

**ANL252**

**Python for Data Analytics**

# **Tutor-Marked Assignment**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **LIM KAIWEN** | **N2270422** |

**Tutorial Group: ­­­­­­­­­­­­ T03**

**Instructor’s Name: MUNISH KUMAR**

**Submission Date: 15/09/2023**

1a)

Plagiarism in coding refers to the author copying the code without providing any valid authorization or credit to the original author who wrote the code.

**Reasons for plagiarism**

1) Lack of knowledge and expertise: Programmers especially junior programmers that may be new to the programming language they are learning and may prone to perform plagiarism especially since there are many sources from which they can copy such as GitHub. (Krishna, 2019)

2) Stress and pressure for high grades: Students may perform plagiarism due to a tight deadline for assignments in school therefore to submit their assignments on time, they perform plagiarism. (Krishna, 2019)

**To avoid plagiarism in coding**

1) Use plagiarism checker:Plagiarism detection tools such as Codeequiry to help verify the code's originality and to ensure it is free from plagiarism. (Leslieshuffman, 2020)

2) Cite the sources and give proper attribution: It is important for programmers to always provide citations for any original source that they copy or borrow from. (Leslieshuffman, 2020)

3) Commenting: Programmers should write comments on their code as it not only provides clarity to the users, but also highlights which code has been borrowed from external sources. (Leslieshuffman, 2020)

Word Count:193

1b)

#Create a Simple BMI calculator using 30 lines

# User input information such as height and weight

weight = float(input("Enter your weight in kilograms: "))

height\_cm = float(input("Enter your height in centimeters: ")) # Accept height in cm

# Convert height from cm to m and calculate BMI

def calculate\_bmi(weight\_kg, height\_cm):

height\_m = height\_cm / 100

return weight\_kg / (height\_m \*\* 2)

#interpret BMI results

def interpret\_bmi\_results(bmi):

if bmi < 18.5:

return "Underweight"

elif 18.5 <= bmi < 24.9:

return "Normal Weight"

elif 24.9 <= bmi < 29.9:

return "Overweight"

else:

return "Obese"

# Calculate BMI and interpret result

bmi = calculate\_bmi(weight, height\_cm)

result = interpret\_bmi\_results(bmi)

# Display results based on entered input values

print(f"You are {result}")

print(f"Your BMI is {bmi:.2f}")

The above code helps the users to calculate and interpret the results of the Body Mass Index (BMI) category based on the entered inputs. Firstly, users will be prompted to key in its height and weight using the ‘input’ function. As weight can have decimals, therefore the inputs are stored as float variables to allow decimal values to be entered. A ‘calculate\_bmi’ function is created that takes two parameters height (cm) and weight (kg). The function uses the formula BMI = weight (kg) / (height\_m2) to convert height from cm to m and determine the BMI. A ‘interpret\_bmi\_results’ is created which will collect the calculated BMI results and return them as a string. The use of the ‘if-elif and else statements’ helps to set the conditions of the BMI categories such as underweight, normal weight, overweight, and obese. The calculated BMI results will be sorted into different categories based on the conditions. The results then will be stored in the ‘result’ variable which will include the calculated BMI results and appropriate BMI category. The ‘print’ function will be used to display the calculated BMI results and the interpreted BMI category to the user. (Rishabh Singh, 2021)

Word count:193

1c)

#Create a Simple BMI calculator

# Convert height from cm to m and calculate BMI

def calculate\_bmi(weight\_kg, height\_cm):

height\_m = height\_cm / 100

return weight\_kg / (height\_m \*\* 2)

#interpret BMI results

def interpret\_bmi\_results(bmi):

if bmi < 18.5:

return "Underweight"

elif 18.5 <= bmi < 24.9:

return "Normal Weight"

elif 24.9 <= bmi < 29.9:

return "Overweight"

else:

return "Obese"

# Input user validation

def input\_valid\_input(prompt, value\_type):

while True:

try:

value = float(input(prompt))

if value <= 0:

raise ValueError(f"{value\_type} must be a positive value.")

return value

except ValueError as e:

print(f"Invalid input: {e}")

# User key in input such as height and weight

weight = input\_valid\_input("Enter your weight in kilograms: ", "Weight")

height\_cm = input\_valid\_input("Enter your height in centimeters: ", "Height (in cm)")

# Calculate BMI and interpret result

bmi = calculate\_bmi(weight, height\_cm)

result = interpret\_bmi\_results(bmi)

# Display results based on entered values

print(f"You are {result}")

print(f"Your BMI is {bmi:.2f}")

