Internal Practice Assessment Resource

Digital Technologies Level 2

This resource supports practice assessment against

[**AS91896**](https://www.nzqa.govt.nz/nqfdocs/ncea-resource/achievements/2019/as91896.pdf) V1 - Use advanced programming techniques to develop a computer program (6 credits)

[**AS91897**](https://www.nzqa.govt.nz/nqfdocs/ncea-resource/achievements/2019/as91897.pdf)V1 - Use advanced processes to develop a digital technologies outcome (6 credits)

**Instructions:**

* Read the entire assessment task and assessment schedules for programming and processes
* Spend four hours in class and 2 hours out of class per week
* Open this document in Desktop version, do not save to any other location as it saves on SharePoint
* Click on the [link to](https://classroom.github.com/a/2_ssJKYg) [accept invitation](https://classroom.github.com/a/2_ssJKYg) to Programming\_Processes\_Assessment on Github. At the end of each session you must update Task 2 and 3 and upload py file to your repository on GitHub
* You must comply with the Occupational Health and Safety in Employment Act 1992, and their subsequent amendments.
* You must ensure you have not breached the Copyright Act
* Complete ALL tasks, attach your final py file with code commenting to Teams and Click ‘Turn In’

**Authenticity Statement:** This is an open book assessment, and you can go to various sources for guidance—for example, OneNote, CodeAvengers. Acknowledge sources of learning, including any help from teacher, friends, and family before or during this assessment. If work is non-authentic in any aspect, no credit will be given regardless of the quality. This will apply to **ALL** pupils involved in the misconduct. There will be no opportunities offered for further assessment in this standard for the current year.

Tick the box to acknowledge that you read and understood the Authenticity Statement

**Resubmission:** A **resubmission** may be offered when a student can gain an Achieved grade, if they correct minor errors or omissions in their work in a short period of time. There is no resubmission for Merit or Excellence grades. Therefore, it is essential to check assessment schedules for both standards regularly.

# Overview/Kupu Arataki

This assessment activity requires you to **plan**, **trial**, **test** and **develop** a computer program using advanced programming techniques. You will use a development process to help you make informed decisions throughout the coding, testing and trialing of your program and show ongoing testing and refinement to improve the functionality and quality of your program.

You will be assessed on how effectively you plan your development, decompose the outcome into smaller components, test and refine your program so that it is a high-quality response to the task (e.g. well-structured, logical, flexible, robust and comprehensively tested).

When planning and developing your program, you must ensure your program:

* uses variables storing at least two types of data (e.g. numeric, text, Boolean)
* uses **sequence** (commands), **selection** (conditionals) and **iteration** (loops) control structures
* takes input from a user, sensor(s), or other external source(s)
* produces output

**AND** includes **two or more** advanced programming techniques, such as planning and writing code that:

* defines and manipulates multidimensional data in collections (e.g. lists, dictionaries)
* creates methods, functions, or procedures that use parameters and return values
* responds to events generated by a graphical user interface (GUI)
* requires non-basic string manipulation, e.g. slice
* uses the functionality of additional non-core libraries.

# Tasks/Hei Mahi

## Brief 1 – Password Manager Program

People sign up for various websites and often it is hard to remember too many password.

You are to create a program in an iterative manner for the users to store the account and its password. The aim is to create just one single program.

You must create a minimum of 3 versions and the last version must be a GUI Version.

## Specifications

* Introduction to the outcome you are developing include purpose, end users etc.
* You must use either dictionary or list to store items or user accounts
* Use conditions to trap errors.
* Make the program user friendly.
* You must develop at least 3 versions and the last version or the last two versions should be of GUI
* Test your program thoroughly for expected, boundary and invalid cases.
* Use table to record your test results.

