|  |  |
| --- | --- |
| **Course** | **TNABT Software Engineering** |
| **Unit Code** | **ICTPRG302** |
| **Unit Title** | **Apply Introductory Programming Techniques** |
|  | |
| **Assessment Task Title** | **Assessment 2 – Skills** |
| **Assessment Type** | **Practical** |

## Overview

The assessment provides the opportunity for you to demonstrate the following skills and knowledge:

* Design and build a simple application to specifications
* Develop algorithms
* Apply programming language syntax, sequence, selection and iteration constructs
* Test and debug a simple application
* Develop design specifications

## Instructions:

This assessment is to be completed in your own time. Time may also be made available for completing the assessment during class sessions.

You are required to enter your responses in the spaces provided in this assessment document, completed assigned activities and, follow instructions to upload resources

To achieve a ‘satisfactory’ result for this assessment you must complete all tasks and be deemed competent in all tasks by your assessor. In the event that you receive an unsatisfactory result, you will be required to review feedback from your assessor and then resubmit the assessment after making required corrections.

You will have one opportunity for resubmission. If your second assessment attempt is ‘not yet satisfactory’ you must contact your teacher or assessor to discuss how to proceed.

All responses must be your own work.

## To be submitted:

* This completed assessment document titled - Assessment 2 – Skills
* Panopto video of IDE debugger use
* Zipped up REMOTE Github repository (.zip)

Upload all documents by the due date to the drop box for ICTPRG302 Assessment 2 on VU Collaborate.

## Assessment scenario – Password Manager

You work as a programmer for a small digital development agency “Apps2U”. Apps2U have asked you to develop an application on behalf of their customer “DigiCore”.   
  
DigiCore have requested a simple password manager because their staff are resorting to insecure practices to record login credentials such as writing them down on sticky notes. The password manager will be used to store and retrieve credentials for websites and other login services used by employees.

## **Student Name: Student ID Number:**

## Task 1.1 Clarify task with required personnel

Discuss the programming specification at task 1.4 with your instructor who is acting as your work supervisor. This is your opportunity to ask any questions to ensure you fully understand what the application should do. Provide brief notes or dot points of the discussion that show your attempts to clarify and understanding the programming specification. *Note that your instructor may work with you individually, in small groups or as one large group for this task*

|  |
| --- |
| * Need option to add username, password and URL/program * Need option to view above and exit * Information to be stored in txt file * Ensure no errors possible with entering and retrieving * New records are to be appended, no updates or removing existing entries * System to cope if one field is empty * File to be encryted and unencrypted following ensuring program is running |

## Task 1.2 Identify design specifications relevant to the task

Complete the following design specification

|  |
| --- |
| **Design specification** |
| Overview of the project |
| Who will use the application |
| Why is the application needed |
| What is the benefit of using the application for the client |
| What are the features and functions of the application? *What is it that the application does overall, what are its individual functions, how will a user navigate the application, what are the inputs, what are the outputs, are there any special features?* |
| Appearance  *How will the user interface look? How will any outputs from the program look?* |
| When is the application due for completion (assessment due date) |
| What questions do you have about the project *What do you not understand about the design of the application?* |
| Summary of this design specification *Describe this entire design specification concisely* |

## Task 1.3 Algorithm Design

Produce a design of your intended program using pseudo-code with explanatory comments. Use the program specification checklist in task 1.4 below to guide the design of your algorithm/s

Contact your instructor for constructive and corrective feedback when complete. Make any corrections to the algorithm design and provide brief notes or bullet points of the changes.

*Note that your instructor may choose to provide feedback individually, in small groups or as a large group.*

|  |
| --- |
| *Enter your pseudo code:*  *Enter brief notes or bullet points of any changes made after supervisor review:* |

## Task 1.4 Create a simple application to specifications

Translate your pseudo code into a Python3 script that adheres to the code layout, white space and comments recommendations of the PEP 8 Style guide for Python code. You may use the provided code examples in the appendices, as part of your script.  
  
