about the entire process. |  | 1. Polymorphism |
| This focuses on what an object does, instead of how it works or how it is represented. Code that follows this principle shows only the required things while hiding the details. |  | 1. Don’t Repeat yourself (DRY) |
| Allows code to be reused when a class includes property of another class, therefore it can carry over everything from the other class. |  | 1. Encapsulation |
| This allows the way that something works to be redefined by either changing how it is done (called overriding) or changing the parts used to get it done (call overloading). |  | 1. Abstraction |

1. Identify the most accurate definition of an activity diagram.

Table 5 - – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. In most object-oriented techniques, activity diagrams are drawn for a single class to show the lifetime behaviour of a single object. |  |
| 1. Activity diagrams describe all the possible states that a particular object (or even the entire system) can get into. |  |
| 1. Activity diagrams are also useful for describing how an object moves through various states within its lifetime. |  |
| 1. Activity diagrams describe activities which involve concurrency and synchronisation, and which are a variation of state diagrams that focuses on the flow of actions and events. |  |

1. Identify the correct statement(s) below regarding different types of diagrams.

Table 6 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Sequence diagrams can sometimes fall short of offering the "big picture.” |  |
| 1. Sequence diagrams are sometimes known as event diagrams or event scenarios. |  |
| 1. Communication diagrams offer the same information as a sequence diagram and the information is presented in the exact same format. |  |
| 1. Communication diagrams can be a useful reference for businesses, organisations, and engineers who need to visualise and understand the physical communications within a program. |  |

1. What are 3 of the main tools used in object-oriented documentation and development techniques?

Table 7 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Class diagrams/templates |  |
| 1. Object diagrams |  |
| 1. Object state diagrams |  |
| 1. Transition diagrams |  |

## Part 2: Refine design

Read each question carefully and answer by selecting the appropriate response.

1. Which of the following are examples of a debugging methodology that could be used to debug a script for a logic error where the program runs successfully but doesn’t give the correct result?

Table 8 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. To find a logic error, one could go through the code line by line, use breakpoints or a watch process. |  |
| 1. Use the debugging tool built into the browser, which allows the program to run one line at a time, stopping at defined points to examine the code. |  |
| 1. Process the inputs by using variables and performing operations, such as arithmetic, repetition, and controlling the flow of execution. |  |
| 1. Use a debug tool to find the syntax error that is causing the logic error. |  |

1. To ensure the correct use of scripting language syntax, there are 3 primary constructs that must be considered. Which of the following correctly describes the 3 primary constructs?

Table 9 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Selection refers to when only certain lines of code are run when a specific condition is met. |  |
| 1. Sequencing involves clustering each of the codes into lines which are then run repeatedly. |  |
| 1. Sequencing runs the codes in a specified order until the end of the code is reached. |  |
| 1. Programming is a construct used to control the end result by selecting which lines need to be run. |  |
| 1. Iteration uses a loop function in order to run the same codes repeatedly |  |

1. Class diagrams are the main building blocks of object-oriented modelling, so it is important to understand the various class diagram relationships and how they affect a solution. Match each type of logical class relationship to its definition.

Table 10 – Matching

| Definition | Answer | Logical class relationship |
| --- | --- | --- |
| Refers to the formation of a particular class because of one class being built as a collection. For example, the class “library” is made up of one or more books, among other materials. |  | 1. Reflexive association |
| Refers to a directional relationship represented by a line with an arrowhead. The arrowhead depicts a container-contained directional flow. |  | 1. Aggregation |
| Refers to a type of relationship wherein one associated class is a child of another by virtue of assuming the same functionalities of the parent class. In other words, the child class is a specific type of the parent class. |  | 1. **Association** |
| This occurs when a class may have multiple functions or responsibilities. For example, a staff member working in an airport may be a pilot, aviation engineer, a ticket dispatcher, a guard, or a maintenance crew member. |  | 1. **Directed association** |
| A broad term that encompasses just about any logical connection or relationship between classes. |  | 1. **Inheritance/Generalisation** |

