A red and blue text on a black background

Description automatically generated

## **Assignment Title Page**

|  |  |
| --- | --- |
| **STUDENT NAME:** | Goh Qing Feng |
| **P.I. NO.:** | Q2181339 |
| **TITLE:** | TMA01 |
| **COURSE CODE:** | ANL252 |
| **TG:** | T03 |
| **SUBMISSION DATE:** | 15/09/2023 |

**Plagiarism and Collusion**

**Plagiarism** is the act of using or passing as one’s own, the ideas or writings of another without acknowledging or crediting the source from which the ideas are taken from.

**Collusion** is the act of submitting any academic work (including assignment, project or report) that was completed by another person and pass these work off as one’s own.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
| **Declaration** | | | | |
|  | | | | |
| I declare that this assignment is my own work, unless otherwise acknowledge or credited by appropriate referencing. I have read and abide by the SUSS Honour Code and I am aware of the penalties associated with plagiarism and collusion listed in the Student Handbook. | | | | |
|  | | | | |
| Initial: | G.Q.F |  | Date: | 15/09/2023 |
|  | | | | |

Question 1a)

Plagiarism in coding happens because of the following:

* Easy access to online coding resources allowing people to effortlessly copy and paste the codes. Examples include copying code snippets on Stack Overflow or ChatGPT.
* Laziness of programmers as they are tempted to plagiarise codes as a shortcut.
* Insufficient knowledge or skill to solve the task.
* Complex and challenging tasks result in individuals copying others’ work as they are unable to comprehend the tasks.
* The collaborative nature of the programming profession. As programmers typically work as a team, it may blur the lines of collaboration and collusion. (Education Technology Solutions, 2021; Lee, 2020).
* Lack of time due to tight deadlines resulting in programmers plagiarising to save time.
* Competition among peers may influence individuals to plagiarize to have efficient and effective codes.

Plagiarism can be avoided by:

* Managing time effectively to have ample time for programmers to comprehend the task and create original and unique code from scratch.
* Practicing and developing coding skills to tackle the task independently.
* Implementing software that reviews for similarity to detect plagiarism (Lee, 2020).
* Being transparent and providing proper attribution as required.
* Seeking assistance on how to comprehend or tackle the task, rather than seeking answers.

Word count: 198

Question 1b)

import random

def shuffle(string):

tempList = list(string)

random.shuffle(tempList)

return ''.join(tempList)

uppercaseLetter1=chr(random.randint(65,90))

uppercaseLetter2=chr(random.randint(65,90))

lowercaseLetter1=chr(random.randint(97,122))

lowercaseLetter2=chr(random.randint(97,122))

digit1=chr(random.randint(48,57))

digit2=chr(random.randint(48,57))

punctuation\_range = [(33, 47), (58, 64), (91, 96), (123, 126)]

selected\_range = random.choice(punctuation\_range)

punctuationSign1=chr(random.randint(selected\_range[0],selected\_range[1]))

punctuationSign2=chr(random.randint(selected\_range[0],selected\_range[1]))

password = uppercaseLetter1 + uppercaseLetter2 + lowercaseLetter1 + lowercaseLetter2 + digit1 + digit2 + punctuationSign1 + punctuationSign2

password = shuffle(password)

print(password)

(Retrieved from <https://www.101computing.net/random-password-generator/> on 10/09/2023)

The Python code starts by importing the ‘random’ module to generate a random password. Then, a ‘shuffle’ function is created to rearrange a string by converting it to a list using the list() function and then randomly reorganizing the items with the random.shuffle() method. For example, the string ‘hello’ is converted into a list [‘h’, ‘e’, ‘l’, ‘l’, ‘o’] and may be randomly reorganized into ['l', 'h', 'l', 'o', 'e']. Next, the ‘’.join() method is used to concatenate the items in the list together without any separator and the “shuffle” function will return the concatenated value.

Then, the code will randomly generate two upper-case letters, two lower-case letters, two digits, and two punctuation signs using the chr() function, the decimal numbers in the ASCII code, and the random.randint() method. The random.randint() method returns a random integer selected from the specified range, while the chr() function returns the string representing a character according to the ASCII code. For example, random.randint(65,90) may return 65. Subsequently, chr(65) will return the string ‘A’.

Finally, the code will add all the randomly generated elements using the + operator, apply the ‘shuffle’ function, and generate a random password.

