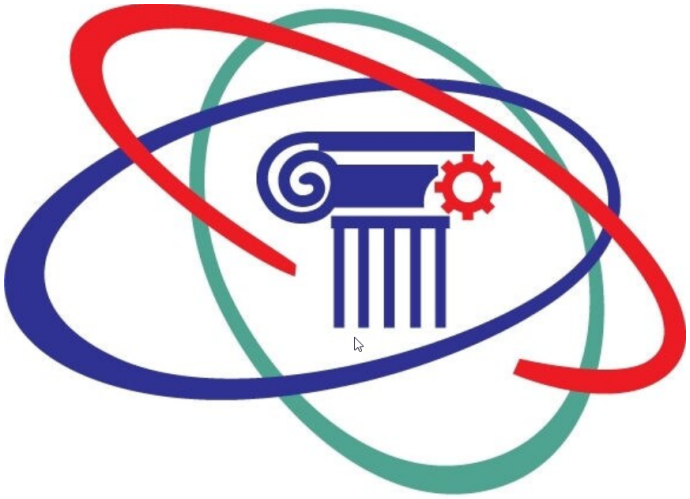
Acropolis Institute of Technology & Research

Department of Computer Science & Information Technology



SEMESTER VI
SESSION JAN-JUNE 2024

Python Program File

Submitted To:

Prof. Sweta Gupta

Submitted By:

Name: Ankit Baingane

Class: CSIT

Enrolment No: 0827CI211028

Section: I

Certificate

Department of Computer Science & Information Technology

Acropolis Institute of Technology & Research,Indore

This is to certify that the experimental work entered in this journal as per the BTech III year syllabus prescribed by the RGPV was done by Ankit Baingane in VI semester in the Laboratory of this institute during the academic year 2023- 2024

Prof.Sweta Gupta (s)

External(s)

INDEX

Name :- **class:-** **Section:-**

S.NO	DATE OF EXPERIMENT	Content	Page No.	Date of submission	REMARK
1		Installation of Python.			
2		Write a program to print different types of data types with multiline and single line comments.			
3		Write a program to print the largest number of three numbers using if else and elif.			
4		Write a program to print number 1-10 using a while loop.			
5		Write a program to check prime numbers between 1-100 using a for loop.			
6		Write a program to concatenate two strings using + operator.			
7		Write a program to reverse a string using slicing.			
8		Write a program to perform different methods of string like: len(atleast 5)			
9		Write a program to traverse all the characters of a string using a for loop.			
10		Write a program to Print abecederian series			
11		Write a program to check whether a string is present in another string or not.			
12		Write a program to print A AB ABC ABCD			
13		Write a program to Create a list with different data types.			
14		Write a program to take user input in a list from eval().			
15		Write a program to append five elements in a list by using for loop and append().			
16		Write a program to sort a list in both ascending and descending order.			
17		Write a program to count the occurrences of an element in a list.			
18		Write a program to find the index of an element in the list.			

19		**Write a program to swap the first and last element of a list.**			
20		**Write a program to swap two no in list with given position.**			
21		**Write a program to check whether a no is in the list or not sublist even and odd.**			
22		**Write a program to demonstrate the difference between remove and pop method in a list.**			
23		**Write a program to insert an element into a list at a position given by the user**			
24		**Explain difference between list, tuple, set and dictionary**			
25		**Write a program to Counts the number of occurrences of item 50 from a tuple.**			
26		**Write a program to Check if all items in the tuple are the same**			
27		**Write a Program to create a tuple which is having integer,float,list and tuple as its elements and print an element present in list of these tuple**			
28		**Write a program to create a tuple with the name of 10 cities of India Check whether a City is present in the tuple or not**			
29		**Write a program to get the number of occurrence of a word in tuple.**			
30		**Write a program to get the index of a word in tuple**			
31		**Write a program to create four tuples viz roll no. name. CGPA and SGPA of students. Print individual student's details using these 4 tuples.**			
32		**write a Python program to create a set.**			
33		**Write a Python program to iterate over sets.**			
34		**Write a Python program to add member(s) to a set.**			
35		**Write a Python program to remove Item(s) from a given set.**			
36		**Write a Python program to remove an item from a set if it is**			

37		**Write a Python program to create an intersection of sets.**			
38		**Write a Python program to create a union of sets.**			
39		**Write a Python program to create set difference**			
40		**Write a Python program to create a symmetric difference.**			
41		**write a Python program to check if a set is a subset of another set.**			
42		**Write a Python program to remove all elements from a given set**			
43		**Write a Python program to find the maximum and minimum values in a set**			
44		**write a Python program to find length of a set**			
45		**Write a Python program to check if a given value is present in a set or not.**			
46		**Write a Python program to check if two given sets have no elements in common.**			
47		**Write a Python program to create a Dictionary of student information. take roll no as key**			
48		**Write a Python progarm to delete a key from the dictionary**			
49		**Write a Python program to add or update the data.**			
50		**Write a Python script to concatenate the following dictionaries to create a new one sample Dictionary:dic1={1:10,2:20}, dic2={3:30,4:40}**			
51		**Write a Python script to check whether a given key already exists in a dictionary**			
52		**Write a Python program to iterate over dictionaries using for loops**			

53		**Write a Python script to generate and print a dictionary that contains a number (between I and n) in the form (x, x*x).Sample dictionary(n=5): Expected output: {1:1,2:4,3:9,4:16,5:25}**			
54		**Write a python function to calculate Sum of two variables.**			
55		**Write a Python function to show the use of Default Parameter**			
56		**Write a function to find the maximum of three numbers.**			
57		**Write a python function to use arbitrary argument**			
58		**Write a Python function to reverse a string**			
59		**Write a Python function to print the document string**			
60		**Write a Python function to demonstrate the scope of local and global variable.**			
61		**Write a Python function to calculate the factorial of a number (a nonnegative integer). The function accepts the number as an argument**			
62		**Write a Python function to check whether a number falls within a given range.**			
63		**Write a Python function that accepts a string and counts the number of upper and lower case letters.**			
64		**Write a Python function that takes a list and returns a new list with distinct elements from the first list.**			
65		**Write a Python function that takes a number as a parameter and checks whether the numberis prime or not**			
66		**Write a Python program to access a function inside a function**			
67		**Write a Python function that checks whether a passed string is a palindrome or not.**			

68		**Write a python function to create and print a list where the values are the squares of numbers between 1 and 30**			
69		**Write a Python program to print the even numbers from a given list**			
70		**Write a program to create a class student with data member name. roll no. Semester. and display the data using object**			
71		**Write a program to demonstrate the use of _init_()**			
72		**Write a program for Inheritence.a) single level b) Multiple c) Multilevel d) Hybrid e) Hierarchical**			
73		**Write a program to demonstrate overriding**			

1.1 Python

Question 1

Explain Installation of Python.

Solution:

```
Windows:

 Download Python Installer: Visit the official Python website,
 download the installer, and choose the appropriate version (32-bit or 64-bit).

 Run Installer: Double-click the installer file and follow the prompts.
 Make sure to check the box that says "Add Python to PATH".

 Complete Installation: Follow the installation wizard's instructions.
 Python will be installed in C:\PythonXX\ (where XX is the version number).

 Verify Installation: Open a command prompt, type python --version, and press Enter.
  You should see the installed Python version.

macOS:

 Install Homebrew (optional): If you prefer Homebrew,
 open Terminal and run the Homebrew installation command.

 Install Python: Use Homebrew (brew install python) or
 download the macOS installer from python.org and run it.

 Verify Installation: Open Terminal and type
 python3 --version to verify the installation.

Linux (Ubuntu/Debian):

 Update Package Lists: Open Terminal and run sudo apt update.

 Install Python: Run sudo apt install python3 in Terminal to install Python 3.

 Verify Installation: After installation, type python3 --version in Terminal to verify.
```

Question 2

To print different types of data types with multiline and single line comments.

