|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task:** | | **AT1** | | |
| **Task Title:** | | **Portfolio [Part 1]** | | |
| **Task Code:** | | **AT1-POR-Part-1** | | |
|  | | **STUDENT TO COMPLETE DETAILS BELOW** | | |
| **Lecturer name** | | **Murray Hay** | | |
| **Student name** | |  | | |
| **Student ID number** | |  | | |
| **Telephone contact number** | |  | | |
| **Email** | |  | | |
| **Acknowledgement** | | By completing and submitting this signed form to my lecturer, I am stating that:   1. The attached submission is completely my own work 2. I have correctly cited all sources of information used in this work (if required) 3. I have kept a copy of this assessment (where practicable) 4. I understand a copy of my assessment will be kept by the NMTAFE for their records   I understand my assessment may be selected for use in the NMTAFE’s validation and audit process to ensure student assessment meets requirements | | |
| **Student Signature** | |  | | |
| **Date** | |  | | |
| Assessment type (): | | | | |
|  | Questioning (Oral/Written) | |  | Portfolio |
|  | Practical Demonstration | |  | Project |
|  | 3rd Party Report | |  | Other – Please Specify |

|  |
| --- |
| The base requirements this assessment task include:   * Web server, Python interpreter and Database server * IDE or editor for developing Python programs * Raspberry Pi with SenseHat or other IoT devices, like Arduino Uno or ESP32 * Access to Office 365 & Microsoft Word   Use of some of these items may not occur in this part of the assessment task. |
| Assessment Due Refer to Blackboard for most accurate dates, which may alter due to unforeseen circumstances.  We also will endeavour to update these document(s) at the same time. |
| Instructions Follow the steps listed in this assessment item.  Submission of the documentation, code, and associated items is at the end of each part of the portfolio.  Each part of the portfolio has a deadline for submission.  It is advantageous to you to attempt to meet the deadline provided. |
| Important If you are using a different configuration of tools and equipment for this assessment item, then assistance in this and subsequent parts of the portfolio to ensure the systems work correctly will be limited. |
| Scenario *You have been given a task to create a program with a tight deadline and no internet access.  You are required to create a Python class representing a deck of cards and demonstrating OOP principles through a card game of Snap.* |
| General Instructions Complete all answers within this document. Follow each step’s instructions carefully, making sure that as you answer each question, or complete each step as described.  Any images must be clear and easily read. If they are from an external source, please note where. |
| Answering Questions When a step includes a question, you must attempt to answer it.  There is a minimum and maximum number of words to use for each answer.  If a step has more than one question, these maxima and minima are a total for all the questions in that specific step.  Make sure all answers are complete sentences unless otherwise indicated. |
| Sources of Information In industry, it is good practice to keep track of where information was obtained. This is especially true if it is a written document, or even code.  If you answer any questions using information from web sites, please include the site name and URL (Web site address) after the answer. Likewise, include the title and author for books and magazine articles. For example:   * RS Electronics Ltd: <https://au.rs-online.com/> * Slack API Documentation, Users List Method: <https://api.slack.com/methods/users.list> |
| Code Storage We advise that you create a GIT repository on GitHub and use this to store a copy of your work.  You may also use OneDrive within your college Office365 to store a backup of your code or keep a copy on a USB thumb drive. |
| How to submit this assessment.  1. Once you have finished the program, demonstrate it to the Lecturer. 2. Upload you code to the blackboard submission. |

|  |  |
| --- | --- |
| H01 | **Create a Deck of cards in python using Oop.**  In this assessment, you will be creating a Python class that represents a standard deck of playing cards. The class will demonstrate the principles of Object-Oriented Programming (OOP) by incorporating methods for shuffling, dealing, and playing a card game of Snap.  Deck of Cards Class:  The class should be named DeckOfCards.  It should have a constructor method that initializes the deck of cards with all 52 playing cards (no Jokers) and stores them in a data structure of your choice.  Implement a method named shuffle that shuffles the deck of cards randomly.  Create a method named deal that takes the number of players as an argument and returns a list of cards, distributing them equally among the players. If there are not enough cards for each player to get an equal share, simply distribute the cards as much as possible.  Write a method named display\_deck that prints the current contents of the deck, allowing you to visualize the order of the cards.  Snap Game:  Define a class named Player that inherits from the DeckOfCards class.  In the Player class, create a method called snap that checks if the top card of the player's hand matches the top card of the central pile (the pile where players "snap" when the cards match). If they match, the player wins that round and collects all the cards from the central pile. If not, the game continues.  Implement a method in the Player class to add the winning cards to the player's hand after a successful Snap.  Create a method in the Player class to check if the player has run out of cards in their hand, indicating the end of the game.  Implement the Snap game logic in your main program by simulating a game with multiple players (at least 2) taking turns to call Snap.  Assessment Criteria:   * Correct implementation of the DeckOfCards class with the appropriate attributes and methods. * Successful inheritance of the Player class from DeckOfCards. * Proper implementation of the shuffle method to randomize the deck. * Accurate implementation of the deal method to distribute cards among players. * Correct implementation of the snap method to determine if a player's card matches the central pile card. * Proper handling of winning cards and updating the player's hand in the Player class. * Accurate implementation of the game logic to determine when the game ends. * Appropriate usage of OOP principles such as encapsulation, inheritance, and polymorphism.   Remember to write clean and well-documented code. The program should demonstrate your understanding of OOP concepts and the logic behind the card game of Snap. You will be evaluated based on the completeness and correctness of your implementation, as well as the adherence to good programming practices.  Good luck with your assessment! If you have any questions or need further clarifications, feel free to ask. |

# Appendix A: Code Style Guidelines

Your code will follow the PEP 8 standard.

Readability Counts  
- Zen of Python

Explicit is better than implicit.  
- Zen of Python

Other code standards available in the Presentation, “Python Coding Standards for North Metropolitan TAFE”.

# Appendix B: Template Instructions

Some basic instructions on using this template.

These instructions include:

* Adding references, and
* What to do **before** submitting the assessment.

## Updating Table of Contents

When this document contains a table of contents, update it using these steps:

1. Go to the page that has the table of contents.
2. Click on the Table of Contents heading.
3. At the top click “Update Table”
4. When it asks, click update the whole table.