Department of Computer Science

Summative Coursework Set Front Page

Module Title	Programming in Python
Module Code	CS2PPNU
Lecturer responsible	Dr Wenwen Liu
Type of Assignment (e.g., technical report, set exercise, in-class test)	Coursework
Individual or Group Assignment	Individual
Weighting of the Assignment	40%
Word count/page limit	Complete notebook prompts. 500-word limit for written prompts.
Expected hrs spent for the assignment (set by lecturer)	16
Items to be submitted	A single **`.zip`** archive containing: 1. Fully executed Jupyter notebook (**`.ipynb`**) 2. Exported HTML copy of above (**`.html`**) 3. Data file: `cardata_modified.csv` 4. Three module files (**`.py`**) 5. Configuration file: `config.json` Required archive structure detailed below.
Work to be submitted on-line via Blackboard Learn (**Gradescope**) by	**2025** April 14th **(**Monday**)** 17**:00**
Work will be marked and returned by	**202**5 **May 5th (**Monday**)**
Artificial Intelligence Tools	May not be used

Note

By submitting this work, you are certifying that you have read the assessment guidelines, which are displayed in the folder of Assessment on the Blackboard course for this module, and that you have conformed to and understand the associated policies and practices, including those on:

- Submitting your own work, not that of other people or systems, and the associated penalties for Academic Misconduct
- Submitting by the specified deadline, and the penalties associated with late submission (if allowed)
- The exceptional circumstances system
- For students with relevant needs, attaching with a green sticker

1. Assessment classifications

This coursework assesses your ability to:

- implement common computer science algorithms to design solutions to programming problems with Python scripts and software applications;
- assemble code that incorporates imperative, functional, and object-oriented programming paradigms;
- demonstrate best practices and conventions for writing clean and efficient code.

In general, you will gain credit for:

- preparing and submitting required files as requested;
- successful implementation of the specified coding tasks;
- writing efficient, functional code;
- providing thoughtful, clear, well-structured written responses.

Your assignment will be marked according to the marking scheme provided below. The scheme is designed so that the collectively weighted assignment mark will correspond to the following qualitative degree classification descriptions:

Classification Range	Typically, the work should meet these specifications:
First Class (>= 70%)	Outstanding/excellent work with correct codes and results. This work demonstrates coding proficiency with high efficiency and based on advanced techniques. Evidence of independent research into the methods used and a thorough justification of applications of these methods is presented clearly.
Upper Second (60-69%)	Good work with few mistakes. Some minor tasks have not been carried out or are not completely correct. Coding with good efficiency. Evidence of good knowledge of core concepts, with good explanations and justifications.
Lower Second (50-59%)	Demonstrates knowledge of core concepts but with some mistakes. Explanations and justifications of methods used are logical but limited in depth. Coding with average efficiency. Most tasks have been carried out with sufficient accuracy.
Third (40-49%)	Some parts of the assignment are missing and/or have partially correct results. Most tasks have not been carried out with sufficient accuracy. Results may not be correct or technically sound. Mistakes in application of knowledge and shows some misunderstandings. Explanations and justifications of methods used are not clear or logical. Coding might be inefficient.
Pass (35-39%)	Some significant part of the assignment is missing and/or has partially correct results. Gaps in knowledge and many mistakes, little evidence of understanding. Methods used are not well explained or justified. Coding is notably inefficient.
Fail (0-34%)	Many aspects of the assignment are missing, or there are large gaps in knowledge and significant mistakes, also showing limited understanding. Lack of logical explanations behind the methods used.

2. Assignment description

This assignment consists of **four tasks**. Each of these will be used to assess your implementation of several components of the Python language.

A detailed breakdown of the Marking Scheme is provided later in this document.

Task 1 – Structuring and Analysing Network Data

Exploring the concept of networks, or graphs, with the **dolphins.tsv** dataset, you will design implementations of code to create edge list and neighbour list representations of the data and to perform simple analyses of its characteristics. Two implementations of the neighbour list creation will differ in efficiency. You will be prompted to verify that your code is working correctly, and you will provide a written assessment of the work you have performed. The functionality of your code will be demonstrated within the provided **CS2PPNU_CW1.ipynb** Jupyter notebook, where further instructions are provided, and will be supplemented by the module file, **network.py**, which you will create.

Task 2 – Data Processing

Using the **cardata.csv** file, you will explore the manipulation of data files. Executing your developed elements of **processing.py** within the **CS2PPNU_CW1.ipynb** Jupyter notebook, you will demonstrate the results of your processing routines. Working through this notebook's prompts, you will read, manipulate, and write data, which w ill be used in Task 3. Processed data will be saved to **cardata_modified.csv**.

Task 3 – Python Classes

Using the **cardata_modified.csv** file produced above, continue working in the **CS2PPNU_CW1.ipynb** Jupyter notebook. There, you will find detailed specifications for the creation of a "Tournament" class in Python. This class will represent instances of competitive tournaments, with functionality to simulate competitions and report information on their outcomes. You will create the code supporting this task in **tournament.py** and **config.json**.

Task 4 – Reflecting on Your Work

Following completion of Tasks 1-3, you will respond to 5 short prompts to reflect upon how the module content was used in this coursework and the most challenging aspect of completing these tasks.

In the above tasks, some sub-tasks will ask you to provide a **written explanation** of the justification behind your coding choices. Within the notebook, code and written

responses should be presented in a set of well-formatted code and Markdown cells in the prompted locations. This work will require the production and submission of additional files; details about these files and how they should be submitted are provided in the notebook and the Assignment Submission Requirements.