Word count: 197

Question 1c)

import random

import string

#Determine number of uppercase letters, lowercase letters, digits, and punctuations

char\_length = 2

#Empty string to store password generated from each iteration

final\_pw = ""

#Loop to generate characters for password

for i in range(char\_length):

uppercase\_letter = random.choice(string.ascii\_uppercase)

lowercase\_letter = random.choice(string.ascii\_lowercase)

digits = random.choice(string.digits)

punctuation = random.choice(string.punctuation)

final\_pw += uppercase\_letter + lowercase\_letter + digits + punctuation

#Shuffle generated password

pw\_list = list(final\_pw)

random.shuffle(pw\_list)

shuffled\_pw = ''.join(pw\_list)

print(shuffled\_pw)

"q8Tz7[D

The rationale behind rewriting the code is:

* The rewritten code allows us to avoid coding plagiarism while maintaining the core logic of the password generation. The rewritten code includes a for-loop function to generate random passwords and uses the ‘string’ module to generate characters.
* The rewritten code is more concise, readable, and easier to modify. The for-loop function would reduce the complexity of having to manually add the characters together to generate the password. The added ‘string’ module provides predefined character sets for uppercase letters, lowercase letters, digits, and punctuation signs. An example is string.punctuation, which contains all punctuations as a string. These predefined character sets will reduce the complexity of manual character range calculations as seen in the original code. The predefined character sets also allow for easy modification as compared to the original code, which required the conversion of the decimal numbers in the ASCII code to characters. Hence, the code is simplified, and the chance of errors is reduced.
* The new code also allows users to specify the password length using the 'char\_length' variable, giving them more flexibility to create secure passwords. The flexibility of the code is particularly important as each user has different needs. If a user wishes to have a more secure and longer password, he may do so by increasing the ‘char\_length’ variable. For example, by increasing the ‘char\_length’ variable to 3, it will randomly generate a password consisting of 3 upper-case letters, 3 lower-case letters, 3 digits, and 3 punctuation signs.
* The use of the 'random.shuffle()' method directly on the character list eliminates the need for a user-defined 'shuffle' function, which only appears once in the original code and adds unnecessary complexity. Overall, the rewritten code simplifies the process of generating passwords and reduces the chance of errors.

Word count: 296

Question 2)

products = ['laptop','mouse','webcam','keyboard','speaker']

#to define query and start the while-loop

query = 'yes'

#shopping list

updated\_items = []

print(f'We have a list of products here: {products}.')

while query == 'yes':

#prompt user to add item to the list

item = str(input("Hello! What do you want to buy? ")).strip().lower()

if item not in products:

#Only allow user to enter products in the list

print(f'{item} is not a valid product. Please try again.')

else:

while True:

try:

price\_of\_item = float(input(f"How much does {item} cost (in SGD)? "))

#Add item to the end of the shopping list

updated\_items.append({'item': item, 'price': price\_of\_item})

print(f'{item} of {price\_of\_item:.2f} has been added to your shopping list.')

#show updated shopping list

print('Your shopping list:')

for i, item in enumerate(updated\_items):

print(f'{i + 1}). {item["item"]} (SGD {item["price"]:.2f})')

#prompt user to continue

query = str(input("Would you like to continue? (yes/no) "))

break

except ValueError:

#Only allow user to enter valid price

print("Please enter a valid price")

Firstly, error handling and reporting mechanisms have been implemented to make the code more reliable. Previously, the code was case-sensitive and would stop running entirely if the user inputted an invalid product. This is an ineffective error-handling mechanism as it will require the user to key in their shopping list from scratch. Furthermore, the code will accept a non-integer for the price of the item while having no error reporting mechanism when the user inputs an invalid price. The improved case-insensitive code will ensure that it will continue to prompt the user to enter a valid input when receiving an invalid product and/or price, ensuring that the program continues to run.

Secondly, the code has improved in its maintainability by adding documentation to help the developers understand the code’s purpose and usage. This made the code more readable, allowing developers to modify and maintain it more easily.

Lastly, the overall quality of the code has improved as the output is more readable and user-friendly. Previously, the shopping list was displayed as a list in a list without much information, such as “[['laptop', '100']]”. Now, it will provide a clearer display of the shopping list, such as “1). laptop (SGD 100.00)”.

Word count: 200

Bibliography

*Combatting Code Plagiarism in Computer Science | Codequiry*. 30 Dec 2018, <https://codequiry.com/code-plagiarism>.

Lee, Christine. *What Is Programming Plagiarism? Why Is It on the Rise?* 28 July 2020, <https://www.turnitin.com/blog/what-is-programming-plagiarism-why-is-it-on-the-rise>.

*Random Password Generator - 101 Computing*. 20 Feb. 2019, <https://www.101computing.net/random-password-generator/>.

Solutions, Education Technology. “Copying Code: Why Programming Plagiarism Is Becoming an Issue in Australian Institutions.” *Education Technology Solutions*, 7 July 2021, <https://educationtechnologysolutions.com/2021/07/copying-code-why-programming-plagiarism-is-becoming-an-issue-in-australian-institutions/>.

Stevens-Huffman, Leslie. “How to Avoid Plagiarism When Taking a Coding Assessment for a Job.” *Dice Insights*, 9 Nov. 2020, <https://www.dice.com/career-advice/how-to-avoid-plagiarism-when-taking-a-coding-assessment-for-a-job>.