1. Which of the following statements that refer to aggregation and composition in object-oriented design are correct?

Table 11 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Person (parent) and leg (child) is an example of aggregation. |  |
| 1. Person (parent) and head (child) is an example of composition. |  |
| 1. Engine (parent) and engine inspection (child) is an example of aggregation. |  |
| 1. Hand (parent) and finger (child) is an example of composition. |  |

1. Identify the correct definition of specialisation within classes in object-oriented design.

Table 12 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Specialisation is the process of extracting shared characteristics from 2 or more classes and combining them into a generalised class. |  |
| 1. Specialisation means creating new subclasses from an existing class. |  |
| 1. Specialisation is the process of combining 2 or more classes and mapping them in a diagram. |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 13 – True or false

| Statement | True or False |
| --- | --- |
| Generalisation within classes in object-oriented design is a relationship between a generalised class (or parent) and a specialised class (or child). |  |

## Part 3: Document design

Read each question carefully and answer by selecting the correct matching response.

1. Match each of the following principles of algorithms with their application in programming.

Table 14 – Matching

| Application in programming | Answer | Principle |
| --- | --- | --- |
| Programs should provide information and feedback to the user, so that they know what’s happening. The script should include feedback such as messages and prompts and present the design and information in a way that is easily understood. |  | 1. Precise |
| The algorithm should provide a solution in the most efficient way, for example fewer lines or loops. When coding, a programmer should reduce the number of operations and frequency of inputs and outputs. |  | 1. Descriptive |
| All the essential features of the solution would be shown in the algorithm, making it easy to code. The programmer won’t need to guess what should be done by the script; it should be straightforward to convert the algorithm into code. |  | 1. Handle errors |
| The algorithm should be desk–checked before coding to find logic errors. When the solution is coded, the same logic should be followed. There should also be a way to handle any runtime errors, such as an error-handling function. |  | 1. User friendly |
| By understanding the problem first, the solution can be correctly described with the algorithm. When the algorithm is written to find the correct solution, the coding should also find the correct solution. |  | 1. Must end |
| By desk-checking your algorithm, any problems with the algorithm not ending (for example an endless loop) can be found. By following the algorithm, the code should also end correctly. |  | 1. Efficient |

1. Match each scripting language syntax with its correct description.

Table 15 -- Matching

| Description | Answer | Scripting language syntax |
| --- | --- | --- |
| Some languages require that the upper and lower case need to be considered when naming variable, functions. Example:  The variable *FirstName* is a different variable to *firstName* |  | 1. Opening and closing statements |
| If using punctuation such as *‘, {, (* with a statement, there must be in pairs to open and close the statement. Example:  *while (condition) {*  *statement(s);*  *}* |  | 1. Case sensitivity |
| Internal documentation that adds more description to a code and makes it easier to maintain. They may be one line or over multiple lines. Example:  *// a one line comment*  */\* a long multi-line*  *Comment \*/* |  | 1. Comments |

1. Match each common scripting technique/construct used to automate processes with its correct description.

Table 16 -- Matching

| Description | Answer | Scripting techniques |
| --- | --- | --- |
| Allows the computer to make simple decisions based on the value of a variable. It may produce one action or give an alternative action. |  | 1. Functions |
| Are used to store multiple values in one variable, which uses index numbers to identify each value. |  | 1. Loops |
| Allow you to execute the same code more than once. It prevents duplication of code by allowing the program to go through the code either a set number of times or until a condition changes. |  | 1. Arrays |
| A named section of a program that performs a specific task. They make sure that the code is modular and reusable and prevents duplication of code. |  | 1. Selection |

1. Match each symbol to the concept it represents in an activity diagram.

Table 17 – Matching

|  |  |  |
| --- | --- | --- |
| Symbol | Answer | Concept |
| Represented by lines and arrows |  | 1. Step |
| Diamond or black rectangle |  | 1. Start |
| A pre-condition symbol (black circle) |  | 1. End |
| A circle surrounding black circle |  | 1. Flow |
| Rectangle with text |  | 1. Branch |

