\*\*Python Fundamentals and OOP Test\*\*

\*\*Candidate Name:\*\* AKSHAY TRIPATHI

\*\*Date:\*\* August 8,2024

\*\*Instructions:\*\*

1. Answer all the questions to the best of your ability.

2. Write your answers and code in the spaces provided.

3. Use the space on the back of the paper if needed.

\*\*Question 1: Data Types\*\*

* 1. Explain the differences between a list and a tuple in Python.

Your Answer:

|  |  |
| --- | --- |
| **LIST** | **TUPLE** |
| 1. List are defined using square brackets.  eg. [‘a’,’k’,’s’,’h’,’a’,’y’] | 1. Tuple are defined using circular brackets. eg. (‘a’,’k’,’s’,’h’,’a’,’y’) |
| 2. List are mutable ie. changeable | 2. Tuple are immutable ie. Can not be modified. |
| 3. list are slower than tuple | 3.Tuples are faster than list |

\*\*Question 2: Arithmetic Operations\*\*

2.1. Write a Python code snippet to calculate the sum of all even numbers from 1 to 50.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: to calculate the sum of all even numbers from 1 to 50.

(Pyhton test Question 2)

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

#variable to store the sum

even\_sum = 0

# creating a loop from 1 to 50

for num in range(1, 51):

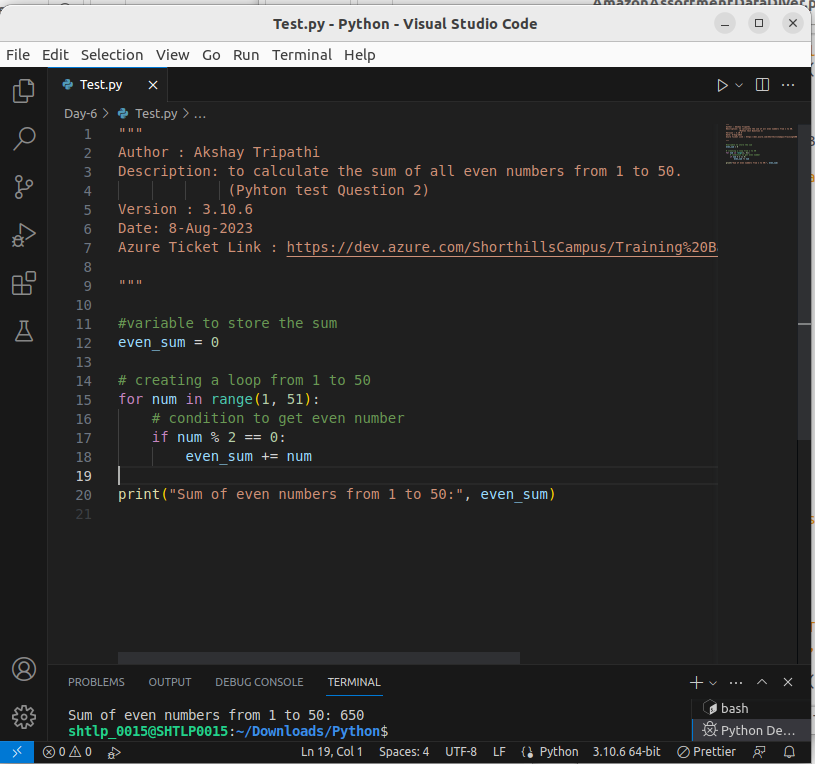
# condition to get even number

if num % 2 == 0:

even\_sum += num

print("Sum of even numbers from 1 to 50:", even\_sum)

```



2.2. Given the list `[2, 4, 6, 8]`, use a loop to calculate the product of all the numbers in the list.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: to calculate the product of all the numbers in the given list..

(Pyhton test Question 3)

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

#Given List

numbers=[2,4,6,8]

product=1 #variable to store product

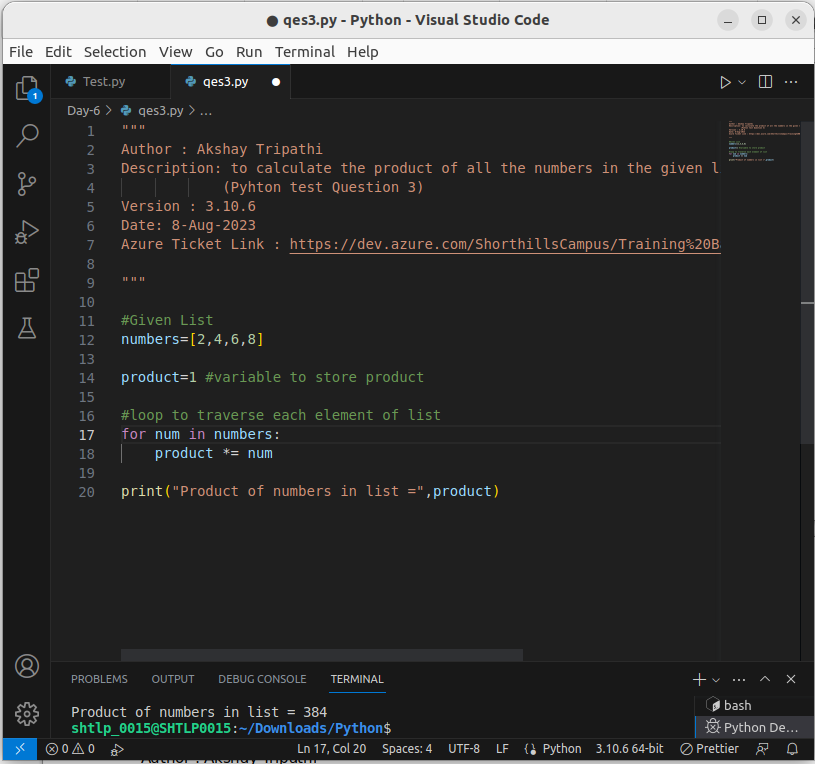
#loop to traverse each element of list

for num in numbers:

product \*= num

print("Product of numbers in list =",product)

```



\*\*Question 3: String Operations\*\*

3.1. Write a Python function called `is\_palindrome` that takes a string as input and returns `True` if the string is a palindrome (reads the same forwards and backwards), otherwise returns `False`.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: to calculate the product of all the numbers in the given list..

(Pyhton test Question 3)

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

#function to check Palindrome

def isPalindrome(s):

return s == s[::-1]

if(isPalindrome("Akshay")):

print("Akshay Palindrome")

else:

print("Akshay is not Palindrome")

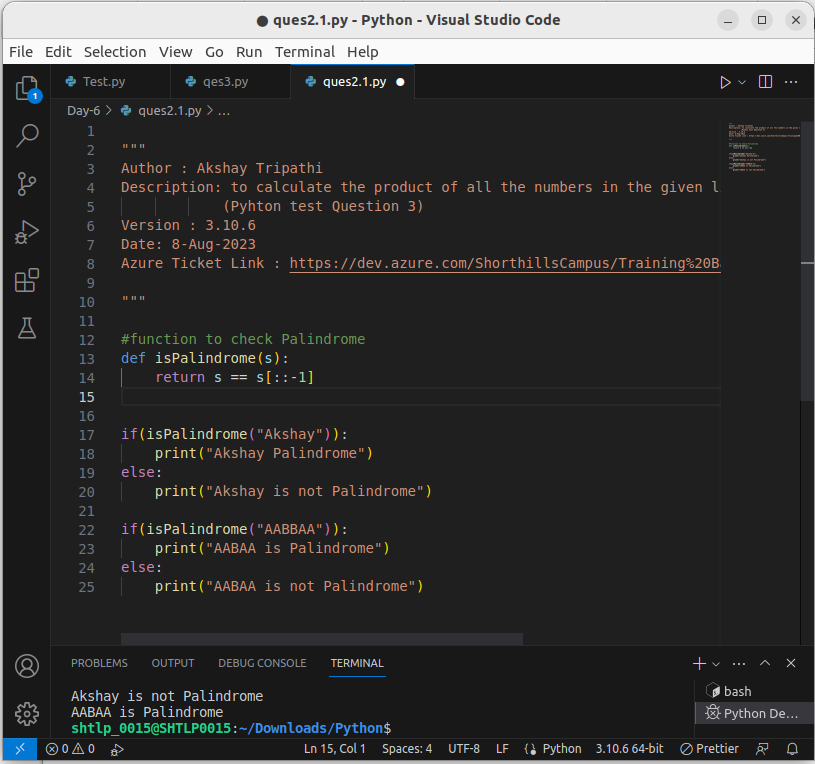
if(isPalindrome("AABBAA")):