# Before you start your program, write a list of your requirements such as variables and what datatypes you require, what means you have to store things, what functions and why you are using them and what widgets are you using for GUI

|  |  |  |
| --- | --- | --- |
| Requirements | Data types if any | Purpose |
| Password variable | String | To save password |
| Account variable | String | To save the account linked to password saved |
| Choice variable | Integer | For the user to input the action they want to program to do |
| Entry widget | String | For the user to enter information used to save or retrieve information |
| Combobox widget |  | So the user can select a specific data to manpulate. |
| Button variable |  | For the user to give inputs to run a command in the program. |
| Label widget |  | For the program to relay information back to the user. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Task 1 My Project

|  |  |
| --- | --- |
| Name of app |  |
| Project management | Give a shared link to Trello. Add screen shots of project management to developmental logs. |
| Version control system | Give a link to GitHub. Add screen shot showing several versions on Github  [OlonYong/PasswordManager (github.com)](https://github.com/OlonYong/PasswordManager) |
| Flowchart/Pseudocode | Give a shared link to draw.io or similar. Add screen shots of plan in Task 5 |

# Task 2 Developmental Logs (Processes)

|  |  |
| --- | --- |
| Date | Log what you’ve done each day on the project. Alternatively, you may use any screen casting software to record your desktop and voice explaining what you would otherwise write in these tables. This is important even if you just want a grade of **Achieved**. **Merit** and **Excellence** require that you show how your project developed throughout including but not limited to use of design thinking steps, independent research, issues you encountered and how you overcame them, ongoing testing procedures. Provide clear evidence of ongoing evaluation, and feedback from self and potential users for improvement.  *Add rows as needed.* |
| 18/03/23 | Program:    Created a basic outline of the password managing program. Has the ability to add a password and retrieve the password. Passwords only save for this session so in iteration 2 the passwords should move between sessions. Iteration two should have more functionality and improve aesthetics. |
| 19/03/23 | Program output:    - external file.  From iteration 1, I decided to store passwords onto an external text file so that passwords can be retrieved even after the session has ended. Two new functions have also been added although not yet implemented, the ability to delete stored passwords and to view all passwords. |
| 20/03/23 | Program output:    Added functionality to delete password and view all saved password options. Added some more aesthetics changes to make the program look cleaner. The second iteration is finished. |
| 22/03/23 | I have written a basic outline of the password manager in GUI. Next I will need to add functionality to all the buttons. |
| 23/03/23 | I have added functionality to adding a new login as well as being able to retrieve the login. Tomorrow, I intend to implement the other two functions. |
| 24/03/23 | Today I have added the functionality for the rest of the functions being view all logins and remove login. I also fixed some bugs regarding the add login function. |
| 25/03/23 | I changed the remove login and view all logins from entry to combobox for ease of access so users do not need to remember all the logins. A new login screen was implemented for additional security as well as a button to set the login information. Iteration 3 has been completed. |

# Task 3 Testing Log

|  |  |
| --- | --- |
| **Ongoing testing is collected as evidence for programming and processes, therefore, update this table during the development. This is different from Test cases. Your testing log must include description of component tested, screen shot of result, what are the errors and how you have fixed them.**  *Add rows as needed.* | |
| **What was the Error (Screenshot)** | **How I fixed it (Screenshot)** |
| The function wouldn’t accept the inputs for account and password. | I had accidently left out the arguments in the function. It didn’t work because I forgot to add it. |
| Each time a new password was added it would override and delete the old password. | The issue was the program cleared the text file before writing information. I had assumed “w” would mean write but it meant truncate or create new file instead. I resolved this issue by changing the file from “w” to “a” meaning open to writing. The program can now read and see if there is pre-existing information and will write new information onto a new line instead. |
| The function should clear the middle line of the text file but it doesn’t. | The previous code doesn’t clear the line in the text file. To solve this issue, I rewrote the code to where all the code was read and saved down. It would then rewrite each line back into the text file if it didn’t match the target account. The account is successfully deleted as it will not be rewritten by the program. |
| As the program scans the text file for a matching account name, every line it did not find the account in the text file it would print a message. | Created a “found” variable to track whether the account had been found or not. The “not found” message will now only show at the end of the loop if the account was not found002E |
| A frame error occurs each time the code is run. | The function for frame1 and the actual frame1 were named the same thing. Due to this, they seemed to interfere with each other causing an error. By changing the name of the function to frames1, they no longer interfere with each other allowing the code to work. |
|  |  |
|  |  |
|  |  |