1. Match each user and technical script documentation methods used in script writing with its correct description.

Table 18 – Matching

| Description | Answer | Method |
| --- | --- | --- |
| Is a form of user documentation providing users with an overview of the script and information regarding the installation and usage of the script. |  | 1. Internal comments in code |
| Provides a list of frequently asked questions with answers to help users with the script related questions. |  | 1. Documentation tools within the language |
| Are annotations in the source code of the script that are ignored by compilers or interpreters. They are usually used as internal documentation to describe code and make the code more readable for developers. |  | 1. Read-Me documentation |
| Usually allow auto-generation of API documentation from comments in the source code. Comments must be written in a specific format using specific tags. |  | 1. FAQ documentation |
| Is the complete expression of what the script does. It consists of the actual text of the program that is compiled or interpreted. |  | 1. Source code |

1. Within the Unified modelling language (UML) specification, diagrams are grouped into 2 groups. Select the appropriate responses.

Table 19 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Behaviour diagrams show the static parts of a system at different levels of abstraction or different views. |  |
| 1. Behaviour diagrams show the dynamic parts of a system that helps to understand how parts of a system change over time. |  |
| 1. Structure diagrams show the dynamic parts of a system that helps to understand how parts of a system change over time. |  |
| 1. Structure diagrams show the static parts of a system at different levels of abstraction or different views. |  |

## Part 4: Identify potential re-use components

1. **Correctly order** for the software development life cycle (SDLC).

Table 20 – Ordering

|  |  |
| --- | --- |
| Steps/Process | Order |
| Maintenance | 6 |
| Requirement collection and analysis | 1 |
| Coding | 3 |
| Design | 2 |
| Installation / Deployment | 5 |
| Testing | 4 |

1. Which of the following statements about development and design methodologies are correct?

Table 21 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. A disadvantage of Waterfall software development is that mistakes made early on are difficult to fix. | x |
| 1. In an Agile project, there is no appetite to reduce the amount of documentation. |  |
| 1. In Agile development, cycles can be as little as a week or 2. | x |
| 1. In a traditional Waterfall project, documentation is not produced at each stage of the SDLC. |  |

1. Which of the following statements about library content and structures are correct?

Table 22 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. With a well-documented library, a programmer does not need to understand the inner workings of the code. | x |
| 1. A library is designed such that the functionality is encapsulated in a module with well-defined functions, procedures, methods, and data structures that meet common requirements. | x |
| 1. A library can only contain a class or collection of related classes, but not contain functions and procedures. |  |
| 1. Libraries of functions and objects, such as those you will find natively within a programming language, are never a form of reusable component. |  |

1. Which of the following statement about types of reusable components and what it entails are correct?

Table 23 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Reusable components come in many forms. | x |
| 1. Objects, classes, and code that you have specifically designed and built yourself can be reused within one application or can even be re-used across different applications. | x |
| 1. It includes third party objects and code, written by others outside your organisation. |  |
| 1. Complex concepts such as LINQ and simple as math library are not included. |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 24 – True or false

| Statement | True or False |
| --- | --- |
| Agile software development involves numerous small iterations of the SDLC and numerous incremental releases of working software. | T |

## Part 5: Evaluate and select re-use components

1. Which of the following statements is correct in relation to re-use component licensing agreements and implementation costs?

Table 25 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Re-usable packages may be changed in any way if you have paid for use. |  |
| 1. Re-usable code is used to save time and money in development. |  |
| 1. Before using any re-usable package, you must be sure that you have the right to use it under licensing agreements. |  |
| 1. Re-usable code does not save money, but it can save time in development. |  |

1. Which of the following statements are correct in relation to re-use component requirements?

Table 26 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. The generality of a component reveals whether it has fewer particular capabilities or not. |  |
| 1. Scalability, maintainability, performance, and portability are all examples of non-functional requirements. |  |
| 1. Errors caught in the functional requirement gathering stage are the cheapest to fix. |  |
| 1. It is relatively easy to change or alter non-functional requirements. |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 27 – True or false

| Statement | True or False |
| --- | --- |
| Project designs which are NOT at risk of overengineering and integration issues can benefit from using pre-existing components. |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 28 – True or false

| Statement | True or False |
| --- | --- |
| In accordance with organisational guidelines, when using pre-existing components, it is important to be familiar with all the clauses in the vendor licensing agreement ensure compliance. |  |

1. Which of the following are organisational guidelines and standards that should be followed in a software development project?

