Cycle -1

1. Program to Print all non-Prime Numbers in an Interval

CODE:

```
start = int(input("Enter the start of the interval : "))
end = int(input("Enter the end of the interval : "))

print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")
print("Non prime numbers in the given interval are : ")
for num in range(start,end + 1):
   if num > 1:
       is_prime = True
       for i in range(2, int(num**0.5) + 1):
           if num % i == 0:
               is_prime = False
               break
       if not is_prime:
           print(num)
```

OUTPUT:

```
"/home/sjcet/anjala007/python/DS & ML/bin/python" /home/sjcet/anjala007/sem 3/DS & ML/Cycle - 1/nonprime.py
Enter the start of the interval : 10
Enter the end of the interval : 20
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Non prime numbers in the given interval are :
10
12
14
15
16
18
20

Process finished with exit code 0
```

2. Program to print the first N Fibonacci numbers.

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")
n = int(input("Enter the number of elements you want to fibonacci series : "))
a=0
b=1
for i in range(0,n):
   print(a,end=" ")
   c=a+b
   a=b
   b=c
```

OUTPUT:

```
"/home/sjcet/anjala007/python/DS & ML/bin/python" /home/sjcet/anjala007/sem 3/DS & ML/Cycle - 1/fib.py
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter the number of elements you want to fibonacci series : 10
0 1 1 2 3 5 8 13 21 34
Process finished with exit code 0
```

3. Given sides of a triangle, write a program to check whether given triangle is an isosceles, equilateral or scalene.

CODE:

```
def type_of_triangle(a,b,c):
   if a==b and b==c:
       print("Triangle is equilateral")
   elif a==b or b==c or a==c:
       print("Triangle is isosceles")
   else:
       print("Triangle is scalane")


print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")
a = int(input("Enter length of side A : "))
b = int(input("Enter length of side B : "))
c = int(input("Enter length of side C : "))

type_of_triangle(a,b,c)
```

OUTPUT:

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter length of side A : 5
Enter length of side B : 5
Enter length of side C : 5
Triangle is equilateral
```

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter length of side A : 5
Enter length of side B : 4
Enter length of side C : 3
Triangle is scalane
```

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter length of side A : 6
Enter length of side B : 3
Enter length of side C : 6
Triangle is isosceles
```

4. Program to check whether given pair of number is coprime

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")

n1 = int(input("Enter the first number : "))
n2 = int(input("Enter the second number : "))
mn = min(n1,n2)
for i in range(1,mn+1):
   if n1%i==0 and n2%i==0:
       hcf = i
if hcf==1:
   print("The number are Co-Prime")
else:
   print("The numbers are not Co-Prime")
```

OUTPUT:

```
"/home/sjcet/anjala007/python/DS & ML/bin/python" /home/sjcet/anjala007/sem 3/DS & ML/Cycle - 1/coprime.py
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter the first number : 15
Enter the second number : 28
The number are Co-Prime

Process finished with exit code 0
```

5. Program to find the roots of a quadratic equation(rounded to 2 decimal places)

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")

import cmath
a = float(input("Enter the value for a : "))
b = float(input("Enter the value for b : "))
c = float(input("Enter the value for c : "))

d = (b**2)-(4*a*c)

sol1 = (-b-cmath.sqrt(d))/(2*a)
sol2 = (-b+cmath.sqrt(d))/(2*a)
print('The solutions are {0} and {1}'.format(sol1,sol2))
```

OUTPUT:

```
"/home/sjcet/anjala007/python/DS & ML/bin/python" /home/sjcet/anjala007/sem 3/DS & ML/Cycle - 1/quadratic.py
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter the value for a : 8
Enter the value for b : 5
Enter the value for c : 9
The solutions are (-0.3125-1.0135796712641785j) and (-0.3125+1.0135796712641785j)

Process finished with exit code 0
```

