

**ANL252**

**Python for Data Analytics**

**Tutor-Marked Assignment**

**July 2023 Presentation**

**Submitted by: Ng Xue Er Ariel (J2271307)**

**Submission Date: 15 September 2023**

**Question 1a**

Coding plagiarism happens due to the following reasons:

1. Easy availability of open-source codes - Large number of open-source software with code repositories and forums are available online e.g. Github and Stack Overflow (Lee, 2020). They are mostly free which provides an easy access for anyone to use the answers.
2. Time Pressure - A lot of practice is required when mastering a new programming language. When coupled with a tight assignment deadline, it creates a challenge for the students. Hence, in combination with the easy availability of codes online, struggling students are inclined to take short-cut solutions (Lee, 2021)

To avoid coding plagiarism, the following actions can be taken:

1. Citation - Cite the source to give credit to the original author or source of the code (LinkedIn, n.d.)
2. Modification - Modify code to your specific needs. Some examples shared by Linkedin (n.d.) includes “adding your own feature or functionality or improving its performance or quality” of the code.
3. Understanding - It refers to understanding the code to learn “how it works, why it works and what it does” (LinkedIn, n.d.). This helps to gain confidence in writing our own code without relying on other sources.

(199 words, including in-text citation)

**Question 1b**

Python code:

#Create a list of colours available, this stores for user's input later

Colour\_choice = ["black","white","pink"]

#Upon using the system, a welcome message is printed

print("Hello, welcome to SuperStore!")

#Creates empty list to store user input

Cart = []

#Enables user input for colour selection

#.lower() was used to convert user input into lower case to match the list

Item = input("Each customer is limited to pick 2 colours. Please indicate the colour of your choice:").lower()

#If-else condition below will check if user input is within the colour\_choice list defined above

#If user input is within the colour\_choice list, the user input is appended to the empty list defined above - Cart

if Item in Colour\_choice:

Cart.append(Item)

#As each customer can pick 2 colours, another query pops up to ask if customer would like to pick another colour

Query = input("Would you like to pick another colour? (Y or N)").upper()

if Query == 'Y' :

Item2 = input("Please indicate the colour of your choice:").lower()

#If customer enters "Y" and input another colour, it checks for (1) duplicate entry in initial selection, (2) colour availability, else error message will pop-up to get user to try again

if Item2 == Item:

print("Duplicate colour selected. Please try again.")

elif Item2 in Colour\_choice:

Cart.append(Item2)

print("Happy Shopping! You have selected to purchase the following colours:")

for x in Cart:

print(x)

else:

print("Colour not available. Please try again.")

#If customer enters "N" for initial query to pick another colour, print results displaying colours chosen in the initial selection

else:

print("Happy Shopping! You have selected to purchase the following colour:")

for x in Cart:

print(x)

#If user input is not within the colour\_choice list, an error message will be prompted to get user to try again

else:

print("Colour not available. Please try again.")

The python code above serves as a user interface designed for customers of ‘SuperStore’. It enables them to make colour selections for shirt purchasing in stores. First, the available colour options are defined within a python list named “Colour\_choices [ ]”. The python list was used to allow modification of the available colours if needed. In this scenario, the list of colours act as an inventory at the backend, that enables customer to verify the availability of their preferred shirt colour. With a restriction of 2 colour picks per customer; if-else condition was used to automate user input twice. In the event that the chosen colour is not available, the program displays an error message prompting user to try again. In contrast, if the chosen colour is available, the if-elif-else conditions was used to automate a series of steps: to first check for duplicate entries as compared to the initial selection and verify the colour availability once more. Finally, it prints the result for the user indicating the selected colours.

Bottom of Form

(170 words)

**Question 1c**

Modified python code from Q1b:

Colour\_choice = ["black","white","pink"]

print("Hello, welcome to SuperStore!")

Cart = []

Item = input("Each customer is limited to pick 2 colours. Please indicate the colour of your choice:").lower()

i=1 # Initialize a counter for colour selections

while i<3 and Item in Colour\_choice: # Limit the loop to run for two colour selections

Cart.append(Item)

Query = input("Would you like to pick another colour? (Y or N)").upper()

if Query == 'Y' :

Item2 = input("Please indicate the colour of your choice:").lower()

if Item2 == Item:

print("Duplicate colour selected. Please try again.")

elif Item2 in Colour\_choice:

Cart.append(Item2)

print("Happy Shopping! You have selected to purchase the following colours:")

for x in Cart:

print(x)

break

else:

print("Colour not available. Please try again.")

else:

print("Happy Shopping! You have selected to purchase the following colour:")

for x in Cart:

print(x)

break

if Item not in Colour\_choice:

print("Color not available. Please try again.")