Solution:

```python
# Single-line comment: Printing different data types
print("Data types:")

# Integer data type
num = 10
print("Integer:", num)
```

```
    # Float data type
    float_num = 3.14
    print("Float:", float_num)

    # String data type
    string = "Hello, World!"
    print("String:", string)

    # Boolean data type
    bool_value = True
    print("Boolean:", bool_value)

    """
    Multiline comment:
    List data type
    """
    my_list = [1, 2, 3, 4, 5]
    print("List:", my_list)

    """
    Multiline comment:
    Tuple data type
    """
    my_tuple = (1, 2, 3, 4, 5)
    print("Tuple:", my_tuple)

    """
    Multiline comment:
    Dictionary data type
    """
    my_dict = {'a': 1, 'b': 2, 'c': 3}
    print("Dictionary:", my_dict)

    """
    Multiline comment:
    Set data type
    """
    my_set = {1, 2, 3, 4, 5}
    print("Set:", my_set)

    print("\n")
    print("Ankit Baingane")
    print("0827CI211028")
```

Output:

```
Data types:
Integer: 10
Float: 3.14
String: Hello, World!
Boolean: True
List: [1, 2, 3, 4, 5]
Tuple: (1, 2, 3, 4, 5)
Dictionary: {'a': 1, 'b': 2, 'c': 3}
Set: {1, 2, 3, 4, 5}
```

Ankit Baingane
0827CI211028

Question 3

To print the largest number of three numbers using if else and elif.

Solution:

```
# Three numbers
num1 = 10
num2 = 20
num3 = 15

# Check which number is largest
if num1 >= num2 and num1 >= num3:
    largest_number = num1
elif num2 >= num1 and num2 >= num3:
    largest_number = num2
else:
    largest_number = num3

# Print the result
print("The largest number among", num1, ",",
    num2, ", and", num3, "is:", largest_number)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

The largest number among 10 , 20 , and 15 is: 20


Ankit Baingane
0827CI211028
```

Question 4

Write a program to print number I-IO using a while loop.

Solution:

```
# Initialize the starting number
number = 1

# Print numbers from 1 to 10 using a while loop
while number <= 10:
    print(number)
    number += 1

print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

Output:

```
1
2
3
4
5
6
7
8
9
10


Ankit Baingane
0827CI211028
```

Question 5

Write a program to check prime numbers between 1-100 using a for loop.

Solution:

```
# Iterate through numbers from 1 to 100
for num in range(1, 101):
    # Check if the number is greater than 1
    if num > 1:
        # Check for factors
        for i in range(2, int(num ** 0.5) + 1):
            if (num % i) == 0:
                break
        else:
            print(num, "is a prime number.")

print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

Output:

```
2 is a prime number.
3 is a prime number.
5 is a prime number.
7 is a prime number.
11 is a prime number.
13 is a prime number.
17 is a prime number.
19 is a prime number.
23 is a prime number.
29 is a prime number.
31 is a prime number.
37 is a prime number.
41 is a prime number.
43 is a prime number.
```

```
47 is a prime number.
53 is a prime number.
59 is a prime number.
61 is a prime number.
67 is a prime number.
71 is a prime number.
73 is a prime number.
79 is a prime number.
83 is a prime number.
89 is a prime number.
97 is a prime number.


Ankit Baingane
0827CI211028
```

Question 6

Write a program to concatenate two strings using + operator

Solution:

```
# Two strings to concatenate
string1 = "Hello world"
string2 = ".This is always constant"

# Concatenate the strings using the + operator
concatenated_string = string1 + " " + string2

# Print the concatenated string
print("Concatenated string:", concatenated_string)

print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

Concatenated string: Hello world .This is always constant


Ankit Baingane
0827CI211028
```

Question 7

Write a program to reverse a string using slicing

Solution:

```
# Input string
input_string = "Hello, World!"
```

```
# Reverse the string using slicing
reversed_string = input_string[::-1]

# Print the reversed string
print("Original String:", input_string)
print("Reversed String:", reversed_string)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

Original String: Hello, World!
Reversed String: !dlroW ,olleH


Ankit Baingane
0827CI211028
```

Question 8

Write a program to perform different methods of string like: len(atleast 5)

Solution:

```
# Input string
input_string = "Hello, World!"

# Length of the string
length = len(input_string)
print("Length of the string:", length)

# Convert string to uppercase
uppercase_string = input_string.upper()
print("Uppercase string:", uppercase_string)

# Convert string to lowercase
lowercase_string = input_string.lower()
print("Lowercase string:", lowercase_string)

# Count occurrences of a substring
substring = "o"
count = input_string.count(substring)
print("Number of occurrences of 'o':", count)

# Replace substring
old_substring = "World"
new_substring = "Python"
replaced_string = input_string.replace(old_substring, new_substring)
print("String after replacement:", replaced_string)

# Check if string starts with a substring
substring_to_check = "Hello"
starts_with = input_string.startswith(substring_to_check)
print(f"String starts with '{substring_to_check}':", starts_with)
```

```python
    # Check if string ends with a substring
    substring_to_check = "!"
    ends_with = input_string.endswith(substring_to_check)
    print(f"String ends with '{substring_to_check}':", ends_with)

    print("\n")
    print("Ankit Baingane")
    print("0827CI211028")
```

Output:

```
Length of the string: 13
Uppercase string: HELLO, WORLD!
Lowercase string: hello, world!
Number of occurrences of 'o': 2
String after replacement: Hello, Python!
String starts with 'Hello': True
String ends with '!': True


Ankit Baingane
0827CI211028
```

Question 9

Write a program to traverse all the characters of a string using a for loop.

Solution:

```python
    # Input string
    input_string = "Hello, World!"

    # Traverse all characters of the string using a for loop
    print("Traversing all characters of the string using a for loop:")
    for char in input_string:
        print(char)

    print("\n")
    print("Ankit Baingane")
    print("0827CI211028")
```

Output:

```
Traversing all characters of the string using a for loop:
H
e
l
l
o
,

W
o
r
```

```
l
d
!


Ankit Baingane
0827CI211028
```

Question 10

Write a program to Print abecederian series

Solution:

```python
# Function to check if a word is abecedarian
def is_abecedarian(word):
    """
    Function to check if a word is abecedarian.

    Parameters:
    word (str): The word to check.

    Returns:
    bool: True if the word is abecedarian, False otherwise.
    """
    # Iterate through each character in the word
    for i in range(len(word) - 1):
        # Check if the current character is greater than the next character
        if word[i] > word[i + 1]:
            return False
    return True

# Main program to print abecedarian series
def print_abecedarian_series():
    """
    Function to print abecedarian series.
    """
    # Starting letter
    start_letter = 'a'

    # Loop to generate and print abecedarian words
    while start_letter <= 'z':
        print(start_letter)
        start_letter = chr(ord(start_letter) + 1)

# Print the abecedarian series
print("Abecedarian series:")
print_abecedarian_series()
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

Output:

```
Abecedarian series:
a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
q
r
s
t
u
v
w
x
...


Ankit Baingane
0827CI211028
```

Question 11

Write a program to check whether a string is present in another string or not.

Solution:

```python
def check_substring(main_string, substring):
    """
    Function to check whether a substring is present in a main string.

    Parameters:
    main_string (str): The main string.
    substring (str): The substring to check.