Table 29 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Use of comment and naming standards when writing code |  |
| 1. Use of iterative design based on reusable components |  |
| 1. Code of conduct |  |
| 1. Data governance policies |  |
| 1. User manuals and admin manuals |  |

## Part 6: Incorporate re-use components into parent projects

1. When undertaking testing, there are 4 clear levels and several testing techniques we can use to test at various levels. Match each level of testing to its description.

Table 30 – Matching

| Description | Answer | Level of testing |
| --- | --- | --- |
| Testing of the full application occurs at this stage to ensure that the system is ready for release to a client. Best practice is that this is undertaken by a tester not involved in the development in an environment that closely mirrors the production environment. |  | 1. Acceptance testing |
| Individual components are combined and tested as a group to make sure they work together as expected. The purpose is to find any problems or issues in the interaction between the components. |  | 1. Unit test |
| The final check by an independent tester or the customer before final sign off on release of the software system or application into production. |  | 1. Integration testing |
| Here the individual components or units are tested to make sure they are fully functional as per the specifications. The first level of testing and components should pass at this level before undertaking the higher levels of testing. |  | 1. System/End to end testing |

1. Which of the following statements are correct regarding re-use component version compatibility and dependency?

Table 31 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| a) Re-usable components are always compatible with the latest operating systems and devices. |  |
| b) Dependencies can lead to slow installation and affect reliability. |  |
| c) Instead of relying on a library for little components, it is appropriate to code smaller features and functions directly into the application. |  |
| d) Dependencies arise when a re-usable component is only compatible with an older operating system. |  |

1. Which of the following are a part of the testing types by re-use components?

Table 32 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Stress Testing |  |
| 1. UI testing |  |
| 1. Data unconditioned testing |  |
| 1. Exploit testing |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 33 – True or false

| Statement | True or False |
| --- | --- |
| Dealing with changes within a software development project is less expensive, than using an explicit waterfall approach due to less analysis and documentation being required to implement new software requirements. |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 34 – True or false

| Statement | True or False |
| --- | --- |
| Testing cannot be made more efficient with the use of automated testing tools. |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 35 – True or false

| Statement | True or False |
| --- | --- |
| Repository tools applicable to using pre-existing components are helpful for software developers; however, they do not promote a ubiquitous development environment, and they fail to mitigate the effect of missing, outdated, or incompatible dependencies. |  |

## Part 7: Conduct research on AI strategies

1. Which of the following statements about path-finding algorithms in game development are correct?

Table 36 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Dijkstra’s algorithm is greedy, meaning that it finds the next best answer in the hopes that the result will be the best solution for the entire problem. |  |
| 1. In Depth First Search, the algorithm searches all nodes of the same depth before moving on to the next level down. |  |
| 1. A \* is considered one of the best techniques in path finding and graph traversal. |  |
| 1. A\* selects nodes based on the value 'f', one parameter equal to the sum of the other 2 parameters 'g' and 'h'. At each step, it selects the node/cell with the highest “f” and processes that node/cell. |  |

1. Match each of the following terms relating to gaming and AI to their correct definition.

Table 37 – Matching

| Definition | Answer | Term |
| --- | --- | --- |
| A subset of AI, particularly focusing on developing algorithms that will help the machine adapt and change in response to new data. |  | 1. Parameter |
| An in-game character or entity that interacts with other items in its environment using artificial intelligence. |  | 1. Scripting |
| A variable inside the model that helps it to make predictions. |  | 1. Event |
| A game action completed by user input. For example, when the player presses a button on their controller and the character on the screen jumps. |  | 1. Agent |
| Another word for programming or coding; designing the core game logic. |  | 1. Machine learning |