Use the program specification below as well as the coding checklist in the appendices to ensure that the application contains all required elements.

|  |  |
| --- | --- |
| **Program specification and checklist**  Your code will be written to the following specification | |
|  | Done |
| Include an options menu for the user allowing them to carry out the following actions   * Add stored credentials (username, password and URL/resource) * View stored credentials * Exit the program |  |
| Return to the menu after each action has completed |  |
| Create a text file for credential storage if a text file does not already exist |  |
| Append new records to the text file without overwriting previous entries |  |
| Display the text file contents in a visually presentable way including spacing and headings |  |
| Handle any input from the user and carry out actions, without errors |  |
| Include embedded explanatory comments (#) to clarify the meaning of the code |  |
| Provide simple rot3 encryption on all written data and, decryption on read data |  |

Record the development of your script using the GitHub development platform. Add your instructor as a collaborator to your repository. Push and pull changes frequently to keep the local and remote repositories synchronised and so as to keep your instructor informed of your progress. Your instructor, acting as your supervisor, will check the repository and make comments during code development. You are to review your code based on those comments and when committing reviewed code using GitHub Desktop, state the words “acted on review” in the summary field of GitHub desktop.  
*Do not produce code with Window elements or complexity beyond what is required.*

## Task 1.5 Debug

Debug your application

* Use an IDE and its inbuilt debugger to debug your script
* Provide a brief Panopto video (3 minutes maximum) with **vocal commentary** that shows the IDE debugger in action, including a variables contents changing, stopping at a breakpoint, stepping over a function, stepping into a function and, identification of the cause of an error. Provide the hyperlink to the video below. *Be brief by not including any more content than is required for the task*

|  |
| --- |
| Upload the Panopto video to the assessment dropbox for this unit so a permanent record of your video is created for VU. Also, for easy access, enter the hyperlink to the Panopto video here and ensure that you have enabled READ access for your instructor |

* Provide a list of three semantic errors you have encountered and how you rectified them.   
  *Note that syntax errors are* ***NOT*** *acceptable. The errors* ***MUST*** *be caused through incorrect logic*

|  |  |
| --- | --- |
| **Error** | **Rectification** |
| *e.g. Code to view the text file is not being ran* | *The menu option was capitalised but user entry was in lower case* |
|  |  |
|  |  |

## Task 1.6 Test

Develop a complete set of test cases to confirm the code meets the program specifications. Record the Test cases below.

|  |  |  |
| --- | --- | --- |
| **Description:  *(what is being tested)*** | **Expected response or output** | **OK** |
| *e.g Add a new record for the first time* | *Text file is created; username, password and URL are add to the text file. Password is encoded.* | *Yes* |
| *Add rows to this table as needed* |  |  |

## Task 1.7 Gain feedback, review and finalise

Seek feedback from your supervisor and review the code.

* Contact your instructor, acting as your supervisor, when your application is complete to confirm that your application meets the initial design specifications and program specifications. Document this discussion via bullet points or brief notes, make any required adjustments to the code and, obtain final verbal sign-off.
* Make one final GitHub commit and included the words "supervisor sign-off" as a comment in your .py file and the words "supervisor sign-off" in the summary field of GitHub Desktop. Ensure you have pushed all commits to remote Github, download your REMOTE GitHub repository in.zip format and submit as part of your assessment.

|  |
| --- |
| *Enter notes of*   * The discussion with your supervisor confirming that your application meets the initial design specifications and program specifications. |

**Appendix A** – Coding Checklist

|  |  |
| --- | --- |
| **Coding Checklist**  Your code must include following criteria. *The checklist is given to* ***ensure*** *all aspects of the assessment are covered. Your assessor will confirm that your code includes the criteria when marking your submission* | |
| Criteria | Tick when complete |
| At least one of each of the following:   * Global variable * Local variable |  |
| At least two library functions  *(internal or external)* |  |
| At least one self-created function |  |
| Clarifying comments |  |
| A data structure  (i.e list, dictionary, tuple or set) |  |
| Manipulation of strings |  |

**Appendix B – Provided code**

You may use the following premade code in your script

**Menu system**

# Give the user some context.

print("\nThis program…………………………")

# Set an initial value for choice other than the value for 'quit'.

choice = ''

# Start a loop that runs until the user enters the value for 'quit'.

while choice != 'q':

    # Give all the choices in a series of print statements.

    print("\n[1] Enter 1 to create an encryption key.")

    print("[2] Enter 2 to …….")

    print("[3] Enter 3 to……")

    print("[q] Enter q to quit.")

    # Ask for the user's choice.

    choice = input("\nMake your choice ")

    # Respond to the user's choice.

    if choice == '1':

        print("\nEnter a name for the encryption key\n")

    elif choice == '2':

        print("\nEnter …….\n")

    elif choice == '3':

        print("\nEnter ……\n")

    elif choice == 'q':

        print("\nExiting the menu\n")

    else:

        print("\nInvalid option, please try again.\n")

# Print a message that we are all finished.

print("Program exit.")

**ROT3 encryption**

clearText = "myPassword"

charSet="0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz`~!@#$%^&\*()\_-=|\}]{[\"':;?/>.<, "

encText = "".join([charSet[(charSet.find(c)+3)%95] for c in clearText])

print(encText)