    Returns:
    bool: True if substring is present in main string, False otherwise.
    """
    return substring in main_string

# Example usage:
main_string = "Hello, World!"
substring1 = "Hello"
substring2 = "Python"
```

```
# Check if substrings are present in the main string
print(f"'\{substring1\}' is present in '\{main_string\}':", check_substring(main_string,
print(f"'\{substring2\}' is present in '\{main_string\}':", check_substring(main_string,
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

Output:

```
'Hello' is present in 'Hello, World!': True
'Python' is present in 'Hello, World!': False


Ankit Baingane
0827CI211028
```

Question 12

Write a program to print
A
ABC
ABCD

Solution:

```
# Number of rows for the pattern
num_rows = 4

# Outer loop to iterate over each rowi
for i in range(1, num_rows + 1):
    # Inner loop to print characters from 'A' to 'A + i - 1'
    for j in range(ord('A'), ord('A') + i):
        print(chr(j), end="")
    print()  # Move to the next line after printing each row
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

Output:

```
A
AB
ABC
ABCD


Ankit Baingane
0827CI211028
```

Question 13

Write a program to Create a list with different data types.

Solution:

```
# Create a list with different data types
my_list = [1, "Hello", 3.14, True, [1, 2, 3]]

# Print the list
print("List with different data types:", my_list)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

List with different data types: [1, 'Hello', 3.14, True, [1, 2, 3]]


Ankit Baingane
0827CI211028
```

Question 14

Write a program to take user input in a list from eval.

Solution:

```
    # Take user input as a string
    user_input_str = input("Enter elements of the list separated by commas: ")

    # Convert the user input string to a list using eval()
    try:
        user_list = eval(user_input_str)
        if not isinstance(user_list, list):
            raise ValueError("Input is not a valid list.")
    except Exception as e:
        print("Error:", e)
    else:
        print("User input list:", user_list)
    print("\n")
    print("Ankit Baingane")
    print("0827CI211028")
```

```
Output:

Enter elements of the list separated by commas: 12,18,24,7,10
Error: Input is not a valid list.


Ankit Baingane
0827CI211028
```

Question 15

Write a program to append five elements in a list by using for loop and append.

Solution:

```
# Initialize an empty list
my_list = []

# Append five elements to the list using a for loop and append() method
for i in range(1, 6):
    my_list.append(i)

# Print the list
print("List after appending five elements:", my_list)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

List after appending five elements: [1, 2, 3, 4, 5]


Ankit Baingane
0827CI211028
```

Question 16

Write a program to sort a list in both ascending and descending order.

Solution:

```
# Original list
my_list = [5, 2, 8, 1, 9]

# Sort the list in ascending order using sort() method
my_list.sort()
print("List sorted in ascending order:", my_list)

# Sort the list in descending order using sorted() function
descending_list = sorted(my_list, reverse=True)
print("List sorted in descending order:", descending_list)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

List sorted in ascending order: [1, 2, 5, 8, 9]
List sorted in descending order: [9, 8, 5, 2, 1]


Ankit Baingane
0827CI211028
```

Question 17

Write a program to count the occurrences of an element in a list.

Solution:

```
# Original list
my_list = [1, 2, 3, 4, 2, 3, 2, 5, 2]

# Element to count occurrences
element_to_count = 2

# Count occurrences of the element in the list
occurrences = my_list.count(element_to_count)

# Print the result
print(f"The element {element_to_count} occurs {occurrences} times in the list.")
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
The element 2 occurs 4 times in the list.


Ankit Baingane
0827CI211028
```

Question 18

Write a program to find the index of an element in the list.

Solution:

```
# Original list
my_list = [10, 20, 30, 40, 50]

# Element to find index
element_to_find = 30

try:
    # Find the index of the element in the list
    index = my_list.index(element_to_find)
    print(f"The index of {element_to_find} in the list is:", index)
except ValueError:
    print(f"The element {element_to_find} is not present in the list.")
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

The index of 30 in the list is: 2


Ankit Baingane
0827CI211028
```

Question 19

Write a program to swap the first and last element of a list.

Solution:

```
# Original list
my_list = [10, 20, 30, 40, 50]
print("List before swaping : ",my_list)
# Swap the first and last elements of the list
if len(my_list) >= 2:
    my_list[0], my_list[-1] = my_list[-1], my_list[0]

# Print the modified list
print("List after swapping first and last elements:", my_list)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

List before swaping :  [10, 20, 30, 40, 50]
List after swapping first and last elements: [50, 20, 30, 40, 10]


Ankit Baingane
0827CI211028
```

Question 20

Write a program to swap two no in list with given position.

Solution:

```
def swap_elements(lst, pos1, pos2):
    """
    Function to swap two elements in a list at given positions.

    Parameters:
    lst (list): The input list.
    pos1 (int): The position of the first element to swap.
    pos2 (int): The position of the second element to swap.

    Returns:
    list: The list with elements swapped.
    """
    # Check if positions are within the range of the list
    if 0 <= pos1 < len(lst) and 0 <= pos2 < len(lst):
        # Swap the elements
        lst[pos1], lst[pos2] = lst[pos2], lst[pos1]
    else:
        print("Positions are out of range. Swapping not possible.")
    return lst

# Original list
```

```
my_list = [10, 20, 30, 40, 50]

# Positions to swap
position1 = 1
position2 = 3

# Swap elements at given positions
modified_list = swap_elements(my_list, position1, position2)

# Print the modified list
print("List after swapping elements at positions", position1, "and", position2, ":", mo
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

List after swapping elements at positions 1 and 3 : [10, 40, 30, 20, 50]


Ankit Baingane
0827CI211028
```

Question 21

Write a program to check whether a no is in the list or not sublist even and odd.

Solution:

```
# Original list
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9]

# Number to check
number_to_check = 5

# Check if the number is present in the list
if number_to_check in my_list:
    print(f"The number {number_to_check} is present in the list.")
else:
    print(f"The number {number_to_check} is not present in the list.")

# Separate the list into two sublists containing even and odd numbers
even_numbers = []
odd_numbers = []

for num in my_list:
    if num % 2 == 0:
        even_numbers.append(num)
    else:
        odd_numbers.append(num)

# Print the sublists
print("Even numbers in the list:", even_numbers)
```

```
print("Odd numbers in the list:", odd_numbers)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

The number 5 is present in the list.
Even numbers in the list: [2, 4, 6, 8]
Odd numbers in the list: [1, 3, 5, 7, 9]


Ankit Baingane
0827CI211028
```

Question 22

Write a program to demonstrate the difference between remove and pop method in a list.

Solution:

```
# Original list
my_list = [1, 2, 3, 4, 5]

# Demonstrate remove() method
removed_element = 3
print("Original list before using remove():", my_list)
my_list.remove(removed_element)
print("List after using remove():", my_list)
print("Removed element:", removed_element)

# Demonstrate pop() method
index_to_pop = 2
popped_element = my_list.pop(index_to_pop)
print("Original list before using pop():", my_list)
print("Popped element at index", index_to_pop, ":", popped_element)
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

Original list before using remove(): [1, 2, 3, 4, 5]
List after using remove(): [1, 2, 4, 5]
Removed element: 3
Original list before using pop(): [1, 2, 5]
Popped element at index 2 : 4


Ankit Baingane
0827CI211028
```

Question 23

Write a program to insert an element into a list at a position given by the user.

Solution:

```
# Original list
my_list = [1, 2, 3, 4, 5]

# Get element and position from the user
element_to_insert = int(input("Enter the element to insert: "))
position = int(input("Enter the position to insert the element: "))

