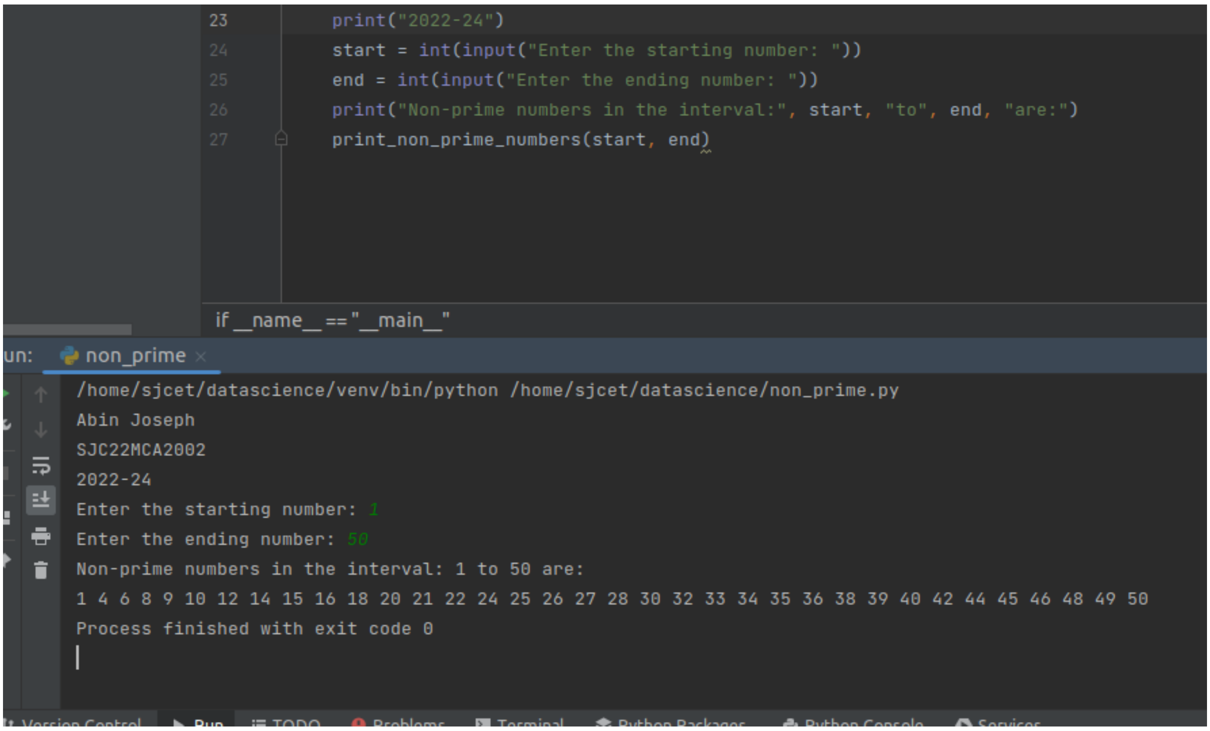
CYCLE 1

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

Code

```
def is_prime(number):
    if number <= 1:
        return False
    elif number <= 3:
        return True
    elif number % 2 == 0 or number % 3 ==0:
        return False
    i = 5
    while i * i <= number:
        if number % i == 0 or number % (i + 2) == 0:
            return False
        i += 6
    return True

def print_non_prime_numbers(start, end):
    for num in range(start, end + 1):
        if not is_prime(num):
            print(num, end=" ")

if __name__ == "__main__":
    print("Abin Joseph")
    print("SJC22MCA2002")
    print("2022-24")
    start = int(input("Enter the starting number: "))
    end = int(input("Enter the ending number: "))
    print("Non-prime numbers in the interval:", start, "to", end, "are:")
    print_non_prime_numbers(start, end)
```