Project Directory and Data Description

The materials needed to complete this assessment are available in a single **CS2PPNU_CW1.zip** file on the CS2PP Blackboard space, under the **Assessment** heading, in the **Coursework 1 of 2: Basics of Python** item. This is outlined below.

The first task relies on a file consisting of tab-separated integers. The second and third tasks rely on a file consisting of comma-separated values (CSV) with a header that briefly describes each column. These files will be used to work through the prompts in **CS2PPNU_CW1.ipynb** that guide manipulation of the data.

```
CS2PPNU_CW1.zip
├── data/
│   ├── dolphins.tsv
│   └── cardata.csv
├── images/
│   └── …a few image files…
├── CS2PPNU_CW1_Individual.pdf
└── CS2PP_CW1.ipynb
```

Bottlenose Dolphin Social Network Data: **dolphins.tsv**

This dataset contains a representation of a social network dataset where dolphins have links between them if they frequently associated with one another. Each line contains 2 integers separated by a tab character. Each value represents an individual dolphin, and each line represents a connection between the listed pair.

Source: http://konect.cc/networks/dolphins/

Car Features and MSRP Data: **cardata.csv**

This dataset includes car features such as make, model, year, and engine type, as scraped from Edmunds and Twitter. It is often used to develop models to predict car prices based on their other characteristics.

Source: https://www.kaggle.com/datasets/CooperUnion/cardataset

Each **row** corresponds to a single kind of vehicle.

The **columns** correspond to:

Make	Car maker
Model	Car model

Year	Car year (Marketing)
Engine Fuel Type	Type of engine fuel category
Engine HP	Engine horsepower (HP)
Engine Cylinders	Number of engine cylinders
Transmission Type	Type of transmission category
Driven_Wheels	Drive wheel category
Number of Doors	Number of doors
Market Category	Market category
Vehicle Size	Vehicle size category
Vehicle Style	Vehicle style category
highway MPG	Highway fuel efficiency in miles per gallon
city mpg	City fuel efficiency in miles per gallon
Popularity	Twitter-based popularity metric
MSRP	Manufacturer suggested retail price ($, USD)

3. Assignment submission requirements

"Front page" of the Submission

The following are **compulsory**. Please add these items to at the **top of your Jupyter notebook** in the provided Markdown cell.

Module Code:	
Assignment report Title:	
Student Number (e.g., 25098635):	
Actual hrs spent for the assignment:	
Which Artificial Intelligence tools used:	

Content of the required work

You must use Python (**version 3.11.7**) Jupyter Notebooks (via **JupyterLab version 4.0.11**). The **base** conda environment included with the Anaconda3 distribution used in this module (**2024.02**) will be sufficient for this work. No tools outside of the Python standard library and IPython/JupyterLab utilities should be used.

Your submission should take the form of a single archive file (based on the one downloaded for this project). Upload the .zip file to Gradescope via the Blackboard submission point. As you do, you should see that Gradescope automatically unzips the files as you submit. This is expected.

You will find the submission point on the module's Blackboard page under **Assessment**. The name of the archive and should be formatted with your 8-digit student ID number, the module code, and the tag "CW1" (e.g., **`12345678_CS2PP_CW1.zip`**).

The final content of your Blackboard submission should have, at minimum, the following structure and contents. Items in **orange** represent new files that you will produce or modify.

```
12345678_CS2PP_CW1.zip
├── data/
│   ├── dolphin.tsv
│   ├── cardata.csv
│   ├── cardata_modified.csv
│   └── config.json
├── CS2PPNU_CW1.ipynb [completed and fully executed]
├── CS2PPNU_CW1.html [exported copy of above]
├── network.py
├── processing.py
├── tournament.py
└── [any auxiliary modules you find necessary]
```

- You do not need to submit the image directory or this pdf.

Code Plagiarism

This coursework is expected to be the result of your own individual effort, **not that of other people or systems**. Do not work closely with others on this coursework. Do not employ pair programming techniques. Copying whole tutorials, scripts or images from external sources is not permitted. Any material you borrow from other sources **to build upon or to support your arguments** should be clearly referenced (use comments to reference element within Python scripts and code cells and supply formal references in a Markdown section at the end); otherwise, it will be treated as plagiarism, which may lead to investigation and subsequent action. Work **inspired by** module materials is permitted, but such material should not be used without significant modification. We understand that similar lines of code are inevitable, however, very similar lines of analysis and reporting spanning significant sections of the coursework will be investigated as potential academic misconduct (e.g., it is highly unlikely that you will independently choose the same model parameters and variable names).

4. Marking scheme

Element	Marks Available
Task 1: Structuring and Analysing Network Data • 2 marks for correct implementation relating to each of the 7 execution steps • 2 marks for each of the 3 written responses	**20**
Task 2: Data Processing • 1 mark for each of the 14 requested pieces of information reported correctly • 2 marks for a correctly constructed `cardata_modified.csv` • 4 marks for well-commented and structured code	**20**
Task 3: The `Tournament` Class • 30 marks for the definition of the `Tournament` class (detailed marking listed in the notebook) • 5 marks for execution of the `Tournament` class and win record display • 5 marks for the comparison of two `Tournament` instance executions • 10 marks for the correct, functional implementation of the `Tournament_optimised` class and written comments	**50**
Task 4: Reflecting on Your Work • 2 marks for each of the 5 requested responses	**10**
Coursework 1 Total	**100**

Refer to the notebook for additional information about requirements and a more granular listing of marks.