# Check if the position is within the range of the list
if 0 <= position <= len(my_list):
    # Insert the element at the specified position
    my_list.insert(position, element_to_insert)
    print("List after inserting the element:", my_list)
else:
    print("Invalid position. Element cannot be inserted.")
print("\n")
print("Ankit Baingane")
print("0827CI211028")
```

```
Output:

Enter the element to insert: 12
Enter the position to insert the element: 2
List after inserting the element: [1, 2, 12, 3, 4, 5]


Ankit Baingane
0827CI211028
```

Question 24

Explain the difference between the list, tuple , set and dictionary ?

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# List
my_list = [1, 2, 3, 4, 5]

print(my_list)
# Lists maintain order
for item in my_list:
    print(item)

# Tuple
my_tuple = (1, 2, 3, 4, 5)

# Tuples are immutable - attempting to modify will raise an error
```

```
    for item in my_tuple:
        print(item)

    # Set
    my_set = {1, 2, 3, 4, 5}

    # Sets are mutable
    my_set.add(6)
    print(my_set)  # Output: {1, 2, 3, 4, 5, 6}

    # Sets do not maintain order
    for item in my_set:
       print(item)

    # Dictionary
    my_dict = {'name': 'John', 'age': 30, 'city': 'New York'}

    # Dictionaries are mutable
    my_dict['age'] = 35
    print(my_dict)  # Output: {'name': 'John', 'age': 35, 'city': 'New York'}

    # Dictionaries do not maintain order (prior to Python 3.7)
    for key, value in my_dict.items():
        print(key, ':', value)
```

Output:

```
Ankit Baingane
0827CI211028


[10, 2, 3, 4, 5]

10
2
3
4
5

1
2
3
4
5

{1,2,3,4,5,6}

1
2
3
4
5
6

{'name': 'John', 'age': 35, 'city': 'New York'}
```

```
name : John
age : 35
city : New York
```

Question 25

Write a program to Counts the number of occurrences of item 50 from a tuple.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def count_occurrences(tuple, item):
count = 0
for element in tuple:
    if element == item:
        count += 1
return count

# Example tuple
example_tuple = (10, 20, 30, 40, 50, 50, 50, 60, 70)

# Count occurrences of item 50
occurrences = count_occurrences(example_tuple, 50)
print("Number of occurrences of item 50:", occurrences)
```

```
Output:

Ankit Baingane
0827CI211028
Number of occurrences of item 50: 3
```

Question 26

Write a program to Check if all items in the tuple are the same

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def all_same(tuple):
# Check if all elements are the same
return all(element == tuple[0] for element in tuple)

# Example tuples
tuple1 = (10, 10, 10, 10)
tuple2 = (10, 20, 10, 10)

# Check if all items in tuple1 are the same
if all_same(tuple1):
    print("All items in tuple1 are the same")
else:
```

```
        print("Not all items in tuple1 are the same")

    # Check if all items in tuple2 are the same
    if all_same(tuple2):
        print("All items in tuple2 are the same")
    else:
        print("Not all items in tuple2 are the same")
```

```
Output:

Ankit Baingane
0827CI211028
All items in tuple1 are the same
Not all items in tuple2 are the same
```

Question 27

Write a Program to create a tuple which is having integer,float,list and tuple as its elements and print an element present in list of these tuple

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Define the tuple with different types of elements
mixed_tuple = (10, 3.14, [1, 2, 3], ('a', 'b', 'c'))

# Function to print an element from a list within the tuple
def print_list_element(tuple):
    # Get the list from the tuple
    list_element = tuple[2]
    # Print an element from the list
    print("Element from the list within the tuple:", list_element[1])

# Call the function to print an element from the list within the tuple
print_list_element(mixed_tuple)
```

```
Output:

Ankit Baingane
0827CI211028
Element from the list within the tuple: 2
```

Question 28

Write a program to create a tuple with the name of 10 cities of India Check whether a City is present in the tuple or not

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def city_present(city, cities_tuple):
# Check if the city is present in the tuple
return city in cities_tuple

# Create a tuple with the names of 10 cities of India
indian_cities = ('Delhi', 'Mumbai', 'Bangalore',
                 'Kolkata','Chennai', 'Hyderabad',
                  'Pune', 'Ahmedabad', 'Jaipur',
                  'Lucknow')

# Input city to check
city_to_check = input("Enter the name of the city to check: ")

# Check if the city is present in the tuple
if city_present(city_to_check, indian_cities):
    print(city_to_check, "is present in the tuple of Indian cities.")
else:
    print(city_to_check, "is not present in the tuple of Indian cities.")
```

Output:

```
Ankit Baingane
0827CI211028
Enter the name of the city to check: Kolkata
Kolkata is present in the tuple of Indian cities.
```

Question 29

Write a program to get the number of occurrence of a word in tuple.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def count_word_occurrences(word, tuple_of_strings):
count = 0
for string in tuple_of_strings:
    # Split the string into words and count occurrences of the target word
    count += string.count(word)
return count

# Example tuple of strings
tuple_of_strings = ("apple banana", "banana orange",
                    "banana apple", "orange mango banana")

# Word to count occurrences of
word_to_count = input("Enter the word to count occurrences: ")

# Count occurrences of the word in the tuple
occurrences = count_word_occurrences(word_to_count, tuple_of_strings)
```

```
print("Number of occurrences of",
      word_to_count, "in the tuple:",
       occurrences)
```

Output:

```
Ankit Baingane
0827CI211028
Enter the word to count occurrences: apple
Number of occurrences of apple in the tuple: 2
```

Question 30

Write a program to get the index of a word in tuple

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def get_word_index(word, tuple_of_strings):
indexes = []
for i, string in enumerate(tuple_of_strings):
    # Split the string into words and check if the word is present
    if word in string:
        indexes.append(i)
return indexes

# Example tuple of strings
tuple_of_strings = ("apple banana", "banana orange",
            "banana apple", "orange mango banana")

# Word to get the index of
word_to_find = input("Enter the word to find its index: ")

# Get the index of the word in the tuple
word_indexes = get_word_index(word_to_find, tuple_of_strings)

if word_indexes:
    print("Indexes of", word_to_find, "in the tuple:", word_indexes)
else:
    print(word_to_find, "not found in the tuple.")
```

Output:

```
Ankit Baingane
0827CI211028
Enter the word to find its index: the
the not found in the tuple.
```

Question 31

Write a program to create four tuples viz roll no. name. CGPA and SGPA of students. Print individual student's details using these 4 tuples.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")


# Function to print individual student's details
def print_student_details(roll_no, name, cgpa, sgpa):
    print("Roll No:", roll_no)
    print("Name:", name)
    print("CGPA:", cgpa)
    print("SGPA:", sgpa)
    print()

# Create tuples for student details
roll_numbers = (101, 102, 103, 104)
names = ("John Doe", "Jane Smith", "Alice Johnson", "Bob Brown")
cgpa = (3.8, 3.9, 3.7, 4.0)
sgpa = (4.0, 3.9, 3.8, 4.0)

# Print individual student's details
for i in range(len(roll_numbers)):
    print("Student", i+1, "details:")
    print_student_details(roll_numbers[i], names[i], cgpa[i], sgpa[i])
print("Ritesh Telkar")
print("0827CI211158")
```

```
Output:

Ankit Baingane
0827CI211028
Student 1 details:
Roll No: 101
Name: John Doe
CGPA: 3.8
SGPA: 4.0

