Government of Western Australia
North Metropolitan TAFE

Portfolio Assessment Task 3

Qualification national code and title	ICT50220 Dip Advanced Programming
Unit/s national code/s and title/s	ICTPRG535 – Build advanced user interfaces ICTPRG547 – Apply advanced programming skills in another language

Assessment type (☑):

☐ Questioning (Oral/Written)

☐ Practical Demonstration

☐ 3rd Party Report

☒ Portfolio

Assessment Resources:

Python3 interpreter.

Python IDE, like PyCharm, or VSCode (the latter is not supported by the college), with the ability to use Python Virtual Environments.

Access to Office 365 and Microsoft Word.

Git and access to GitHub.

Use of some of these items may not occur in this part of the assessment task.

Assessment Due

This item is due:

- **Week 10, 17:00 (5pm) on the day of the scheduled lecture**

Refer to Blackboard for most accurate dates, which may alter due to unforeseen circumstances. We also endeavour to update these documents at the same time.

It is advantageous for you to attempt to meet this deadline.

Assessment Instructions:

Portfolio Assessment Task 3

Qualification national code and title	ICT50220 Dip Advanced Programming
Unit/s national code/s and title/s	ICTPRG535 – Build advanced user interfaces ICTPRG547 – Apply advanced programming skills in another language

Scenario

You are employed as a junior software developer at a boutique software house called *Softwares-R-Us* in Perth.

You have been assigned to the Games team and will provide some of the code required for various games that the company builds and releases.

You will be updating the existing code from the previous tasks to sort a list of players based on their scores.

Instructions

Your code must adhere to certain style guides, the most important being PEP-8 – Style Guide for Python Code. You should familiarise yourself with PEP-8 before continuing.

Use of a Git repository is standard practice at *Softwares-R-Us* and most of the steps require the use of Git and GitHub.

There are multiple steps in this task, and you must perform each step to a satisfactory level. Please note that you'll need the results of the tasks outlined in this assessment tool for future tasks as well.

You may answer any questions in the provided template (if available) and the use of screenshots is encouraged. However, you must also provide the actual source code as a ZIP-file of the project for any programming tasks. *Make sure to remove the Virtual Environment folder (*`venv` *or* `.venv`*) from the ZIP-file before uploading.*

You must document your code properly. Use docstrings where needed including but not limited to entire classes and methods. Use inline comments to clarify certain parts of code.

If you use any external resources, you should provide references.

Assessment Instrument:

Portfolio Assessment Task 3

Qualification national code and title	ICT50220 Dip Advanced Programming
Unit/s national code/s and title/s	ICTPRG535 – Build advanced user interfaces ICTPRG547 – Apply advanced programming skills in another language

Step 1 – Knowledge Question (40-70 words)

In your own words, describe what sorting is in general.

Step 2 – Knowledge Question (60-100 words)

Research sorting algorithms. Describe advantages and disadvantages for at least three different sorting algorithms. Please provide references for external resources.

Step 3 – Knowledge Question (20-50 words)

In your own words, describe why you generally need comparison operators to successfully sort a list of objects.

In addition, describe how you *could* sort a list of objects without adding comparison operators.

Step 4 – Implement comparison operators and sort

In this step, you will be sorting a list of player objects based on their score. This step has multiple parts to it.

a. Add a private instance variable to the `Player` class that will hold the score (a *positive* integer value). Provide a getter (property) and a setter method for this value.
b. Implement the comparison operators `__eq__`, `__ge__`, etc.) for the `Player` class. Implement each so that the comparison uses the score.
c. Add unit tests to test the new operators.
d. Implement a static method for the `Player` class that will sort a list of `Player` objects in descending order (higher scores come first). Choose an algorithm of your liking based on the answers you provided to the Knowledge Questions and *describe why you chose it.*
*IMPORTANT: you **must** implement this algorithm yourself; you may not use Python's* `sort` *method.*
e. Add unit tests to test the new sorting functionality.
f. Add and commit all changes, then push your changes to the remote repository on GitHub.

Submitting your work

ZIP your entire project into a single file.

Make sure to *remove the Python Virtual Environment folder (called* **venv** *or* **.venv***) from the ZIP-file* before uploading it into the Blackboard assessments area.