1) Input user validation: By using the function ‘input\_valid\_input’, it helps to input user validation for both height and weight parameters using a single function. This function helps to minimise redundancy and simplify maintenance as any adjustment to the user validation can be executed within the function, ensuring consistency in the error messages and validation rules. (Borislav Hadzhiev,2023)

2) Improve readability: By incorporating user validation, the code also becomes more user-friendly for the user. The value\_type parameter, which the input\_valid\_input function accepts, indicates whether the input is for weight or height. By displaying informative prompts and error warnings, this will improves the code readability. (Borislav Hadzhiev,2023)

3) Improve user experience: By rewriting the code, it also enhances the user experience. The input\_valid\_input function helps to ensure that inputs that are entered by the user are positive and valid. For example, if a user keys in a negative value for weight, it will display an error message such as “Invalid input: Weight must be a positive value” which will prompt the user to key in a positive value for the weight. This validation also applies to the parameter, height. Hence, this validation logic helps to reduce error and ensures that the results for the BMI calculation are accurate. (Borislav Hadzhiev,2023)

4) Enhanced user-friendly interaction: The code also has a user-friendly interaction function which is supported by the ‘while loop function’. Implementing a while loop in the code provides the user with numerous opportunities to rectify their inputs instead of abruptly halting the program or displaying an error. Hence, the user experience is improved, and it is able to reduce the frustration of the users. (Borislav Hadzhiev,2023)

Word count: 275

2)

#Function to add products items into the shopping list

def add\_item\_to\_shopping\_list(products, updated\_items):

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

# Ensure user only selects items that are in the list

if item not in products:

print(f'Product "{item}" not found in the list. Please choose from {products}.')

return

# Input validation for price (must be a positive float)

while True:

try:

price\_per\_item = float(input(f'How much is the {item} (in SGD per item): '))

if price\_per\_item <= 0:

raise ValueError("Price must be a positive number.")

break # Exit the loop if the input is valid

except ValueError as e:

print(f"Invalid input: {e}")

total\_price = price\_per\_item

entered\_input = [item, price\_per\_item, total\_price]

updated\_items.append(entered\_input)

# Function to display the user's shopping list

def display\_shopping\_list(updated\_items):

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

total\_amount = 0

for item, price\_per\_item, total\_price in updated\_items:

print(f'{item}: ${price\_per\_item:.2f}')

total\_amount += total\_price

print(f'Total amount: ${total\_amount:.2f}')

# Display the list of items for the users

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

updated\_items = []

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

while True:

add\_item\_to\_shopping\_list(products, updated\_items)

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

if query.lower() != 'yes':

break

display\_shopping\_list(updated\_items)

Firstly, in terms of code structure, the rewritten code is more organised and easier to understand. It follows a modular approach where it has different functions with specific tasks such as def\_add\_item\_to\_shopping\_list and def\_display\_shopping list which enhance code readability as compared to the original code where all the logic is in a single block which makes it difficult to understand the individual steps. In addition, in terms of the variable name, the rewritten code includes variable names such as price\_per\_item and total\_price which is easy for the users to understand as compared to the original code which uses generic variable names such as query. (Borislav Hadzhiev,2023)

Secondly, in terms of validation, the rewritten code has input validation to improve the reliability and ensure that users provide valid positive float values as compared to the original code which is lacking in terms of validation and can be frustrating for the user if they accidentally enter a non-numeric value. (Borislav Hadzhiev,2023)

Lastly, in terms of maintainability, the modular structure of the rewritten code makes it much easier to maintain. In addition, additional functions can be implemented easily by making changes to individual functions without affecting the entire code. (Borislav Hadzhiev,2023)

Word Count: 200

**References**

1. Krishna, S. (2019, November 13). *Dealing with plagiarism in coding classes*. Compute Thought. https://computethought.blog/2019/01/11/dealing-with-plagiarism-in-coding-classes/

2. Leslieshuffman. (2020, November 9). *How to avoid plagiarism when taking a coding assessment for a Job*. Dice Insights. <https://www.dice.com/career-advice/how-to-avoid-plagiarism-when-taking-a-coding-assessment-for-a-job>

**Code References**

Question 1

3. Rishabh Singh, R. S. (2021, February 17). *How to build a BMI calculator in python*. DEV Community. https://dev.to/mindninjax/how-to-build-a-bmi-calculator-in-python-4g2g

4. Borislav Hadzhiev(2023, February 20). *How to validate user input in python*. bobbyhadz. (n.d.). <https://bobbyhadz.com/blog/python-input-validation>

Question 2

5. Borislav Hadzhiev(2023, February 18) *How to take a list from user input in python*. bobbyhadz. (n.d.-a). https://bobbyhadz.com/blog/python-add-user-input-to-list