6. Program to check whether a given number is perfect number or not(sum of factors =number)

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")

num = int(input("Enther the number : "))
sum = 0
for i in range(1,num):
   if(num%i==0):
       sum = sum + i;

if(sum==num):
   print("The number entered is a perfect number")
else:
   print("The number entered is not a perfect number")
```

OUTPUT:

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enther the number : 25
The number entered is not a perfect number

SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enther the number : 6
The number entered is a perfect number
```

7. Program to display amstrong numbers upto 1000

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")

limit = 1000

for num in range(100, limit + 1):
   order = len(str(num))
   sum = 0
   temp = num
   while temp > 0:
       digit = temp % 10
       sum += digit ** order
       temp //= 10

   if num == sum:
       print(num)
```

OUTPUT:

```
"/home/sjcet/anjala007/python/DS & ML/bin/python" /home/sjcet/anjala007/sem 3/DS & ML/Cycle - 1/amstrong.py
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
153
370
371
407
```

8. Store and display the days of a week as a List, Tuple, Dictionary, Set. Also demonstrate different ways to store values in each of them. Display its type also.

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")

day_list = [
"Sunday","Monday","Tuesday","wednesday","Thursday","Friday","Saturday"]
print("Lists values : ",day_list)
print("Type of List :",type(day_list))

day_tuple =
("Sunday","Monday","Tuesday","wednesday","Thursday","Friday","Saturday")
print("Tuple values : ",day_tuple)
print("Type of tuple : ", type(day_tuple))

day_dict =
{1:"Sunday",2:"Monday",3:"Tuesday",4:"wednesday",5:"Thursday",6:"Friday",7:"Sa
turday" }
print("Dictionary : ",day_dict)
print("Type of Dictionary : ",type(day_dict))

day_set =
{"Sunday","Monday","Tuesday","wednesday","Thursday","Friday","Saturday"}
print("Dictionary : ",day_set)
print("Type of Set : ",type(day_set))
```

OUTPUT:

```
"/home/sjcet/anjala007/python/DS & ML/bin/python" /home/sjcet/.config/JetBrains/PyCharmCE2023.2/scratches/8.py
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Lists values :  ['Sunday', 'Monday', 'Tuesday', 'wednesday', 'Thursday', 'Friday', 'Saturday']
Type of List : <class 'list'>
Tuple values :  ('Sunday', 'Monday', 'Tuesday', 'wednesday', 'Thursday', 'Friday', 'Saturday')
Type of tuple :  <class 'tuple'>
Dictionary :  {1: 'Sunday', 2: 'Monday', 3: 'Tuesday', 4: 'wednesday', 5: 'Thursday', 6: 'Friday', 7: 'Saturday'}
Type of Dictionary :  <class 'dict'>
Dictionary :  {'Sunday', 'wednesday', 'Tuesday', 'Friday', 'Monday', 'Saturday', 'Thursday'}
Type of Set :  <class 'set'>
```

9. Write a program to add elements of given 2 lists

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")

list1 = [1,2,3,4,5,6]
list2 = [11,22,33,44,55,66]

if len(list1) == len(list2):
   result = [list1[i] + list2[i] for i in range(len(list1))]
   print("Result : ",result)
else:
   print("The lists have different lengths so addition is not possible")
```

OUTPUT:

```
"/home/sjcet/anjala007/python/DS & ML/bin/python" /home/sjcet/anjala007/sem 3/DS & ML/Cycle - 1/listsum.py
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Result :  [12, 24, 36, 48, 60, 72]

Process finished with exit code 0
```

10. Write a program to find the sum of 2 matrices using nested List.

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")

X = [[1,2,3],[4,5,6],[7,8,9]]
Y = [[1,2,3,],[4,5,6],[7,8,9]]

result = [[X[i][j] + Y[i][j] for j in range
(len(X[0]))] for i in range(len(X))]

for r in result:
   print(r)
```

OUTPUT:

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
[2, 4, 6]
[8, 10, 12]
[14, 16, 18]
```

11. Write a program to perform bubble sort on a given set of elements.

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")
def bubble_sort(arr):
    n = len(arr)
    # Traverse through all elements in the list
    for i in range(n):
        # Flag to optimize the sorting process
        swapped = False
        # Last i elements are already in place, so we don't need to compare
them again
        for j in range(0, n - i - 1):
            # If the element found is greater than the next element, swap them
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
                swapped = True
        # If no two elements were swapped in the inner loop, the list is
already sorted
        if not swapped:
            break
# Example usage
elements = [64, 34, 25, 12, 22, 11, 90]
print("Original List:", elements)
bubble_sort(elements)
print("Sorted List:", elements)
```

OUTPUT:

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Original List: [64, 34, 25, 12, 22, 11, 90]
Sorted List: [11, 12, 22, 25, 34, 64, 90]

Process finished with exit code 0
```

12. Program to find the count of each vowel in a string(use dictionary)

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")
def count_vowels(string):
    # Initialize a dictionary to store the counts of each vowel
    vowel_count = {'A': 0, 'E': 0, 'I': 0, 'O': 0, 'U': 0}
    # Convert the input string to uppercase to handle both lowercase and
uppercase vowels
    string = string.upper()
    # Iterate through the characters in the string
    for char in string:
        if char in vowel_count:
            vowel_count[char] += 1
    return vowel_count
# Input string
input_string = input("Enter a string: ")
# Call the function to count vowels
vowel_counts = count_vowels(input_string)
# Display the counts
for vowel, count in vowel_counts.items():
    print(f"{vowel}: {count}")
```

OUTPUT:

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter a string: anjalaMICHAEL
A: 4
E: 1
I: 1
O: 0
U: 0
```

13. Write a Python program that accepts a positive number and subtract from this number the sum of its digits and so on. Continue this operation until the number is positive (eg: 256-> 2+5+6=13 256-13=243 243-9=232……..)

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")
def sum_of_digits(n):
    # Function to calculate the sum of digits of a number
    digit_sum = 0
    while n > 0:
        digit_sum += n % 10
        n //= 10
    return digit_sum
def main():
    try:
        # Input a positive number
        num = int(input("Enter a positive number: "))
        if num <= 0:
            print("Please enter a positive number.")
            return
        while num > 0:
            # Calculate the sum of digits of the current number
            current_sum = sum_of_digits(num)
            # Print the operation and the result
            print(f"{num} - {current_sum} =", num - current_sum)
            # Update the number for the next iteration
            num -= current_sum
    except ValueError:
        print("Invalid input. Please enter a valid positive number.")
if __name__ == "__main__":
    main()
```

OUTPUT:

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter a positive number: 100
100 - 1 = 99
99 - 18 = 81
81 - 9 = 72
72 - 9 = 63
63 - 9 = 54
54 - 9 = 45
45 - 9 = 36
36 - 9 = 27
27 - 9 = 18
18 - 9 = 9
9 - 9 = 0
```

14. Write a Python program that accepts a 10 digit mobile number, and find the digits which are absent in a given mobile number

CODE:

```
print("SJC22MCA-2007 : ANJALA MICHAEL")
print("Batch : MCA 2022-24")
def find_absent_digits(mobile_number):
    all_digits = set("0123456789")
    mobile_digits = set(mobile_number)
    absent_digits = all_digits - mobile_digits
    return sorted(list(absent_digits))
try:
    mobile_number = input("Enter a 10-digit mobile number: ")

    if len(mobile_number) == 10 and mobile_number.isdigit():
        absent_digits = find_absent_digits(mobile_number)

        if absent_digits:
            print("Absent digits in the mobile number:", ',
'.join(absent_digits))
        else:
            print("The mobile number contains all digits from 0 to 9.")
    else:
        print("Invalid input. Please enter a valid 10-digit mobile number.")
except ValueError:
    print("Invalid input. Please enter a valid 10-digit mobile number.")
```

OUTPUT:

```
SJC22MCA-2007 : ANJALA MICHAEL
Batch : MCA 2022-24
Enter a 10-digit mobile number: 9496788616
Absent digits in the mobile number: 0, 2, 3, 5
```