# Assessment event 2 of 3: Project

## Criteria

### Unit code and name

ICTPRG434 | Automate processes

ICTPRG439 | Use pre-existing components

ICTGAM423 | Apply artificial intelligence in game development

ICTPRG430 | Apply introductory object-oriented language skills

### Qualification/Course code and name

ICT40120 CERT IV in Information Technology Game Development

## Student details

Student name

Gavin Lampe

Student number

880644379

Version: 20221220

Date created: 20 December 2022

© TAFE NSW 2023  
RTO Provider Number 90003 | CRICOS Provider Code: 00591E

This assessment can be found in the TAFE NSW [Learning Bank](https://share.tafensw.edu.au/share/logon.do?.page=searching.do?in%3DC1b145167-45e0-41ec-9f64-92af668e3e54%26q%3D%26type%3Dstandard%26sort%3Drank%26dr%3DAFTER%26page%3D1).

The content in this document is copyright © TAFE NSW 2023 and should not be reproduced without the permission of TAFE NSW. Information contained in this document is correct at time of printing: 05 July 2023. For current information please refer to our website or your teacher or assessor as appropriate.

## Assessment instructions

Table 1 Assessment instructions

| **Assessment details** | **Instructions** |
| --- | --- |
| **Assessment event overview** | The objective of this assessment is to assess your knowledge and performance in introductory programming tasks using an object-oriented programming language including tool usage, documentation, debugging, and testing techniques.  Write scripts to automate solutions by using basic scripting processes and application-specific scripting options.  Identify, evaluate and incorporate pre-existing (re-use) components from a library, or other source, as part of a software project.  Research, develop and implement artificial intelligence (AI) solutions in games.  This assessment is in 3 parts:   * Part 1: GDD * Part 2: Create a monster battler game * Part 3: Testing |
| **Unit assessment guide** | Refer to the unit assessment guide (UAG) before attempting this assessment event. The UAG contains information including assessment requirements and how to achieve a satisfactory result. |
| **Submission instructions** | When you complete this assessment:   * read the checklist at the end of the assessment to make sure you have completed everything * keep a copy of all the electronic and hardcopy assessments you submit to TAFE NSW * make sure you have completed the assessment declaration before you submit. |

## Task instructions

The assessor will use the criteria outlined in the following tasks to determine if you have satisfactorily completed this assessment event. Follow these instructions to ensure you demonstrate the required knowledge and skills.

## Part 1: Create a GDD

Using the template provided create a simple Game Design Document (GDD) for your project in part 2.

Within your GDD please provide instructions on how to play your game. The short manual will be instructing the players on how to play your game. Include screen shots and written steps to follow.

## Part 2: Create a monster battler game

Create a monster battler game with the following features:

1. Monster Battle Mechanics: Develop a battle system where two monsters engage in combat. Each monster takes turns selecting actions from a list of available actions. (At least 4 different actions)
2. Player-Controlled Monster: Allow the player to control one of the monsters in the battle. Implement controls that enable the player to choose actions and strategies during the battle. (At least 4 different actions)
3. AI-Controlled Monster: Create an AI-controlled monster that utilises a state machine for decision-making. The state machine should have at least three distinct states, each influencing the monster's behaviour and move selection. For example, the AI might have a defensive state where it priorities healing when low on health, or an aggressive state where it chooses powerful attacking moves when high on health.
4. Display AI State: Incorporate a text box in the game's interface that displays the current state of the AI-controlled monster. This provides visibility to the player regarding the AI's decision-making process and current strategy.
5. 2D Game: Implement the project as a 2D game. You may design and create visually appealing sprites and backgrounds to enhance the gaming experience.

Make sure your can close your game with an on-screen button.

If there was any issues with your game after submission you will be given feedback. It will identify any areas not covered or covered incorrectly by your code. Please amend any omissions and errors and resubmit.

## Part 3: Testing

Ensure you have debugged your code with no syntax and semantic errors in your code.

Run the following tests on your game, ensuring each test preforms correctly.

You may need to break down tests into multiply smaller tests. Write what is expected to happened in the Expected Results column and what actually occurs in the Actual results column.

|  |  |  |
| --- | --- | --- |
| **Test** | **Expected Results** | **Actual Results** |
| Player can perform each ability | \* All actions should take one turn except Run Away.  Spell Attack should:  1. Always deal damage  2. Play spell animation  Melee Attack should:  1. Sometimes miss  2. Deal damage  3. Play melee animation  Heal should:  1. Heal an amount  Run away should:  1. Enemy health = 100  2. Remove player sprite  3. Show Coward screen | \* PASS  Spell Attack  1. PASS  2. PASS  Melee Attack  1. PASS  2. PASS  3. PASS  Heal  1. PASS  Run Away  1. PASS  2. PASS  3. PASS |
| AI can perform all abilities | 1. Sleeping until hit.  Idle animation plays  2. Simple attack when ‘normal’. Plays animation  3.Heavy attack when ‘angry’. Plays animation  4.Plays ‘hurt’ animation when hit  5. Can regain health  6. Turns red when dead | 1. PASS  2. PASS  3. PASS  4. PASS  5. PASS  6. PASS |
| States change according to situation | 1. Sleeping state until health goes below 100  2. Normal state when health is below 100 and above 70 and below 50 but above 30  3. Angry state when health is 70 or lower and above 50  4. Heal state when health is 30 or lower  5. Death state when health is less than 1 | 1. PASS  2. PASS  3. PASS  4. PASS  5. PASS |