print("AABAA is Palindrome")

else:

print("AABAA is not Palindrome")

```



3.2. Given the string `”Python is fun”`, write a Python code snippet to reverse the words in the string so that it becomes `”fun is Python”`.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: the string `”Python is fun”`, write a Python code snippet to reverse the words in the string so that it becomes `”fun is Python”`

(Pyhton test Question 3.2)

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

given\_string="Pyhton is fun"

# Split the string into words using whitespace

words = given\_string.split()

# Reversing the order of words

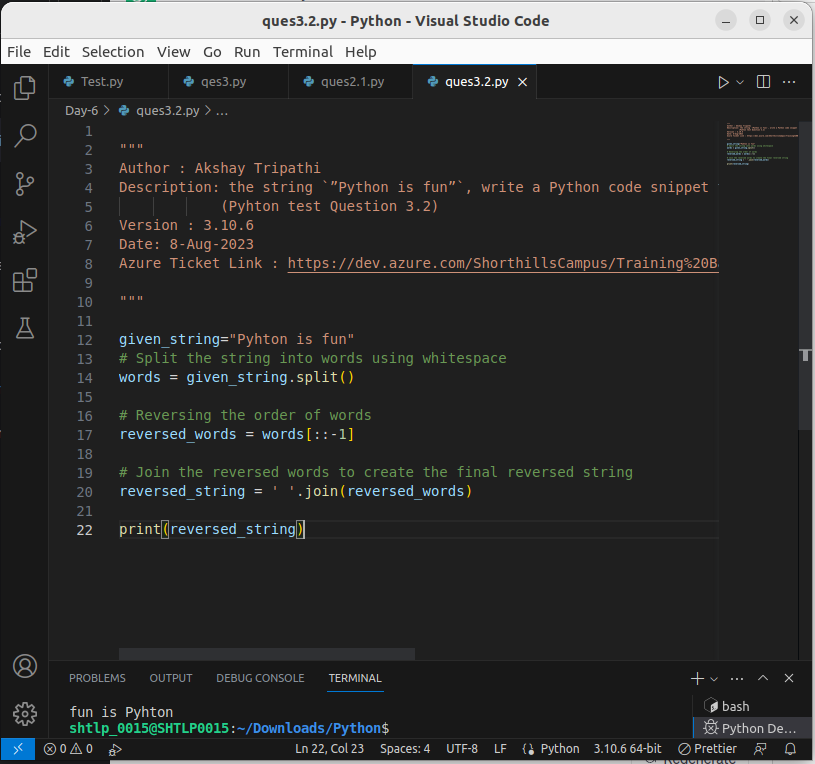
reversed\_words = words[::-1]

# Join the reversed words to create the final reversed string

reversed\_string = ' '.join(reversed\_words)

print(reversed\_string)

```



\*\*Question 4: List Operations\*\*

4.1. Write a Python function called `unique\_elements` that takes a list as input and returns a new list containing only the unique elements in the input list.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: to calculate unique elements from input list

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

def UniqueElements(input\_list):

# Initialize an empty list to store unique elements

unique\_list = []