Corresponding output from above Python code:

Hello, welcome to SuperStore!

Each customer is limited to pick 2 colours. Please indicate the colour of your choice: Pink

Would you like to pick another colour? (Y or N) Y

Please indicate the colour of your choice: White

Happy Shopping! You have selected to purchase the following

colours:

pink

white

The 3 rationales for rewriting the code are as follows:

1. Reduce code duplication to improve code maintainability – The initial codes uses if-else conditions for the same set of instructions. This results in a nested code blocks and when there are more than two iterations, it will require duplicating the code within multiple if-else branches to handle different user inputs which make the code to be hard to maintain. Hence, a while loop is used to mitigate code duplication as a while loop constructs a loop in the programme to continuously repeat the same set of instructions until the conditions are met. This way, the same set of instructions can be executed iteratively without the need of duplicating the codes. Thereby, creates a more maintainable codebase.
2. Simplify code to enhance code readability – The nested code blocks from the if-else condition makes the code lengthy and hard to read. Thereby using the while loop allows for a more concise and organised code which in return enhances code readability.
3. Continuous interaction to improve overall quality of program – The while loop create a more interactive experience for the users’ as the program continuously prompt for user input until a specific exit condition is met. In this case, if a user selects a duplicated or incorrect colour, the program will promptly requests for another colour selection rather than displaying an error message and requesting user to retry. This in turn enhances the overall quality of the program.

(245 words)

**Question 2**

Modified python code as per Appendix 1:

products = ["laptop","mouse","Webcam","keyboard","speaker"] #Create a list of products

query = 'yes'

updated\_items = [] #Creates empty list to store user input

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

while query == 'yes':

item = str(input('Hello! What do you want to buy?')).lower() #Added .lower() to convert user input into lower case to match the products in list

if item not in products: #Check if item from user input is within the list of products

print(f'Wrong product! Please try again')

elif item in products:

try:

price\_of\_item = float(input("How much is it (in SGD)?")) #Added validation to handle errors for input if price input is negative or inappropriate

if price\_of\_item > 0:

entered\_input = [item, price\_of\_item]

updated\_items.append(entered\_input) #Append item and price of item in the empty list defined above

else:

print("Invalid input. Please enter a valid price.")

except ValueError:

print("Invalid input. Please enter a valid price.")

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

if query == 'no':

break

print('This is your updated shopping list:')

for item, price\_of\_item in updated\_items:

print(f"- {item}: SGD ${price\_of\_item:.2f}") #Formatting to round up price to 2dp

The code in Appendix 1 can be improved through the following suggestions:

1. Error handling – As the code consist of several user inputs, it is inevitable that the user might input invalid or wrong values. As such, the code was enhanced to use the try-except block when users input for the price of item. It checks if the price is a non-negative float. In the event that an invalid price is entered, the program will display an error message to prompt user to try again preventing incorrect data to be displayed. This in turn enhances the reliability of program.
2. Formatting display – As the code requires a price input, using price\_of\_item.2f helps to round up the price display to 2 decimal places. This ensures that the displayed prices are formatted consistently which prevents floating-point precision issues. Hence, enhancing the reliability of code.
3. Adding comments – The use of comments enhances the readability as it helps explain the purpose of each code section (R, 2023). As a result, this makes it easier for other developers to understand the code.

(178 words, including in-text citation)

**References**

Lee, C. (2020, July 28). What is Programming Plagiarism? Why Is It on the Rise? Turnitin.

Retrieved from: <https://www.turnitin.com/blog/what-is-programming-plagiarism-why-is-it-on-the-rise>

Lee, C. (2021, July 14). What is source code plagiarism and what does it have to do with

academic integrity? Turnitin. Retrieved from: <https://www.turnitin.com/blog/what-is-source-code-plagiarism-and-what-does-it-have-to-do-with-academic-integrity#:~:text=1.,as%20your%20own%20without%20attribution>.

LinkedIn (n.d.). What's your process for avoiding plagiarism when using code from other

sources? Retrieved from: <https://www.linkedin.com/advice/1/whats-your-process-avoiding-plagiarism-when-using#:~:text=Modify%20the%20code&text=You%20should%20also%20avoid%20using,the%20code%20you%20are%20using>.

R. A. S. (2023, August 18). Comments in Python: Why are They Important And How to Use

Them. Retrieved from: <https://www.simplilearn.com/tutorials/python-tutorial/comments-in-python#:~:text=Writing%20good%20comments%20in%20Python,to%20grasp%20the%20programming%20language>.

Wu, K. Y., & Zhu, S. (2023). ANL252 Python for data analytics. Singapore University of

Social Sciences.