

**ANL252**

**Python for Data Analytics**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Tutor-Marked Assignment**

**July 2023 Presentation**

|  |  |
| --- | --- |
| Name | Goh Jia Yi Samuel |
| SUSS PI No | Q2311483 |

Q1a)

Reasons for Plagiarism in Coding:

Lack of Understanding: Inadequate comprehension of code concepts leads to copying without comprehension.

Easy Access: Open-source code and online resources make copying code easier.

Unethical Behaviour: Some students resort to plagiarism due to a lack of ethical values.

Time Constraints: Pressure to meet deadlines might tempt students to plagiarize.

Peer Pressure: Peer competition can drive students to dishonest practices.

*(Turnitin, 2022)*

How to Avoid Plagiarism in Coding:

Learn and Understand: Deepen understanding of code principles to write original solutions.

Practice and Experience: Regular coding practice enhances coding skills and self-reliance.

Use References Ethically: When referencing existing code, credit the source properly.

Read and Modify: Study existing code, grasp its logic, and adapt it to your unique needs.

Debugging and Optimization: Customizing code for efficiency showcases understanding and originality.

*(OIT, n.d.)*

Implementing these strategies fosters originality, upholds academic integrity, and prepares students for ethical coding practices in professional settings.Q1b)

# Initialize an empty cart list and dictionary of item prices

cart = []

item\_prices = {"coffee": 1.00,"tea": 1.50, "juice": 2.00}

# Define out-of-stock items

out\_of\_stock\_items = ["tea"]

# Function to display cart contents and calculate total cost

def display\_cart(cart):

total\_cost = 0

print("Items in your cart:")

for item in cart:

print(item)

total\_cost += item\_prices[item]

print(f"Total cost: ${total\_cost}")

# Main shopping loop

while True:

print("\nAvailable items: coffee, tea, juice")

choice = str(input("Add an item to your cart (or 'done' to finish shopping): ").lower())

if choice == "done":

display\_cart(cart)

break

elif choice in item\_prices:

if choice in out\_of\_stock\_items:

print(f"{choice.capitalize()} is out of stock. Cannot add to cart.")

else:

cart.append(choice)

print(f"{choice.capitalize()} added to your cart.")

else:

print("Invalid choice. Please enter a valid item.")

# End of shopping

print("Thank you for shopping with us!")

This code simulates a simplified vending machine shopping experience. It initializes an empty shopping cart and assigns prices to different items. It identifies certain items as out of stock. The user interacts with the program by adding items to the cart. The program calculates and displays the contents of the cart, including the total cost. If an item is out of stock, it informs the user. The code ensures the user's input is valid and responds accordingly. The shopping process continues until the user decides to finish. The code aims to replicate a shopping scenario while efficiently managing the cart, prices, and availability of items, providing a basic understanding of programming logic and user interaction.

Q1c)

# Initialize an empty cart list and dictionary of item prices

cart = []

item\_prices = {"coffee": 1.00, "tea": 1.50, "juice": 2.00}

# Define out-of-stock items

out\_of\_stock\_items = ["tea"]

# Function to display cart contents and calculate total cost

def display\_cart(cart):

total\_cost = 0

print("Items in your cart:")

for item in cart:

print(item)

total\_cost += item\_prices[item]

print(f"Total cost: ${total\_cost}")

# Main shopping loop

while True:

print("\nAvailable items: coffee, tea, juice")

choice = input("Add an item to your cart (or 'done' to finish shopping): ").lower()

if choice == "done":

display\_cart(cart)

break

elif choice in item\_prices:

if choice in out\_of\_stock\_items:

print(f"{choice.capitalize()} is out of stock. Cannot add to cart.")

else:

cart.append(choice)

print(f"{choice.capitalize()} added to your cart.")

else:

print("Invalid choice. Please enter a valid item.")

# End of shopping

print("Thank you for shopping with us!")

Originality and Learning Enhancement:

Rewriting the code promotes originality and creativity in problem-solving.

By reimagining the logic and structure, it encourages a deeper understanding of coding concepts.

Engaging in the process of rewriting enables active learning, as it necessitates applying knowledge to create new solutions.

Ethical Coding Practices:

Avoiding coding plagiarism upholds ethical principles by respecting the intellectual property of others.

Properly acknowledging sources, even in coding, fosters an atmosphere of honesty and integrity.

Adhering to ethical practices in coding aligns with broader societal expectations and professional standards.

Demonstration of Mastery:

Rewriting the code showcases a higher level of mastery over the subject matter.

The ability to reinterpret and reshape the code demonstrates a command over programming concepts and logical thinking.

It signifies a deeper engagement with the material beyond mere replication, indicating true comprehension and expertise.

Skill Development:

The process of rewriting challenges one's coding skills, encouraging experimentation and problem-solving.

Identifying and implementing different strategies to accomplish the same task strengthens coding proficiency.

This approach helps learners develop versatility and adaptability in approaching coding challenges.

Preparation for Real-World Scenarios:

In real-world programming scenarios, originality and problem-solving are vital.

Rewriting code prepares individuals for diverse coding challenges they might face in professional settings.

Employers value programmers who can innovate, adapt, and find unique solutions to problems.

*(Upadhyaya, 2023)*

Q2)

Modularization:

Break down the code into separate functions for different tasks like input gathering, validation, and displaying the updated shopping list. This improves readability and allows for easier debugging and testing of individual components.

Use Meaningful Variable Names:

Choose descriptive variable names to enhance code clarity. For example, replace query with continue\_shopping, and entered\_input with item\_details.

Exception Handling:

Implement exception handling to handle potential errors gracefully, such as invalid input for prices. This improves reliability and prevents the program from crashing unexpectedly.

Documentation and Comments:

Add comments to explain the purpose of the code and each function. This provides clarity for anyone reading or maintaining the code in the future.

Formatted Strings:

Utilize f-strings consistently for formatted output to improve readability and reduce concatenation.

Validate Input:

Validate user input for prices to ensure they are valid numeric values before appending them to the updated\_items list.

Use of Functions:

Create a function to display the product list to eliminate redundancy and improve maintainability.

*(Jagannath, 2023)*

def display\_product\_list(products):

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

def shopping\_program():

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

continue\_shopping = "yes"

updated\_items = []

display\_product\_list(products)

while continue\_shopping == "yes":

item = input("Hello! What do you want to buy? ")

if item not in products:

print("Wrong product! Please try again.")

continue

try:

price\_of\_item = float(input("How much is it (in SGD)? "))

except ValueError:

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

continue

item\_details = [item, price\_of\_item]

updated\_items.append(item\_details)

continue\_shopping = input("Would you like to continue? (yes/no) ")

print(f"This is our updated shopping list: {updated\_items}")

shopping\_program()

Reference:

Turnitin. (2022, March 8). What is programming plagiarism? Why is it on the rise? Retrieved from <https://www.turnitin.com/blog/what-is-programming-plagiarism-why-is-it-on-the-rise>

Office of Information Technology (OIT). (n.d.). Avoiding plagiarism. Retrieved from

<https://www.oit.edu/library/help/avoiding-plagiarism#:~:text=It%20is%20also%20considered%20plagiarism,actual%20code%20copied%20from%20elsewhere>

Upadhyaya, A. (2023). Coding improves creativity. Medium.

<https://medium.com/@upadhyayananyaofficial46/coding-improves-creativity-fcc2424db5c5>

Jagannath, S. (2023) *Writing clean, readable and maintainable code*, *DebugPointer*.

<https://debugpointer.com/code/write-clean-readable-code>