# loop to find unique elemnts from list and append in unique\_list

for item in input\_list:

if item not in unique\_list:

unique\_list.append(item)

return unique\_list

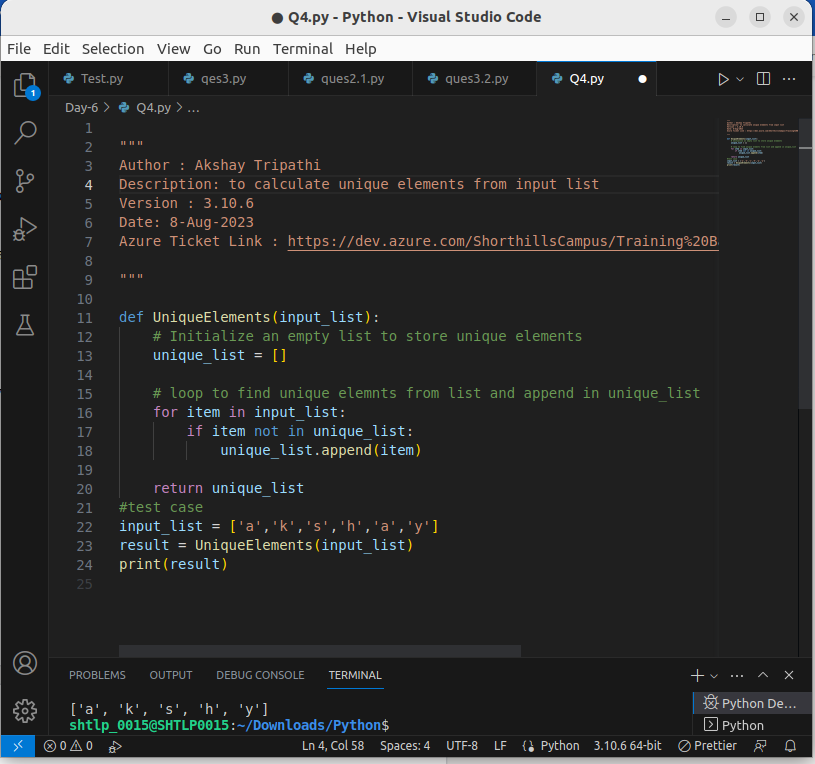
#test case

input\_list = ['a','k','s','h','a','y']

result = UniqueElements(input\_list)

print(result)

```



4.2. Given two lists `[1, 2, 3]` and `[3, 4, 5]`, write a Python code snippet to find the common elements between the two lists.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: to find common elements from two list

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

#Function to find common elements from two lists

def FindCommonElements(list1, list2):

# a third list to store common elements

common\_elements = []

# Iterate through the elements in the first list

for item in list1:

# Check if the item is present in the second list

if item in list2:

common\_elements.append(item) # Add common element to the list

return common\_elements

# Test the function

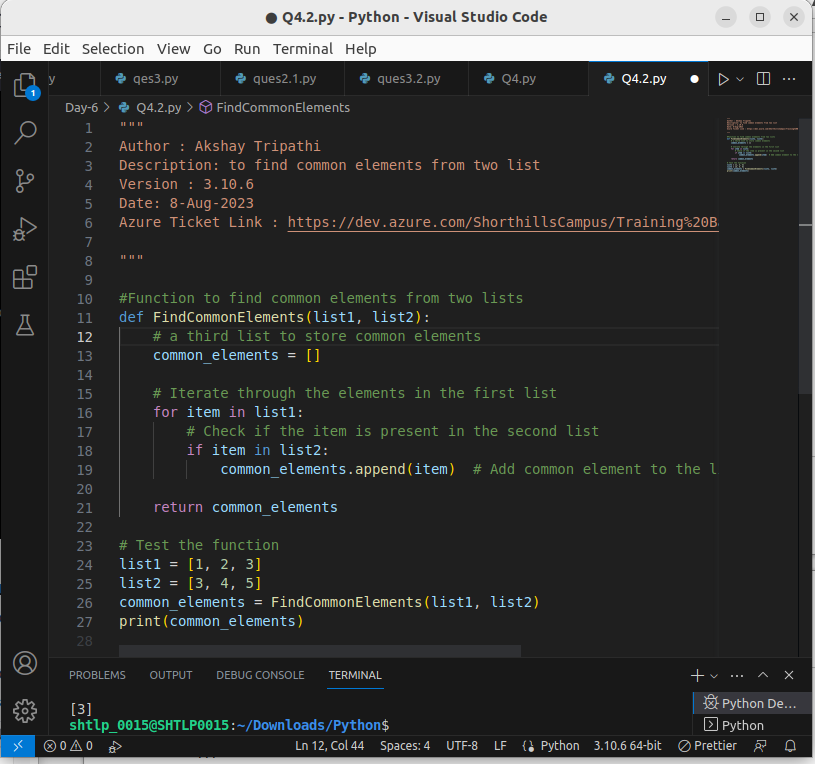
list1 = [1, 2, 3]

list2 = [3, 4, 5]

common\_elements = FindCommonElements(list1, list2)

print(common\_elements)