1. Read the statement/s carefully and indicate **true** or **false**.

Table 38 – True or false

| Statement | True or False |
| --- | --- |
| When creating an NPC, one problem that developers tend to face is the concept of ‘predictability’. Human players often learn the behaviours of NPCs, and then abuse the systems and code that controls them. |  |

1. Which of the following are industry standards and guidelines that are potentially applicable to game developers?

Table 39 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| a) Code of ethics |  |
| b) International Game Developers Association |  |
| c) To make fun games |  |
| d) Pre-Production |  |

1. Read the statement carefully and indicate **true** or **false**.

Table 40 – True or false

| Statement | True or False |
| --- | --- |
| The abbreviation SOLID is widely recognised as the most effective object-oriented programming philosophy. |  |

## Part 8: Design, implement and test AI game strategy

1. Which of the following statements about object-orientated programming practices are correct?

Table 41 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. A single responsibility, job, or purpose should be assigned to each class. |  |
| 1. Derived or child classes should not be able to replace their base or parent classes without causing unexpected behaviour. |  |
| 1. No client should be forced to use an interface that is irrelevant to them. |  |
| 1. High-level modules/classes should not be dependent on low-level modules/classes, but rather on abstractions. |  |
| 1. Classes, modules, functions, and other software entities should be open to modification but closed to extension. |  |

1. Which of the following statements about the overall architecture of game engines are correct?

Table 42 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Unity is extremely powerful but inaccessible to novice game developers. |  |
| 1. A developer may not have to employ a Game Engine at all, but this will require them to handle most of the technical work themselves. |  |
| 1. Utility AI evaluates all the different actions that an NPC could take and awards a score to each one. |  |
| 1. The state transition logic in behaviour trees is organised in a hierarchical tree structure with the states as leaves, rather than being scattered over the individual states. |  |
| 1. Goal orientated action planning can be used to create intelligent characters in a variety of gaming genres but relies on the use of massive finite state machines. |  |

1. Match each of the following components of a test plan to their correct description.

Table 43 – Matching

| Descriptions | Answer | Component |
| --- | --- | --- |
| Specifies when testing must come to a halt. This section outlines the intended outcomes of QA activities, providing testers with a baseline against which to measure actual outcomes. |  | 1. Defect management |
| Outlines supporting equipment for testing, reporting, and other related tasks. |  | 1. Exit parameters |
| Specifies how bugs will be reported, to whom they will be notified, and what each bug report must include. |  | 1. Risk management |
| Describes the issues that may arise during software testing and that the program may face if it is released without enough testing. |  | 1. Environment |
| Describe the setting for the characteristics, configuration, and availability. |  | 1. Tools |

1. Identify whether the statement about how NPC’s work is **true** or **false**.

Table 44 – True or false

| Statement | True or False |
| --- | --- |
| State machines, which are a standard means of controlling diverse software and hardware, are frequently used to control NPCs. State machines are used in video games to determine the behaviour of AI agents. |  |

1. Select all the object-oriented programming practices.

Table 45 – Multiple Choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Liskov’s substitution principle |  |
| 1. Dependency inversion principle |  |
| 1. Single responsibility principle |  |
| 1. Duplicate code principle |  |
| 1. Interface segregation principle |  |

## Part 9: Evaluate game and confirm with required personnel

1. Read the statement carefully and indicate **true** or **false**.

Table 46 – True or false

| Statement | True or False |
| --- | --- |
| When creating an AI game, the design brief lays out the critical parts of the planned project, such as its goals, scope, and strategy. It informs design decisions and guides workflow from conception to completion. |  |

