

**ANL252**

**Python for Data Analytics**

# **Tutor-Marked Assignment**

**July 2023 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **BEVERLYN TAN JIAMIN** | **E2081428** |

**Tutorial Group: ­­­­­­­­­­­­ T03**

**Instructor’s Name: Prof. Kumar Munish**

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

**Question 1**

**(a)**

Why plagiarism in coding happens

1. Availability of solutions – Codes that are available for reuse can be easily obtained on the internet, such as GitHub, where prior course participants have uploaded answers to topics and assignments (Lee, C., 2023).
2. Lack of comprehension and time constraints – Students who are stressed out or having difficulty in class could feel pressured to use shortcuts in their assignments. Lee, C., 2023).
3. Failure to acknowledge the original author’s work – Failure to acknowledge the source and authorship of the source-code with relevant documentation; or using phoney or false references (Cosma, G., & Joy, M. ,n.d.).

How to avoid this issue

1. Acknowledge appropriately – Cite the source code to properly acknowledge the author’s work (Development, W., n.d.).
2. Modify the code – Tailor the codes to your needs while demonstrating originality by utilising the source-code as a base and not a mere copy-paste. Additionally, refrain from slight modifications and excessive similarity (Development, W., n.d.).
3. Check for plagiarism – Employ plagiarism-checking tools to review your code and spot for similarities. Make modifications to the code if there are any potential plagiarism risk. (Development, W., n.d.).

**(b)**

The purpose of the following code is to make it simple for users to find their grades by simply entering their Section A and Section B scores. The final score will then be added up, and the user will be prompted with the relevant grade and message. Instead of manually adding up the results and matching them to the letter grade system, this enables quick calculation and matching to the respective grade.

Python code:

# To code a function to assign different scores to their respective grade

def gradefinder(grade):

if (grade >= 80):

print ("You got an A, well done!")

elif (grade >= 65):

print ("You got a B, almost there!")

elif (grade >= 50):

print ("You got a C, you need to work harder!")

else:

print ("You got a F, I'll see you in the next semester!")

# To code a function to total up the score and assign them to their respective grade through the gradefinder fucntion

def score(SectionA, SectionB):

finalscore = SectionA + SectionB

(gradefinder(finalscore))

return

#To prompt the user to input their scores

SectionA = int(input("Input your score for Section A: "))

SectionB = int(input("Input your score for Section B: "))

score(SectionA,SectionB)

The corresponding output for the code would be:

Input your score for Section A: 3

Input your score for Section B: 55

You got a C, you need to work harder!

(c)

Rationale behind rewriting the code

1. To allow error handling – Since a score can only be an integer, the rewritten code includes error handling components which ensures that the user provides a valid input unlike the preceding code. This enhances user experience and prevents errors driven by unintended input.
2. To allow better readability and maintainability – The rewritten code is made easier to comprehend and maintain by utilising descriptive function and variable names, make it more open to collaboration and future revisions. For example, the function ‘score’ in the preceding code is confusing its purpose cannot be inferred from its name. However, in the revised code, the function ‘compute\_overall\_score’ explicitly conveys its functionality making it more meaningful.
3. Better coding practices – Better coding practices, such as meaningful function name, input validation and a distinct separation of concerns, are used in the rewritten code (Dichcovsky, P., 2023). For instance, the rewritten code uses guidance-oriented error messages rather than merely alerting the user that there is an error when receiving an invalid input.

Rewritten code:

# To code a function to assign different scores to their respective grade

def determine\_grade(score):

if score >= 80:

return "A, well done!"

elif score >= 65:

return "B, almost there!"

elif score >= 50:

return "C, you need to work harder!"

else:

return "F, I'll see you in the next semester!"

# To compute the total score of Part A and Part B

def compute\_overall\_score(PartA, PartB):

return PartA + PartB

# To ensure that input value must be a integer else the error message will keep appearing

def only\_integer\_input(prompt):

while True:

user\_input = input(prompt)

if user\_input.isdigit():

return int(user\_input)

else:

print("Error! Please enter a valid integer.")

# To assign the functions to their respective roles

PartA\_score = only\_integer\_input("Input your score for Section A: ")

PartB\_score = only\_integer\_input("Input your score for Section B: ")

Total\_score = compute\_overall\_score(PartA\_score, PartB\_score)

grade\_message = determine\_grade(Total\_score)

# To print intended output

print(f"Your total score is {Total\_score}. You got a {grade\_message}")

The corresponding output for the code would be:

Input your score for Section A: 14

Input your score for Section B: A

Error! Please enter a valid integer.

Input your score for Section B: 22

Your total score is 36. You got a F, I'll see you in the next semester!

**Question 2**

The first point to enhance its readability would be to make the variable more readable by being more descriptive. For example instead of ‘products’, ‘available\_products’ can be used which directly communicate the intended meaning.