Student 2 details:
Roll No: 102
Name: Jane Smith
CGPA: 3.9
SGPA: 3.9

Student 3 details:
Roll No: 103
Name: Alice Johnson
CGPA: 3.7
SGPA: 3.8

Student 4 details:
Roll No: 104
Name: Bob Brown
```

```
CGPA: 4.0
SGPA: 4.0
```

Question 32

write a Python program to create a set.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Create a set
my_set = {1, 2, 3, 4, 5}

# Print the set
print("Set:", my_set)
```

```
Output:

Ankit Baingane
0827CI211028
Set: {1, 2, 3, 4, 5}
```

Question 33

Write a Python program to iterate over sets.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Define a set
my_set = {1, 2, 3, 4, 5}

# Iterate over the set and print each element
print("Elements of the set:")
for element in my_set:
    print(element)
```

```
Output:

Ankit Baingane
0827CI211028
Elements of the set:
1
2
3
4
5
```

Question 34

Write a Python program to add member(s) to a set.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Define a set
my_set = {1, 2, 3, 4, 5}

# Print the initial set
print("Initial set:", my_set)

# Add a single member to the set
my_set.add(6)
print("Set after adding a single member:", my_set)

# Add multiple members to the set using update() method
my_set.update([7, 8, 9])
print("Set after adding multiple members:", my_set)
```

```
Output:

Ankit Baingane
0827CI211028
Initial set: {1, 2, 3, 4, 5}
Set after adding a single member: {1, 2, 3, 4, 5, 6}
Set after adding multiple members: {1, 2, 3, 4, 5, 6, 7, 8, 9}
```

Question 35

Write a Python program to remove Item(s) from a given set.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

my_set = {1, 2, 3, 4, 5}

# Print the initial set
print("Initial set:", my_set)

# Remove a single item from the set using discard() method
my_set.discard(3)
print("Set after removing a single item:", my_set)

# Remove multiple items from the set using discard() method
items_to_remove = {1, 4}
my_set.difference_update(items_to_remove)
print("Set after removing multiple items:", my_set)
```

```
Output:

Ankit Baingane
0827CI211028
Initial set: {1, 2, 3, 4, 5}
Set after removing a single item: {1, 2, 4, 5}
Set after removing multiple items: {2, 5}
```

Question 36

Write a Python program to remove an item from a set if it is

Solution:

```python
print("Ankit Baingane")
print("0827CI211028")

def remove_item(set_name, item):
if item in set_name:
    set_name.remove(item)
    print(f"Item '{item}' removed from the set")
else:
    print(f"Item '{item}' is not present in the set")

# Define a set
my_set = {1, 2, 3, 4, 5}

# Print the initial set
print("Initial set:", my_set)

# Remove an item from the set
item_to_remove = int(input("Enter the item to remove: "))
remove_item(my_set, item_to_remove)

# Print the set after removal
print("Set after removal:", my_set)
```

```
Output:

Ankit Baingane
0827CI211028
Initial set: {1, 2, 3, 4, 5}
Enter the item to remove: 3
Item '3' removed from the set
Set after removal: {1, 2, 4, 5}
```

Question 37

Write a Python program to create an intersection of sets.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Define two sets
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}

# Find the intersection of the sets using the intersection() method
intersection_set = set1.intersection(set2)

# Print the intersection set
print("Intersection of set1 and set2:", intersection_set)
```

Output:

```
Ankit Baingane
0827CI211028
Intersection of set1 and set2: {4, 5}
```

Question 38

Write a Python program to create a union of sets.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Define two sets
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}

# Find the union of the sets using the union() method
union_set = set1.union(set2)

# Print the union set
print("Union of set1 and set2:", union_set)
```

Output:

```
Ankit Baingane
0827CI211028
Union of set1 and set2: {1, 2, 3, 4, 5, 6, 7, 8}
```

Question 39

Write a Python program to create set difference

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Define two sets
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}

# Find the set difference using the difference() method
difference_set = set1.difference(set2)

# Print the set difference
print("Set difference (set1 - set2):", difference_set)
```

Output:

```
Ankit Baingane
0827CI211028
Set difference (set1 - set2): {1, 2, 3}
```

Question 40

Write a Python program to create a symmetric difference.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Define two sets
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}

# Find the symmetric difference using the symmetric_difference() method
symmetric_difference_set = set1.symmetric_difference(set2)

# Print the symmetric difference
print("Symmetric difference of set1 and set2:", symmetric_difference_set)
```

Output:

```
Ankit Baingane
0827CI211028
Symmetric difference of set1 and set2: {1, 2, 3, 6, 7, 8}
```

Question 41

write a Python program to check if a set is a subset of another set.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
```

```
def is_subset(set1, set2):
# Check if set1 is a subset of set2
return set1.issubset(set2)

# Example usage:
set1 = {1, 2, 3}
set2 = {1, 2, 3, 4, 5}

if is_subset(set1, set2):
    print("set1 is a subset of set2")
else:
    print("set1 is not a subset of set2")
```

Output:

```
Ankit Baingane
0827CI211028
set1 is a subset of set2
```

Question 42

Write a Python program to remove all elements from a given set

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def remove_all_elements(some_set):
some_set.clear()  # Clears all elements from the set

# Example usage:
my_set = {1, 2, 3, 4, 5}
print("Before removing elements:", my_set)

remove_all_elements(my_set)
print("After removing elements:", my_set)
```

Output:

```
Ankit Baingane
0827CI211028
Before removing elements: {1, 2, 3, 4, 5}
After removing elements: set()
```

Question 43

Write a Python program to find the maximum and minimum values in a set

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def find_max_min(some_set):
if len(some_set) == 0:
    print("The set is empty.")
else:
    max_value = max(some_set)
    min_value = min(some_set)
    return max_value, min_value

# Example usage:
my_set = {5, 3, 9, 2, 8, 1}

result = find_max_min(my_set)
if result:
    max_val, min_val = result
    print("Maximum value:", max_val)
    print("Minimum value:", min_val)
```

```
Output:

Ankit Baingane
0827CI211028
Maximum value: 9
Minimum value: 1
```

Question 44

write a Python program to find length of a set

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def find_set_length(some_set):
return len(some_set)

# Example usage:
my_set = {1, 2, 3, 4, 5}
set_length = find_set_length(my_set)
print("Length of the set:", set_length)
```

```
Output:

Ankit Baingane
0827CI211028
Length of the set: 5
```

Question 45

Write a Python program to check if a given value is present in a set or not.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def check_value_in_set(some_set, value):
return value in some_set

# Example usage:
my_set = {1, 2, 3, 4, 5}
value_to_check = 3

if check_value_in_set(my_set, value_to_check):
    print("The value", value_to_check, "is present in the set.")
else:
    print("The value", value_to_check, "is not present in the set.")
```

```
Output:

Ankit Baingane
0827CI211028
The value 3 is present in the set
```

Question 46

Write a Python program to check if two given sets have no elements in common.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def no_common_elements(set1, set2):
return set1.isdisjoint(set2)

# Example usage:
set1 = {1, 2, 3}
set2 = {4, 5, 6}

if no_common_elements(set1, set2):
    print("The sets have no common elements.")
else:
    print("The sets have common elements.")
```

```
Output:

Ankit Baingane
0827CI211028
The sets have no common elements.
```

Question 47

Write a Python program to create a Dictionary of student information. take roll no as key

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def create_student_dictionary():
num_students = int(input("Enter the number of students: "))
student_dict = {}

for _ in range(num_students):
    roll_no = input("Enter roll number: ")
    name = input("Enter name: ")
    age = int(input("Enter age: "))
    grade = input("Enter grade: ")
    student_dict[roll_no] = {"Name": name, "Age": age, "Grade": grade}

return student_dict

def main():
    student_info = create_student_dictionary()
    print("\nStudent Information:")
    for roll_no, info in student_info.items():
        print("for student ")
        print(f"Roll No: {roll_no}")
        print(f"Name: {info['Name']}")
        print(f"Age: {info['Age']}")
        print(f"Grade: {info['Grade']}")
        print()

if __name__ == "__main__":
    main()
```

```
Output:

Ankit Baingane
0827CI211028
Enter the number of students: 3
Enter roll number: 101
Enter name: abhay
Enter age: 23
Enter grade: A
Enter roll number: 102
Enter name: akaay
Enter age: 22
Enter grade: C
Enter roll number: 103
Enter name: asif
Enter age: 21
Enter grade: B

Student Information:
Roll No: 101
Name: abhay
Age: 23
```

```
Grade: A

Roll No: 102
Name: akaay
Age: 22
Grade: C

Roll No: 103
Name: asif
Age: 21
Grade: B
```

Question 48

Write a Python progarm to delete a key from the dictionary

Solution:

```python
print("Ankit Baingane")
print("0827CI211028")

def delete_key_from_dict(student_dict, roll_no):
if roll_no in student_dict:
    del student_dict[roll_no]
    print(f"Key '{roll_no}' deleted successfully.")
else:
    print(f"Key '{roll_no}' not found in the dictionary.")

def main():
    # Example student dictionary
    student_dict = {
        "001": {"Name": "Alice", "Age": 18, "Grade": "A"},
        "002": {"Name": "Bob", "Age": 19, "Grade": "B"},
        "003": {"Name": "Charlie", "Age": 20, "Grade": "C"}
    }

    print("Original Dictionary:")
    print(student_dict)

    roll_no_to_delete = input("\nEnter the roll number to delete: ")
    delete_key_from_dict(student_dict, roll_no_to_delete)

    print("\nUpdated Dictionary:")
    print(student_dict)

if __name__ == "__main__":
    main()
```

```
Output:

Ankit Baingane
0827CI211028
Original Dictionary:
```

```
{'001': {'Name': 'Alice', 'Age': 18, 'Grade': 'A'},
 '002': {'Name': 'Bob', 'Age': 19, 'Grade': 'B'},
 '003': {'Name': 'Charlie', 'Age': 20, 'Grade': 'C'}}

Enter the roll number to delete: 003
Key '003' deleted successfully.

Updated Dictionary:
{'001': {'Name': 'Alice', 'Age': 18, 'Grade': 'A'},
'002': {'Name': 'Bob', 'Age': 19, 'Grade': 'B'}}
```

Question 49

Write a Python program to add or update the data.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

def add_or_update_data(dictionary, key, value):
"""
Function to add or update data in a dictionary.

Parameters:
dictionary (dict): The dictionary to modify.
key: The key to add/update in the dictionary.
value: The value corresponding to the key.

Returns:
None
"""
dictionary[key] = value


my_dictionary = {'a': 1, 'b': 2, 'c': 3}

print("Original Dictionary:", my_dictionary)


add_or_update_data(my_dictionary, 'd', 4)
print("Dictionary after adding 'd':", my_dictionary)


add_or_update_data(my_dictionary, 'b', 5)
print("Dictionary after updating 'b':", my_dictionary)
```

```
Output:

Ankit Baingane
0827CI211028
Original Dictionary: {'a': 1, 'b': 2, 'c': 3}
Dictionary after adding 'd': {'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

```
Dictionary after updating 'b': {'a': 1, 'b': 5, 'c': 3, 'd': 4}
```

Question 50

Write a Python script to concatenate the following dictionaries to create a new one sample Dictionary:dic1=1:10,2:20, dic2=3:30,4:40

Solution:

```python
print("Ankit Baingane")
print("0827CI211028")
def concatenate_dicts(*dicts):
"""
Concatenate multiple dictionaries into a new one.

Parameters:
*dicts: Variable number of dictionaries to concatenate.

Returns:
dict: The concatenated dictionary.
"""
concatenated_dict = {}
for d in dicts:
    concatenated_dict.update(d)
return concatenated_dict

# Sample dictionaries
dic1 = {1: 10, 2: 20}
dic2 = {3: 30, 4: 40}

# Concatenate dictionaries
concatenated_dict = concatenate_dicts(dic1, dic2)

# Output the concatenated dictionary
print("Concatenated Dictionary:", concatenated_dict)
```

```
Output:

Ankit Baingane
0827CI211028
Concatenated Dictionary: {1: 10, 2: 20, 3: 30, 4: 40}
```

Question 51

Write a Python script to check whether a given key already exists in a dictionary

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def check_key_existence(dictionary, key):
"""
Check whether a given key exists in a dictionary.

Parameters:
dictionary (dict): The dictionary to check.
key: The key to check for existence.

Returns:
bool: True if the key exists, False otherwise.
"""
return key in dictionary

# Sample dictionary
my_dict = {'a': 1, 'b': 2, 'c': 3}

# Key to check
key_to_check = 'b'

# Check if the key exists in the dictionary
if check_key_existence(my_dict, key_to_check):
    print(f"The key '{key_to_check}' exists in the dictionary.")
else:
    print(f"The key '{key_to_check}' does not exist in the dictionary.")
```

Output:

```
Ankit Baingane
0827CI211028
The key 'b' exists in the dictionary.
```

Question 52

Write a Python program to iterate over dictionaries using for loops

Solution:

```
print("Ankit Baingane")
print("0827CI211028")

# Sample dictionary
my_dict = {'a': 1, 'b': 2, 'c': 3}

# Iterate over keys
print("Iterating over keys:")
for key in my_dict:
    print(key)

# Iterate over values
print("\nIterating over values:")
for value in my_dict.values():
    print(value)
```

```python
# Iterate over key-value pairs
print("\nIterating over key-value pairs:")
for key, value in my_dict.items():
    print(key, "->", value)
```

```
Output:

Ankit Baingane
0827CI211028
Iterating over keys:
a
b
c

Iterating over values:
1
2
3

Iterating over key-value pairs:
a -> 1
b -> 2
c -> 3
```

Question 53

Write a Python script to generate and print a dictionary that contains a number (between I and n) in the form (x, x*x).Sample dictionary (n=5): Expected output: 1:1,2:4,3:9,4:16,5:25

Solution:

```python
print("Ankit Baingane")
print("0827CI211028")

def generate_squared_dict(n):
"""
Generate a dictionary containing numbers and their squares from 1 to n.

Parameters:
n (int): The maximum number to include in the dictionary.

Returns:
dict: The generated dictionary.
"""
squared_dict = {}
for x in range(1, n+1):
    squared_dict[x] = x * x
return squared_dict
```

```
    # Sample value of n
    n = 8

    # Generate the dictionary
    result_dict = generate_squared_dict(n)

    # Print the generated dictionary
    print("Generated Dictionary:", result_dict)
```

Output:

```
Ankit Baingane
0827CI211028
Generated Dictionary: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}
```

Question 54

Write a python function to calculate Sum of two variables.

Solution:

```
    print("Ankit Baingane")
    print("0827CI211028")
    def sum_of_two_variables(a, b):
    """
    Calculate the sum of two variables.

    Parameters:
    a: The first variable.
    b: The second variable.

    Returns:
    The sum of a and b.
    """
    return a + b

    # Example usage:
    result = sum_of_two_variables(5, 7)
    print("Sum of the two variables:", result)
```

Output:

```
Ankit Baingane
0827CI211028
Sum of the two variables: 12
```

Question 55

Write a Python function to show the use of Default Parameter

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def greet(name, greeting="Hello"):
"""
Function to greet a person with a specified greeting.

Parameters:
name: The name of the person to greet.
greeting (optional):
The greeting to use. Defaults to "Hello" if not specified.

Returns:
str: The greeting message.
"""
return f"{greeting}, {name}!"

# Example usage:
print(greet("Ram"))  # Uses the default greeting "Hello"
```

```
Output:

Ankit Baingane
0827CI211028
Hello, Ram!
Good morning, Siya!
```

Question 56

Write a function to find the maximum of three numbers

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def find_maximum(a, b, c):
"""
Function to find the maximum of three numbers.