1. Which of the following statements about beta testing are correct?

Table 47 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| a) The product should have all the features that are planned for the release version. |  |
| b) Test participants should have expertise in the field of programming. |  |
| c) The product should be usable, allowing for infrequent crashes. |  |
| d) Test participants should complete real-world scenarios while interacting with an app, and they should do it in the real environment (not in lab environments). |  |

1. Which of the following statements about the client feedback procedure are correct?

Table 48 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Ensure the client’s feedback is stored to be handled at end of year reviews. |  |
| 1. Encourages the client to provide feedback in a constructive manner. |  |
| 1. May result in providing additional information or assistance to a client. |  |
| 1. Responding to client feedback is optional. |  |

1. Match each of the following phases of production to its correct definition?

Table 49 – Matching

|  |  |  |
| --- | --- | --- |
| Phase of production | Answer | Definition |
| Will take up around 20% of the game design process. It involves planning, mapping, scheduling, research and gathering of resources for the production process ahead. |  | 1. Production |
| Will take 20% of the design process. This includes sound design and visual effects, quality assurance, refining and packaging and marketing. |  | 1. Pre-production |
| Will take up about 60% of the design process. It is when the bulk of the work is done. It's all systems go! Animation, 3D modelling, programming teams all begin working in tandem. |  | 1. Postproduction |

1. Select all that apply to new possibilities of AI.

Table 50 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. AI enhances the attractiveness of game players over time. |  |
| 1. Artificial intelligence data creates simulated gaming ambience with motives, situations, and activities in mind. |  |
| 1. Limited data is needed for AI algorithms to ensure the best possible reactions to events. |  |
| 1. Artificial Intelligence is now used in different ways to lead the gaming industry on a high level into the future. |  |

## Part 10: Establish problem and applicable guidelines

1. A common approach to transforming an identified problem into a computer program is the software development life cycle. Match each step to the activity performed.

Table 51 – Matching

| Activity | Answer | Step |
| --- | --- | --- |
| Create a solution based on the analysis of the requirements. |  | 1. Code |
| Define the process needed to automate and consider input and outputs addressed in this topic. |  | 1. Design |
| Write the instructions in a computer language. |  | 1. Document |
| Write user and technical information resources. |  | 1. Analyse |
| Ensure the program produces expected results. |  | 1. Test and debug |

1. Which of the following statements is correct regarding the analysis phase?

Table 52 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. The most important step in finding the correct solution is writing the code out. |  |
| 1. Functional issues are more important than implementation issues in the analysis phase. |  |
| 1. Breaking a problem down into a single problem statement can help you focus on a solution. |  |
| 1. Implementation issues refer to what is being attempted, while functional issues involve how it be achieved. |  |

1. Which of the following is not a part of understanding a requirement?

Table 53 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. What format the input is in? |  |
| 1. What format the output is in? |  |
| 1. What are the required outputs? |  |
| 1. Where does the input come from? |  |

1. When reading a policy and procedure document for automating processes, which policies should the document contain?

Table 54 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Scope of the policy |  |
| 1. Program code |  |
| 1. Purpose of the policy |  |
| 1. Definitions of key terms |  |

1. Identify which of the following questions about design requirements are correct.

Table 55 – Multiple choice

|  |  |
| --- | --- |
| Questions | Correct |
| a) What program blocks can be recognised? |  |
| b) What is the expected input from the algorithm? |  |
| c) How is the solution composed from these blocks? |  |
| d) What the next steps of the design process are? |  |

## Part 11: Develop algorithms and solutions to given problem

1. Which of the following statements about algorithms are correct?

Table 56 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. An algorithm is like a standard operating procedure created by an organisation to carry out a routine activity. |  |
| 1. Creating an algorithm is always useful no matter how large or small the task. |  |
| 1. Scripts or programs are often written to automate tedious tasks. |  |
| 1. An algorithm is not user-friendly, inefficient and causes errors. |  |

1. Which of the following are principles to follow in designing a good algorithm?

Table 57 – Multiple choice

| Principle | Select the correct answer/s |
| --- | --- |
| a) An algorithm should be user-friendly and provide information and feedback to the user running it. |  |
| b) An algorithm should maximise its use of the computer’s resources. |  |
| c) An algorithm should include a course of action for all possible situations. |  |
| d) An algorithm should represent the solution correctly. |  |

1. Two common ways to express the structure of an algorithm are with pseudocode and flowcharts. Which of the following statements are correct about expressing algorithms?