# Task 4 Test cases (Programming)

Test cases describe different scenarios that occur when the user inputs data into the system you build. There are three basic types – expected, boundary and invalid. To call it comprehensive testing, you must plan several test cases in the first four columns before the development and complete the fifth column before you hand in:

* Expected: These are values that you would expect the user to enter in normal circumstances. They fall within the expected range of input. For example: age = 5 or 18 or 50 or any number from 0-100
* Boundary: These are values that are just at, below and above the boundary values
* that are just on the edge of what you would expect the user to enter in normal circumstances. If the minimum age of the user is 13. Then age = 12 or 13 or 14 are boundary values.
* Exceptional (Invalid): These are values that you wouldn’t expect the user to enter in normal circumstances. They fall outside the expected range of input. For example: age = 150 or ten or -15 or 13.5 or 1 8.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| User input (*A short explanation)* | Test Case  (*What would the user input?)* | Input Type  *(expected, boundary or invalid)* | Output  *(How would your system respond?)* | Actual Output  *(Screenshot)* |
| Enter password: | Something | Expected | Program asks for password | Enter password: |
| The user should input their password | The password to save | Expected is any characters. | The program saves the password for future use. |  |
| Choice input | The number of the preferred action | The expected input is an integer between 1 and 5. Boundary values are 1 and 5. | An invalid message is shown before showing the prompt again until a valid entry |  |

Task 5 Flowchart or Pseudo code of your program

Add screenshots of draw.io for main routine and each function. They must be clear and readable. There must be several iterations of your flowchart or Pseudo code (show teacher a sample of your pseudo code for approval) during your development.

Iteration 1:

Ask user input

If input = 1

Ask input account name

Ask for password input

Add password and account name to list

loop

If input = 2

ask for account input

retrieve matching password from saved list

if matching password found: print password

if no matching password found: print password not found

loop

if input = 3

end program.

Iteration 2:

Ask user for input

If input = 1

Run newpassword function

Ask user for account

Ask user for password

Open external text file

Save accounts as dictionary on text list.

Loop

If input = 2

Run getpassword function

Ask user for account

Open external text file

Retrieve matching password for inputted account

If password is found to match input:

Print password

If password is not found to match:

Print password not found

Loop

If input = 3

Run deletepassword function

Ask for user account

Open external text file

If account in text file:

Clear line

If account not in text file

Continue

Loop

If input = 4

Run saved function

Open external text file

Print all contents of text file

Loop

If input = 5:

Exit program

Iteration 3:

When login button pressed

If entries are filled:

Check if login information matches saved

If true:

Enter frame 2

Ask user for input

If addlogin button pressed:

Run newpassword function

Get account entry

Get password entry

Open external text file

Save accounts as dictionary on text list.

If retrieve password button pressed:

Run getpassword function

Get entry value

Open external text file

Retrieve matching password for inputted account

If password is found to match input:

show password

If password is not found to match:

show password not found

If remove login button pressed:

Run deletepassword function

Get combobox entry

Open external text file

If account in text file:

Clear line

If reset login button pressed

Run clearlogin function

Open external text file

Clear contents

**Task 6:** Explain each iteration in your own words. (What are the developmental features you added in each version.)

Iteration 1

For iteration 1, it is the most basic framework for the password manager. The iteration asks users for their account and password before saving into a list for retrieval in future. The program prompts the user with a numbered list of possible actions. This can be improved in iteration 2 to where passwords are saved even when a session is ended. Aesthetic changes can also be made for more user friendly experience as well as more functionality like the ability to view existing logins and delete logins.

Iteration 2

For iteration 2, it expands upon iteration 1 implementing more functionality as well as having better aesthetics. The program prompts users with five possible options asking for a numbered input. The second iteration has improved from the first iteration saving passwords into an external text file so that passwords can be saved indefinitely rather than resetting after each session. A third iteration can improve the program by adding additional security to make sure anyone cant access all passwords stored.