Parameters:
a: The first number.
b: The second number.
c: The third number.

Returns:
The maximum of the three numbers.
"""
if a >= b:
    if a >= c:
```

```
            return a
        else:
            return c
    else:
        if b >= c:
            return b
        else:
            return c

    # Example usage:
    result = find_maximum(10, 5, 8)
    print("Maximum of the three numbers:", result)
```

```
Output:

Ankit Baingane
0827CI211028
Maximum of the three numbers: 10
```

Question 57

Write a python function to use arbitrary argument

Solution:

```
    print("Ankit Baingane")
    print("0827CI211028")
    def print_arguments(*args):
    """
    Function to print arbitrary arguments passed to it.

    Parameters:
    *args: Arbitrary number of arguments.

    Returns:
    None
    """
    for arg in args:
        print(arg)

    # Example usage:
    print_arguments(1, 2, 3)
    print_arguments('Hello', 'world', '!')
```

```
Output:

Ankit Baingane
0827CI211028
1
2
3
Hello
world
!
```

Question 58

Write a Python function to reverse a string

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def reverse_string(input_string):
"""
Function to reverse a given string.

Parameters:
input_string (str): The string to be reversed.

Returns:
str: The reversed string.
"""
return input_string[::-1]

# Example usage:
original_string = "hello"
reversed_string = reverse_string(original_string)
print("Original string:", original_string)
print("Reversed string:", reversed_string)
```

```
Output:

Ankit Baingane
0827CI211028
Original string: hello
Reversed string: olleh
```

Question 59

Write a Python function to print the document string

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def print_docstring(func):
"""
Function to print the docstring of a given function.

Parameters:
func: The function whose docstring is to be printed.

Returns:
None
```

```
    """
    print(func.__doc__)

    # Example usage:
    def example_function():
        """
        This is an example function.
        It does nothing.
        """
        pass

    print_docstring(example_function)
```

Output:

```
Ankit Baingane
0827CI211028

   This is an example function.
   It does nothing.
```

Question 60

Write a Python function to demonstrate the scope of local and global variable.

Solution:

```
    print("Ankit Baingane")
    print("0827CI211028")
    global_variable = "I am global"

    def demonstrate_scope():
        """
        Function to demonstrate the scope of local and global variables.
        """
        local_variable = "I am local"
        print("Inside the function:")
        print("Local variable:", local_variable)
        print("Global variable:", global_variable)

    # Call the function
    demonstrate_scope()

    # Attempt to access local_variable
    # outside the function - this will raise an error
    # print("Outside the function:")
    # print("Local variable:", local_variable)

    # Access global_variable outside the function
    print("Accessing global variable outside the function:", global_variable)
```

Output:

```
Ankit Baingane
0827CI211028
Inside the function:
Local variable: I am local
Global variable: I am global
Accessing global variable outside the function: I am global
```

Question 61

Write a Python function to calculate the factorial of a number (a nonnegative integer). The function accepts the number as an argument

Solution:

```python
print("Ankit Baingane")
print("0827CI211028")
def factorial(n):
"""
Function to calculate the factorial of a nonnegative integer.

Parameters:
n (int): The number whose factorial is to be calculated.

Returns:
int: The factorial of the input number.
"""
if n < 0:
    return "Factorial is not defined for negative numbers"
elif n == 0 or n == 1:
    return 1
else:
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Example usage:
number = 5
print(f"The factorial of {number} is:", factorial(number))
print("\n")
```

```
Output:

Ankit Baingane
0827CI211028
The factorial of 5 is: 120
```

Question 62

Write a Python function to check whether a number falls within a given range.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def check_range(number, start, end):
"""
Function to check whether a number falls within a given range.

Parameters:
number: The number to check.
start: The start of the range (inclusive).
end: The end of the range (inclusive).

Returns:
bool: True if the number falls within the range, False otherwise.
"""
return start <= number <= end

# Example usage:
num = 7
start_range = 5
end_range = 10

if check_range(num, start_range, end_range):
    print(f"{num} falls within the range [{start_range}, {end_range}]")
else:
    print(f"{num} does not fall within the range [{start_range},
     {end_range}]")
```

```
Output:

Ankit Baingane
0827CI211028
7 falls within the range [5, 10]
```

Question 63

Write a Python function that accepts a string and counts the number of upper and lower case letters.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def count_upper_lower(string):
"""
Function to count the number of upper and lower case letters in a string.

Parameters:
string (str): The input string.

Returns:
tuple: A tuple containing the count of upper
case letters and lower case letters, respectively.
"""
upper_count = 0
```

```
    lower_count = 0
    for char in string:
        if char.isupper():
            upper_count += 1
        elif char.islower():
            lower_count += 1
    return upper_count, lower_count

    # Example usage:
    input_string = "Hello World"
    upper, lower = count_upper_lower(input_string)
    print("Number of upper case letters:", upper)
    print("Number of lower case letters:", lower)
```

Output:

```
Ankit Baingane
0827CI211028
Number of upper case letters: 2
Number of lower case letters: 8
```

Question 64

Write a Python function that takes a list and returns a new list with distinct elements from the first list. sample list:[1,2,3,3,3,3,4,5],Unique list:[1,2,3,4,5]

Solution:

```
    print("Ankit Baingane")
    print("0827CI211028")
    def distinct_elements(input_list):
    """
    Function to return a new list with distinct elements from the input list.

    Parameters:
    input_list (list): The input list.

    Returns:
    list: A new list with distinct elements.
    """
    return list(set(input_list))

    # Example usage:
    original_list = [1, 2, 2, 3, 3, 4, 5, 5]
    distinct_list = distinct_elements(original_list)
    print("Original List:", original_list)
    print("Distinct List:", distinct_list)
```

Output:

```
Ankit Baingane
0827CI211028
```

```
Original List: [1, 2, 2, 3, 3, 4, 5, 5]
Distinct List: [1, 2, 3, 4, 5]
```

Question 65

Write a Python function that takes a number as a parameter and checks whether the numberis prime or not

Solution:

```python
print("Ankit Baingane")
print("0827CI211028")
def is_prime(number):
"""
Function to check whether a number is prime or not.

Parameters:
number (int): The number to check.

Returns:
bool: True if the number is prime, False otherwise.
"""
if number <= 1:
    return False
elif number <= 3:
    return True
elif number % 2 == 0 or number % 3 == 0:
    return False
else:
    i = 5
    while i * i <= number:
        if number % i == 0 or number % (i + 2) == 0:
            return False
        i += 6
    return True

# Example usage:
num = 19
if is_prime(num):
    print(f"{num} is a prime number.")
else:
    print(f"{num} is not a prime number.")
print("\n")
```

Output:

```
Ankit Baingane
0827CI211028
19 is a prime number.
```

Question 66

Write a Python program to access a function inside a function

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def outer_function():
"""
Outer function.
"""
    print("This is the outer function.")

    def inner_function():
        """
        Inner function.
        """
        print("This is the inner function.")

# Call the inner function
inner_function()

# Call the outer function
outer_function()
```

```
Output:

Ankit Baingane
0827CI211028
This is the outer function.
This is the inner function.
```

Question 67

Write a Python function that checks whether a passed string is a palindrome or not.

