# Assessment event 2: Skills

## Criteria

### Unit code and name

ICTICT449 - Use version control systems in development environments

ICTPRG430 - Apply introductory object-oriented language skills

ICTPRG441 - Apply skills in object-oriented design

### Qualification/Course code and name

ICT40120 | Certificate IV in Information Technology

## Student details

Student name

Oscar Dryden

Student number

881100187

Version: 20230829

Date created: 29 August 2023

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

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: 16 June 2025. 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 aim of this assessment event is to assess your knowledge and performance in  This assessment is in 5 parts:   * Part 1: Plan * Part 2: TDD * Part 3: Create High-score Game * Part 4: Test the high-score system * Part 5: Git submission   And is supported by:   * Assessment feedback   **Note**: This assessment may contain links to external resources. Access to the long URL is provided via the External resources – Links and URLs section located at the end of this document. |
| **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, submit it for marking:   * 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. |

**Part 1: Plan**

You have been tasked with creating a small highscore game. There must be some gameplay that can be scored, such as points collected, distance travelled or speed doing a task.

There should be power ups and obstacles to enhance game play.

You must use a modular approach using object-oriented design and polymorphism to create your game.

You will need to implement a high-score table using C#

In your game’s code, you must do the following:

1. Read from and write to one text file
2. Implement a class that uses arrays of primitive data types twice.
3. Implement one class that contains two overloaded constructors.
4. Implement one class that uses interfaces.
5. Implement polymorphism once for code extensibility
6. Select and use at least
   1. three language data types
   2. three operators
   3. three expressions
7. Implement at least two classes that each contain at least four instance variables
8. Use correct language syntax for at least
   1. one selection (if, switch)
   2. two iteration (loops)

**Part 2: TDD**

Your plan will be written in a Technical Design Document (TDD) to ensure you meet design specifications to satisfy the requirements. Create a Technical Design Document (TDD) using the Template provided. The TDD design will clarify user requirements with user on submission.

**Part 3: Create High-score Game**

Create the high-score game.

Don’t forget to develop your game according to your planned design and following C# coding conventions.

**Part 4: Test the high-score system**

1. Unit tests isolate parts of your game to test. Perform two tests on your high-score code and write the results in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Expected outcome | Actual Outcome | Pass? |
| Type in a name in the title screen. | The name gets saved between scripts and scenes | Works as expected. | Pass |
| Ending the game with a score higher than the lowest score | The name typed in gets saved and overrides the lowest score | Works as expected. | Pass |

1. Provide three screenshots of you using breakpoints to debug your code to inspect variables.

A computer screen with a screen and text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Part 5: Git submission

You must submit with a link to Github, you must commit to a staging branch and push changes from staging to main. Please follow git standard practices.

Submit your assessment on moodle, you will require a link to your git, and if there is any, documention as a PDF.

You will receive any feedback on moodle, respond to feedback, and resubmit the assignment with any changes

This page is not required for online assessment submissions.

### Student assessment declaration

This assessment is my original work and has not been:

* copied from any source without proper referencing
* written for me by any other person except where such collaboration has been approved by a teacher or assessor.

Student signature and date

### Reasonable adjustment

Reasonable adjustment was in place for this assessment event.

If so, please provide details of any reasonable adjustment strategies that were implemented:

[Insert reasonable adjustment strategies]

### Assessment outcome

Satisfactory  Unsatisfactory

Comments

[Insert 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