Table 58 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. You should follow a consistent style and logic of writing pseudocode. |  |
| 1. You should begin coding as soon as you have completed your pseudocode and flowchart. |  |
| 1. There are standards and style guides for developing pseudocode to ensure minimal variation in pseudocode syntax. |  |
| 1. Pseudocode is a programming language that can be used to create programs. |  |
| 1. Flowcharts are a diagram-based way of representing processes and algorithms. |  |

1. Match each control structure to its correct description

Table 59 – Matching

| Description | Answer | Control structure |
| --- | --- | --- |
| Where one task is performed after another in a specified order. In pseudocode, each task is on a separate line and has the same indentation as the other tasks. |  | 1. Iteration |
| A choice is made between alternative courses of action. A condition is tested (often using a comparison operation) and tasks are performed depending on whether the test is true or false. |  | 1. Sequence |
| Tasks are performed repeatedly for a certain number of times or until some condition is met. |  | 1. Selection |

1. Which of the following are correct about a sequence regarding pseudocode?

Table 60 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Each task is represented by rectangles and are executed from the top to bottom. |  |
| 1. Selection is represented by the IF-THEN-ELSE construct. |  |
| 1. The tasks are executed one after the other, beginning with the task at the top and ending with the task at the bottom. |  |
| 1. Each task is on a separate line and has the same indentation as the other tasks in the sequence. |  |

1. Which if the following are helpful when ensuring an algorithm takes all situations into account?

Table 61 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. The aid of an experienced programmer in the final check before coding begins |  |
| 1. The elimination of testing stages |  |
| 1. Desktop testing other ways to verify the code and the logic behind it |  |
| 1. The process of explaining the algorithm to someone else |  |

## Part 12: Design and write script

1. When writing script and code in Blueprint, data types help ensure that programs run correctly. Match each data type to its description.

Table 62 – Matching

| Description | Answer | Data type |
| --- | --- | --- |
| Variables with true or false values particularly in conditional expressions. |  | 1. Array |
| Used for variables with textual data. |  | 1. String |
| These types of variables hold items of the same data type. |  | 1. Boolean |
| These types of variables describe the properties of an item where each property could have different data types. |  | 1. Object |

1. In Blueprint, a Function is code written to perform a specific task. Which of the following statements are correct regarding Functions?

Table 63 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. To use a Function that has been defined, it is called by writing its name and providing the required arguments or parameters. |  |
| 1. When a Function is called, it has multiple entry points designated by a node. |  |
| 1. When a Function is created, the objects that can and cannot access those Functions cannot be specified. |  |

1. Desk-checking the solution means checking the algorithm by following the instructions that have been created using different sets of inputs. Which of the following are valid steps of desk-checking?

Table 64 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Desk-check the algorithm using the sample inputs. This is the process of recording the value of each variable as it executes the statements in the algorithm. |  |
| 1. Compare the results obtained from the algorithm with those manually calculated earlier. |  |
| 1. Execute the algorithm only if a logic error is present or a change in variable has been missed. |  |
| 1. Ensure that different sets of test data are used to demonstrate the correctness of the algorithm. |  |

1. When creating required abstract design when writing the algorithm, which of the following should be completed?

Table 65 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Accumulating processing steps from output steps to modify the code immediately. |  |
| 1. Finding the requirements to write an algorithm that will automate the issue. |  |
| 1. Design algorithm by using the input-process-output model and translating this into pseudocode. |  |
| 1. Identifying each variable as either an input, output, or both. |  |
| 1. Disassembling your blocks of code and sequences in the correct order. |  |

1. In the case of the following game creation scenario, what is a required step?

You have been tasked with creating a program to accept multiple scores from a user when shooting a virtual arrow at a target, and to display the individual score and the average of all scores, as well as message when the shot is complete, according to the following criteria:

* The score will be input from the keyboard and be a whole number.
* A mark that is in the middle or second-to-middle circle will display a ‘Great job!’ message.
* Any other mark will display a ‘Better luck next time’ message.
* The program will end when a shot does not hit the target at all.