Iteration 3

In iteration 3 I have gone for a GUI based program rather than a command text based program. By implementing this, navigating and using the program becomes much more intuitive for the user. The third program has the same features as the second iteration functioning more or less the same after adaptations to GUI. The most significant change in iteration three is an initial login screen to prevent others from accessing sensitive data. Without the correct login the user is not able to view any saved logins thus protecting the privacy of the data. A new function was also added which resets the login information in case the user wants to change the password.

Task 7 Explanation of relevant implications

For **achievement**, choose *at least* **three** implications from the list in OneNote. Define what is the relevant implication , describe why it relevant for this project and explain how you addressed those implications that relate to your project. For **merit/excellence** you will need to show evidence of at least five of these relevant implications.

|  |  |  |
| --- | --- | --- |
| Define what the implication is. | Describe why it is relevant | Explain with annotated screen shots of how you addressed this implication |
| Functionality | The program should have enough functionality to be a useful addition to the users lives. To achieve this, the program should function smoothly and have enough feature and functionality to make it useful or worthwhile to the user. | I made sure to make the program as intuitive for the user as possible. Many functions were also added to ensure anything that the user wanted the program to do it can.    As can be seen the program has many functions allowing the user to perform a variety of actions. The program also runs smoothly without any bugs. |
| Usability | The program should be accessible and usable by anyone. It should be intuitive and user friendly without any prior knowledge. The program should prevent users from accidently making any errors. | As can be seen the colours of text contrast the background allowing even visually challenged users to easily use the program. After thorough testing no errors were found. All aspects of the program are self explanatory or obvious to the user. Clear instructions on how to operate the program are given as shown in the screenshot. |
| Ethical | The program should not offend any group or person. This means there should not be any racist or sexist remarks/comments. It should also not plagiarize from other sources without permission. | All text in the program do not have any racist or sexist comments that would offend any group or individual. I made sure not to plagiarize any code from the internet but instead took inspirations and adapted it to meet fair use laws. None of my code was directly taken from any specific source. |
| Data Security | The program should not leak or expose sensitive information to unintended audiences. A password manager program should protect the data stored by the user. User privacy and personal information can be threatened if the program leaked the data. | A login was added for security when using the program. The user can set a username and password to prevent anyone else from viewing saved passwords. When typing especially sensitive information like passwords they are hashed out for additional security. All saved logins are stored locally so hackers are unable to obtain data remotely though the program. |
| Aesthetics | A program should have appealing and clean aesthetics to improve user experience. | The program was adapted into a GUI form rather than text based for better visibility and better aesthetics. The program should have a clean look where elements are spaced out without clutter so that the user is more able to focus on a specific element. As can be seen on the screenshot each element is adequately spaced out from each other. Spacing should also have consitancy for better looks. Contrasting colours should be sued for better looks. |
|  |  |  |

Task 8 Reflective summary

Present a summary of how the information from planning, project management and design thinking processes, relevant implication, and testing and trialing assisted you to develop a high-quality outcome. This may include annotated screen shots of Trello, program and iterative process that guided your development process and helped you complete all components and manage your time effectively.

I had set myself a goal to work for at least 30 minutes each day on the program. This made sure I had no stress and gave me adequate time to produce a high quality outcome. Trello was not particularly useful to me as I had already assigned 30 minutes each day to work on the project and it could be worked on at any time. Any time I felt most productive I worked on the project leading to an overall higher quality work. Adequate planning of the program prior to coding was useful to lay out everything I wanted to do so I could implement them smoothly without many compatibility bugs when I tried adding new stuff to preexisting code. It also help smooth out the development process knowing exactly what I was doing at all times.

Task 9 Source of learning

Acknowledge sources of learning, including any help from teacher, friends, and family before or during this assessment.

YouTube tutorials were a useful source of learning for this assignment to learn to implement external text files. W3Schools and python.org were also common website to learn new functions or to clarify how to correctly implement each function. Friends were asked to provide constructive feedback or to fix some bugs I was not able to fix myself.

Links:  
<https://www.w3schools.com/python>

<https://docs.python.org/3/tutorial/>