```



\*\*Question 5: Object-Oriented Programming (OOP)\*\*

5.1. Define a class `BankAccount` with attributes `account\_number`, `account\_holder`, and `balance`. Implement methods `deposit` and `withdraw` to update the balance.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: to Define a class `BankAccount` with attributes `account\_number`, `account\_holder`, and `balance`. Implement methods `deposit` and `withdraw` to update the balance.

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

class BankAccount:

def \_\_init\_\_(self, account\_number, account\_holder, balance):

self.account\_number = account\_number

self.account\_holder = account\_holder

self.balance = balance

def deposit(self, amount):

self.balance += amount

def withdraw(self, amount):

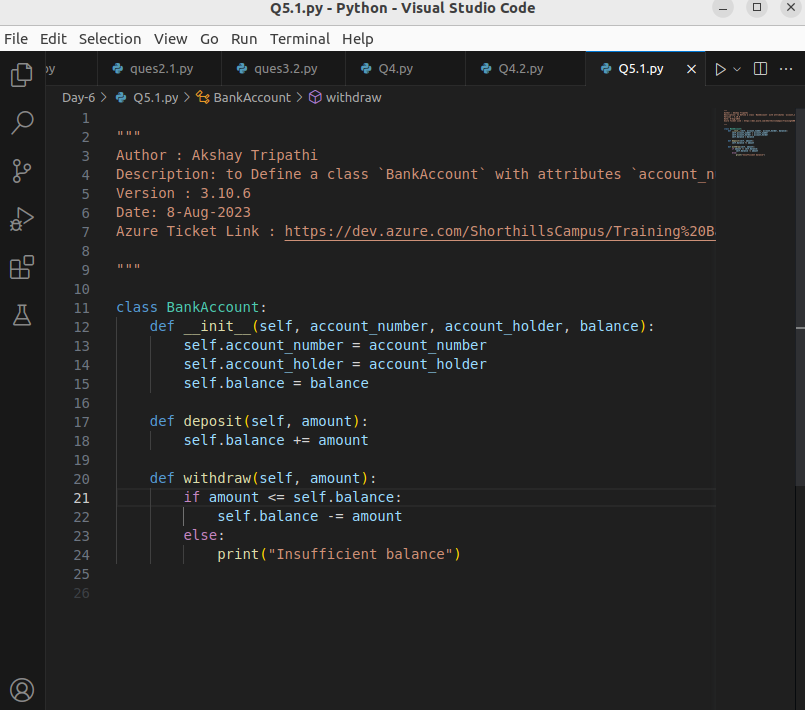
if amount <= self.balance:

self.balance -= amount

else:

print("Insufficient balance")

```



5.2. Create an instance of the `BankAccount` class, make a deposit of $1000, and then withdraw $300. Print the final balance.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: to Define a class `BankAccount` with attributes `account\_number`, `account\_holder`, and `balance`. Implement methods `deposit` and `withdraw` to update the balance.

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

class BankAccount:

def \_\_init\_\_(self, account\_number, account\_holder, balance):

self.account\_number = account\_number

self.account\_holder = account\_holder

self.balance = balance

def deposit(self, amount):

self.balance += amount

def withdraw(self, amount):

if amount <= self.balance:

self.balance -= amount

else:

print("Insufficient balance")

# Create an instance of BankAccount

account = BankAccount("1930006790", "Akshay", 0)

# Make a deposit of $1000

account.deposit(1000)

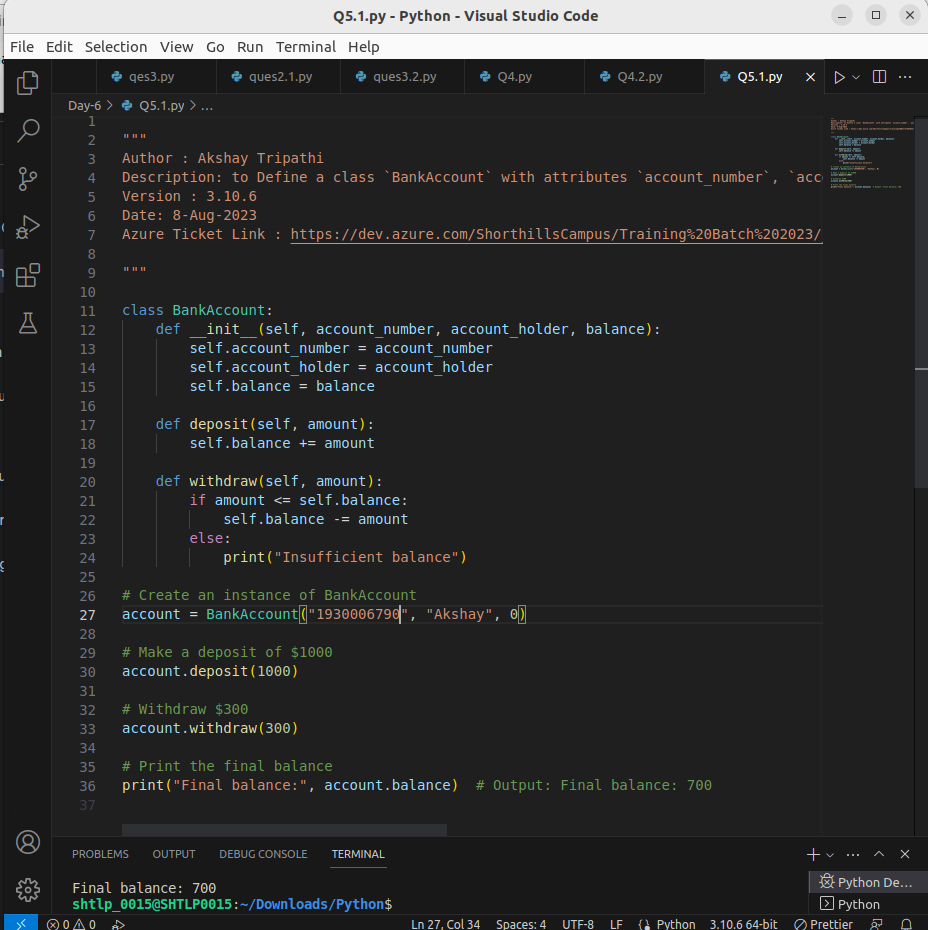
# Withdraw $300

account.withdraw(300)

# Print the final balance

print("Final balance:", account.balance) # Output: Final balance: 700

```



\*\*Question 6: Code Quality\*\*

6.1. Explain the concept of “code modularity” and how it contributes to code quality and maintainability.

Your Answer:

**Code Modularity:** Braking code into smaller task modules is know as code modularity.

Code modularity is like building house(code) using small bricks(modules). Instead of making one big piece in a go , we can make smaller, peices first ,ie. Modules which are task oriented.