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def is_palindrome(string):
"""
Function to check whether a passed string is a palindrome or not.

Parameters:
string (str): The string to check.

Returns:
bool: True if the string is a palindrome, False otherwise.
"""
# Convert the string to lowercase and remove spaces
clean_string = string.lower().replace(" ", "")
```

```
    # Compare the string with its reverse
    return clean_string == clean_string[::-1]

    # Example usage:
    input_string = "A man a plan a canal Panama"
    if is_palindrome(input_string):
        print(f"'{input_string}' is a palindrome.")
    else:
        print(f"'{input_string}' is not a palindrome.")
```

Output:

```
Ankit Baingane
0827CI211028
'A man a plan a canal Panama' is a palindrome.
```

Question 68

Write a python function to create and print a list where the values are the squares of numbers between 1 and 30

Solution:

```
    print("Ankit Baingane")
    print("0827CI211028")
    def squares_list():
    """
    Function to create and print a list where the values
    are the squares of numbers between 1 and 30.

    Returns:
    list: A list containing the squares of numbers between 1 and 30.
    """
    squares = [x ** 2 for x in range(1, 31)]
    return squares

    # Example usage:
    squares = squares_list()
    print("List of squares of numbers between 1 and 30:", squares)
```

Output:

```
Ankit Baingane
0827CI211028
List of squares of numbers between 1 and 30: [1, 4, 9, 16, 25, 36, 49, 64, 81,
100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441,
484, 529, 576, 625, 676, 729, 784, 841, 900]
```

Question 69

Write a Python program to print the even numbers from a given list

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
def print_even_numbers(input_list):
"""
Function to print the even numbers from a given list.

Parameters:
input_list (list): The input list.

Returns:
None
"""
even_numbers = [num for num in input_list if num % 2 == 0]
print("Even numbers from the given list:", even_numbers)

# Example usage:
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
print_even_numbers(numbers)
```

```
Output:

Ankit Baingane
0827CI211028
Even numbers from the given list: [2, 4, 6, 8, 10]
```

Question 70

Write a program to create a class student with data member name. roll no. Semester. and display the data using object

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
class Student:
"""
Class representing a student.
"""
def __init__(self, name, roll_no, semester):
    """
    Constructor to initialize the data members of the Student class.

    Parameters:
    name (str): The name of the student.
    roll_no (str): The roll number of the student.
    semester (str): The semester of the student.
    """
    self.name = name
```

```
        self.roll_no = roll_no
        self.semester = semester

    def display_data(self):
        """
        Method to display the data of the student.
        """
        print("Name:", self.name)
        print("Roll No:", self.roll_no)
        print("Semester:", self.semester)

    # Create an object of the Student class
    student1 = Student("Ramesh Kumar", "12345", "Spring 2024")

    # Display the data using the object
    print("Student Data:")
    student1.display_data()
```

Output:

```
Ankit Baingane
0827CI211028
Student Data:
Name: Ramesh Kumar
Roll No: 12345
Semester: Spring 2024
```

Question 71: Write a program to demonstrate the use of init

Solution:

```
    print("Ankit Baingane")
    print("0827CI211028")
    class Car:
    """
    Class representing a car.
    """
    def __init__(self, make, model, year):
        """
        Constructor to initialize the data members of the Car class.

        Parameters:
        make (str): The make of the car.
        model (str): The model of the car.
        year (int): The manufacturing year of the car.
        """
        self.make = make
        self.model = model
        self.year = year
        self.odometer_reading = 0  # Additional attribute
```

```
    def get_car_info(self):
        """
        Method to display information about the car.
        """
        car_info = f"{self.year} {self.make} {self.model}"
        return car_info

    def read_odometer(self):
        """
        Method to read the odometer reading of the car.
        """
        print(f"This car has {self.odometer_reading} miles on it.")

    # Create an object of the Car class
    my_car = Car("Toyota", "Corolla", 2022)

    # Display information about the car
    print("Car Information:", my_car.get_car_info())

    # Read the odometer reading of the car
    my_car.read_odometer()
```

```
Output:

Ankit Baingane
0827CI211028
Car Information: 2022 Toyota Corolla
This car has 0 miles on it.
```

Question 72

Write a program for Inheritence.a) single level b) Multiple c) Multilevel d) Hybrid e) Hierarchical

Solution:

```
    print("Ankit Baingane")
    print("0827CI211028")

    # Single level inheritance
    class Parent:
        def parent_method(self):
            print("This is the parent method.")

    class Child(Parent):
        def child_method(self):
            print("This is the child method.")

    # Single level inheritance
    child_obj = Child()
    child_obj.parent_method()
    child_obj.child_method()

    # Multiple inheritance
```

```
class Parent1:
    def method1(self):
        print("This is method 1 of Parent1.")

class Parent2:
    def method2(self):
        print("This is method 2 of Parent2.")

class Child1(Parent1, Parent2):
    def child_method1(self):
        print("This is the child1 method.")

# Multiple inheritance
child_obj1 = Child1()
child_obj1.method1()
child_obj1.method2()
child_obj1.child_method1()

# Multilevel inheritance
class Grandparent:
    def grandparent_method(self):
        print("This is the grandparent method.")

class Parent3(Grandparent):
    def parent_method2(self):
        print("This is the parent method.")

class Child2(Parent3):
    def child_method2(self):
        print("This is the child method.")

# Multilevel inheritance
child_obj2 = Child2()
child_obj2.grandparent_method()
child_obj2.parent_method2()
child_obj2.child_method2()

# Hybrid inheritance
class Parent4:
    def method1(self):
        print("This is method 1 of Parent4.")

class Parent5:
    def method2(self):
        print("This is method 2 of Parent5.")

class Parent6:
    def method3(self):
        print("This is method 3 of Parent6.")

class Child3(Parent4, Parent5, Parent6):
    def child_method3(self):
        print("This is the child method.")

# Hybrid inheritance
child_obj3 = Child3()
```

```
child_obj3.method1()
child_obj3.method2()
child_obj3.method3()
child_obj3.child_method3()

# Hierarchical inheritance
class Parent7:
    def parent_method4(self):
        print("This is the parent method.")

class Child4(Parent7):
    def child1_method4(self):
        print("This is the child1 method.")

class Child5(Parent7):
    def child2_method4(self):
        print("This is the child2 method.")

# Hierarchical inheritance
child1_obj = Child4()
child1_obj.parent_method4()
child1_obj.child1_method4()

child2_obj = Child5()
child2_obj.parent_method4()
child2_obj.child2_method4()
```

Output:

```
Ankit Baingane
0827CI211028
This is the parent method.
This is the child method.
This is method 1 of Parent1.
This is method 2 of Parent2.
This is the child1 method.
This is the grandparent method.
This is the parent method.
This is the child method.
This is method 1 of Parent4.
This is method 2 of Parent5.
This is method 3 of Parent6.
This is the child method.
This is the parent method.
This is the child1 method.
This is the parent method.
This is the child2 method.
```

Question 73

Write a program to demonstrate overriding

Solution:

```
print("Ankit Baingane")
print("0827CI211028")
class Parent:
def show_message(self):
    print("This is the parent message.")

class Child(Parent):
    def show_message(self):
        print("This is the overridden message.")

# Create objects of both Parent and Child classes
parent_obj = Parent()
child_obj = Child()

# Call the method on each object
print("Calling show_message() method of Parent class:")
parent_obj.show_message()

print("\nCalling show_message() method of Child class:")
child_obj.show_message()
```

Output:

```
Ankit Baingane
0827CI211028
Calling show_message() method of Parent class:
This is the parent message.

Calling show_message() method of Child class:
This is the overridden message.
```