Table 66 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Remove the value function in the script. |  |
| 1. A loop is required, as multiple scores can be entered. |  |
| 1. Complex mathematical operation is required to create this statement. |  |
| 1. No extra steps are required. |  |

## Part 13: Verify and review and document script and code

1. A debugger is a program that is used to test and debug programs. Select the common features in debugging.

Table 67 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| a) Single stepping |  |
| b) Double stepping |  |
| c) Stopping or breaking |  |
| d) Slicing |  |

1. What is a debugging methodology that could be used to debug a script for a syntax error, which crashes the program?

Table 68 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. A syntax error in script can be debugged by reviewing the code itself looking for incorrect syntax, using syntax checking tools such as an IDE syntax checker (For example, Brackets has JSLint and ESLint installed). |  |
| 1. By using the developer tools that are installed by default in all modern browsers. |  |
| 1. By using an online checking tool that helps with script syntax errors. |  |
| 1. By using a try-catch statement to resolve the syntax error. |  |

1. Which of the following are script documentation methods used at a technical level?

Table 69 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Describing the system and problem |  |
| 1. Creating an algorithm |  |
| 1. Sourcing code or program listing |  |
| 1. Making internal comments within the program listing |  |
| 1. Change request register |  |

1. Read the statements about identifying problems and identify which of them are correct.

Table 70 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. It is unimportant to have collaboration when you must first meet the basic requirements of the problem statement. |  |
| 1. After the script is written it is important that you and your team can identify any problems within it because only then can you consider what additional features may (or may not) be required. |  |
| 1. You and your team need to consider if the chosen solution is practical for the typical environments, it will be used in. |  |
| 1. Identifying problems in the script is easy without team members because it ensures that nothing was missed by an induvial. |  |

1. Read the following statements about errors and indicate which of them are correct.

Table 71 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. Logic errors can give unpredictable results and may crash your program. |  |
| 1. A fatal error causes a program to terminate with warning. The program restarts and any changes made in the program are saved. |  |
| 1. When the interpreter finds a syntax error, it stops and outputs an error message. |  |
| 1. An exception or run-time error can happen when the script interpreter knows what to do with a piece of code but can’t perform the action. |  |

1. Which of the following statements are correct about the sign-off process?

Table 72 – Multiple choice

| Answer choices | Select the correct answer/s |
| --- | --- |
| 1. The sign-off process involves passing non-reviewed documentation to one stakeholder who has no responsibility for signing off on the documentation. |  |
| 1. The appropriate person or people to approve and sign-off the application will vary depending on the project stakeholders involved. |  |
| 1. An appropriate person or people usually include the client’s competitor and may also include a junior team member. |  |
| 1. Any changes that need to be made to the application after sign-off will restart the review and sign-off process. |  |
| 1. A document that shows the application has been approved and signed-off should include information such as the name of each person, position, date and signature. |  |

## Assessment feedback

*NOTE: This section must have the Teacher/Assessor and student signature to complete the feedback. If you are submitting through the TAFE NSW online learning platform, your Teacher/Assessor will give you feedback via the platform.*

### Assessment outcome

Satisfactory

Unsatisfactory

**Assessor feedback**

Has the assessment declaration for this assessment event been signed and dated by the student?

Are you assured that the evidence presented for assessment is the student’s own work?

Was reasonable adjustment in place for this assessment event?

*If yes, ensure it is detailed on the assessment document.*

*Comments*:

### Assessor name, signature and date

### Student acknowledgement of assessment outcome

*Would you like to make any comments about this assessment?*

### Student name, signature and date