And by doing small task we make full program .

**Contribution to code quality and maintainability:**

1. **Easy Reading:** Smaller parts are easier to understand,

2. **Reusing Blocks:** we can use same blocks (code) again and again in different places.

3. **Easily find** error:Due to modularity we can get exactly in which code module there is an error and fix that block without affecting others.

6.2. List three guidelines from PEP 8 that promote better code readability.

Your Answer:

1. **Indentation: T**his makes it clear which lines belong together or which lines are in a block of loop functions etc

2. **Naming Classes, variables, or Methods** : Class name should always starts with capital letter and while naming methods and variable make sure that meaning of that variable or function should be clear from that and use a proper naming conventions like Camel case.

3. **Comments: B**est way to explain about a block of code that what it is doing so that any developer could easily get that .Thus these are very important to make your code readable

\*\*Question 7: Coding Exercise\*\*

Write a Python function called `prime\_factors` that takes an integer as input and returns a list of its prime factors. For example, if the input is `56`, the output should be `[2, 2, 2, 7]`.

Your Code:

```python

# Write your code here

"""

Author : Akshay Tripathi

Description: Write a Python function called `prime\_factors` that takes an integer as input and returns a list of its prime factors. For example, if the input is `56`, the output should be `[2, 2, 2, 7]`.

Version : 3.10.6

Date: 8-Aug-2023

Azure Ticket Link : https://dev.azure.com/ShorthillsCampus/Training%20Batch%202023/\_workitems/edit/2914

"""

def PrimeFactors(n):

factors = [] # a list to store prime factors

divisor = 2

#function to store all prime divisor in factors list

while n > 1:

if n % divisor == 0:

factors.append(divisor) # Add the divisor

n //= divisor # Updating n

else:

divisor += 1 # Move to the next divisor

return factors

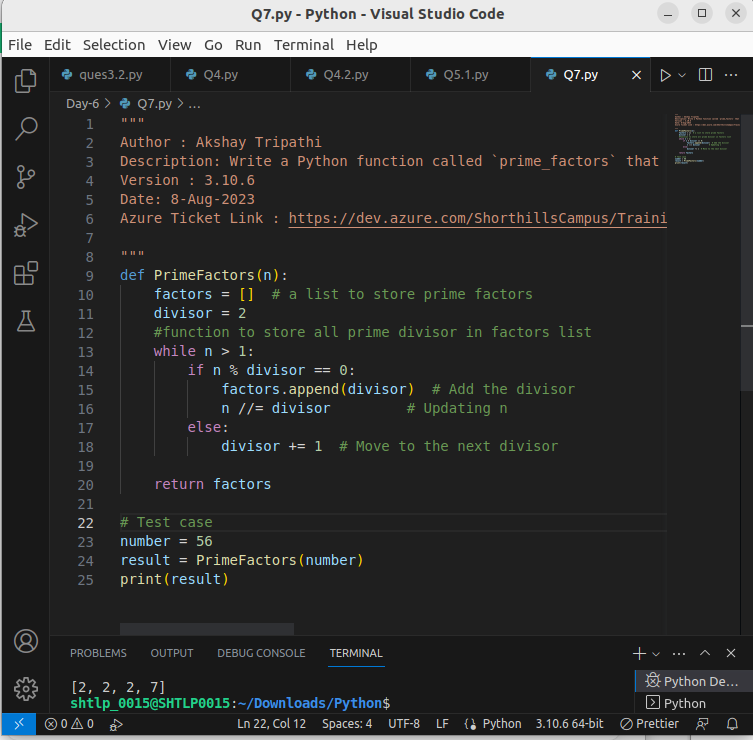
# Test case

number = 56

result = PrimeFactors(number)

print(result)

```



\*\*End of Test\*\*

Thank you for completing the test! Make sure to review your answers and code before submitting.

---