The second point is to provide comments in the code to allow users to better comprehend its purpose.

The third point is to allow user re-enter the input when an invalid input is received rather than breaking the programme to enhance reliability. Furthermore, by providing directional messages to the user rather than just stating that there was an error improves the quality of the programme.

The fourth point is that consistent naming should be used. Unlike in Appendix 1 where item and products are used, the revised code only uses ‘product’ to avoid any confusion that may arise which allows the code to be maintained easily while improving user comprehension.

The final point is to use built-in Python functions such as ‘.lower()’ and ‘float’ to enhance the program’s usability and overall quality. These functions enable the programme to take into consideration any unexpected inputs that a user might provide.

Revised Code:

# To use a more descriptive variable name to make it more readable

available\_products = ['laptop','mouse','webcam','keyboard','speaker']

user\_input = 'yes'

shopping\_list = []

# To let the customers know what products are avaliable

print(f'We have the following available products here: {available\_products}.')

# To convert the input string into lowercase in the event user input contains uppercase letters

# To remove any trailing whitespace and convert the user input to lowercase letters

while user\_input.lower() == 'yes':

product = input("Hello! What product would you like to purchase? ").strip().lower()

# To allow user to retry when an invalid product is selected

if product not in available\_products:

print(f'Sorry, the selected product is not available. Please select from {available\_products}')

continue

# To allow user to retry when an invalid price is entered

while True:

try:

price = float(input(f'How much is {product} (in SGD)? ')) # To convert the user's input into a floating-point number

if price.is\_integer():

break

else:

print('Sorry, invalid input received. Please enter a number as the valid price.')

except ValueError:

print('Sorry, invalid input received. Please enter a number as the valid price.')

# To display the shopping list with the product names

shopping\_list.append(product)

# To remove any trailing whitespace and convert the user input to lowercase letters

user\_input = input("Would you like to add more products to your shopping list? (yes/no) ").strip().lower()

#To show the user the final shopping list with prices

print('Your shopping list: ')

for product in shopping\_list:

print(f'Product: {product}, Price: {price} SGD')

print('Thank you for shopping with us, see you again!')

Relevant output:

We have the following available products here: ['laptop', 'mouse', 'webcam', 'keyboard', 'speaker'].

Hello! What product would you like to purchase? MousE

How much is mouse (in SGD)? 33

Would you like to add more products to your shopping list? (yes/no) yes

Hello! What product would you like to purchase? LaPTop

How much is laptop (in SGD)? aa

Sorry, invalid input received. Please enter a number as the valid price.

How much is laptop (in SGD)? 33

Would you like to add more products to your shopping list? (yes/no) no

Your shopping list:

Product: mouse, Price: 33.0 SGD

Product: laptop, Price: 33.0 SGD

Thank you for shopping with us, see you again!

**References**

Bose, S. (2023). *Coding standards and best practices to follow*. BrowserStack. https://www.browserstack.com/guide/coding-standards-best-practices

#### Built-in Exceptions — Python 3.8.17 documentation. (2023). https://docs.python.org/3.8/library/exceptions.html#ValueError

Development, W. (n.d.). *What’s your process for avoiding plagiarism when using code from other sources?* How to Avoid Plagiarism When Using Code from Other Sources. https://www.linkedin.com/advice/1/whats-your-process-avoiding-plagiarism-when-using

Dichcovsky, P. (2023) *13 good coding practices in 2023*. Qulix. https://www.qulix.com/about/blog/good-coding-practices/

Cosma, G., & Joy, M. (n.d.). *Source code plagiarism*. Source Code Plagiarism | Plagiarism Online,. http://web.science.mq.edu.au/~mtaylor/ponline/index.php?id=source-code-plagiarism#:~:text=Source%2Dcode%20plagiarism%20in%20programming,as%20his%2Fher%20own%20work.

Lee, C. (2023). *What is source code plagiarism and what does it have to do with academic integrity?*. Turnitin. https://www.turnitin.com/blog/what-is-source-code-plagiarism-and-what-does-it-have-to-do-with-academic-integrity

#### Python float(). (2023). https://www.programiz.com/python-programming/methods/built-in/float

#### Python if, if...else Statement (With Examples). (2023). https://www.programiz.com/python-programming/if-elif-else

#### Python String isdigit(). (2023). https://www.programiz.com/python-programming/methods/string/isdigit#google\_vignette

Python String strip(). (2023). https://www.programiz.com/python-programming/methods/string/strip

Sharovar, E. (2022). *Code rewriting: When and why*. Waverley. https://waverleysoftware.com/blog/code-rewriting-when-and-why/

#### Try and Except in Python - Python Tutorial . (2023). https://pythonbasics.org/try-except/