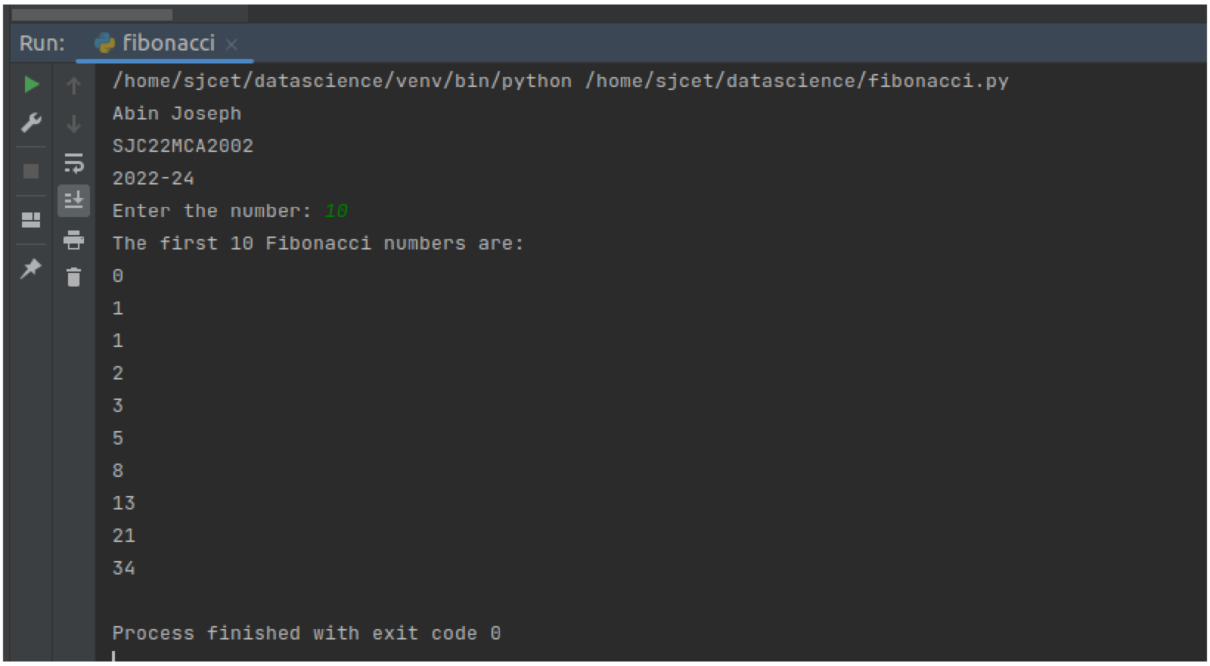
```
    print("2022-24")
    start = int(input("Enter the starting number: "))
    end = int(input("Enter the ending number: "))
    print("Non-prime numbers in the interval:", start, "to", end, "are:")
    print_non_prime_numbers(start, end)
```

```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/non_prime.py
Abin Joseph
SJC22MCA2002
2022-24
Enter the starting number: 1
Enter the ending number: 50
Non-prime numbers in the interval: 1 to 50 are:
1 4 6 8 9 10 12 14 15 16 18 20 21 22 24 25 26 27 28 30 32 33 34 35 36 38 39 40 42 44 45 46 48 49 50
Process finished with exit code 0
```

2. Program to print the first N Fibonacci numbers.

Code

```
def fibonacci(n):
    fibonacci_sequence = []
    a, b = 0, 1
    for _ in range(n):
        fibonacci_sequence.append(a)
        a, b = b, a + b
    return fibonacci_sequence

print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")
N = int(input("Enter the number: "))
if N <= 0:
    print("Please enter a positive integer")
else:
    fibonacci_number = fibonacci(N)
    print("The first", N, "Fibonacci numbers are:")
    for num in fibonacci_number:
        print(num)
```

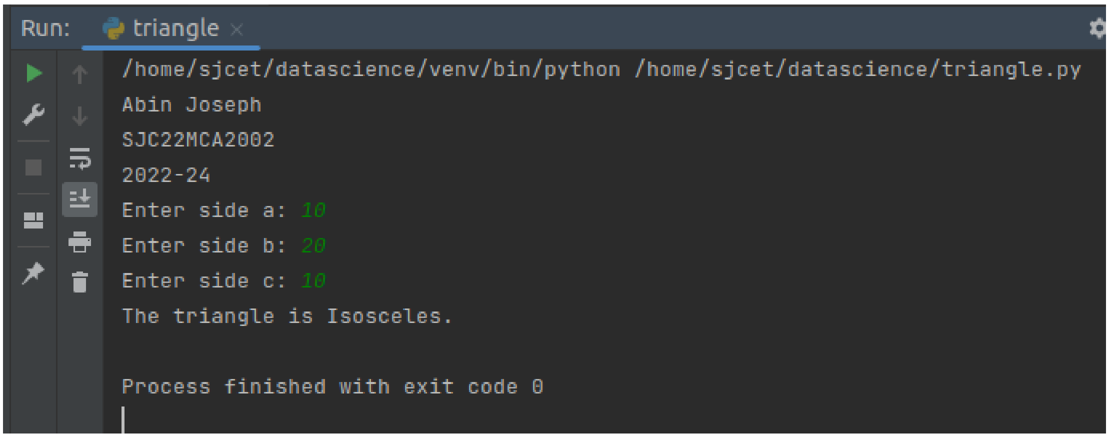


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/fibonacci.py
Abin Joseph
SJC22MCA2002
2022-24
Enter the number: 10
The first 10 Fibonacci numbers are:
0
1
1
2
3
5
8
13
21
34

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 triangle_type(a, b, c):
   if a == b == c:
      return "Equilateral"
   elif a == b or a == c or b == c:
      return "Isosceles"
   else:
      return "Scalene"

print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")
a = float(input("Enter side a: "))
b = float(input("Enter side b: "))
c = float(input("Enter side c: "))
triangle = triangle_type(a, b, c)
print(f"The triangle is {triangle}.")
```

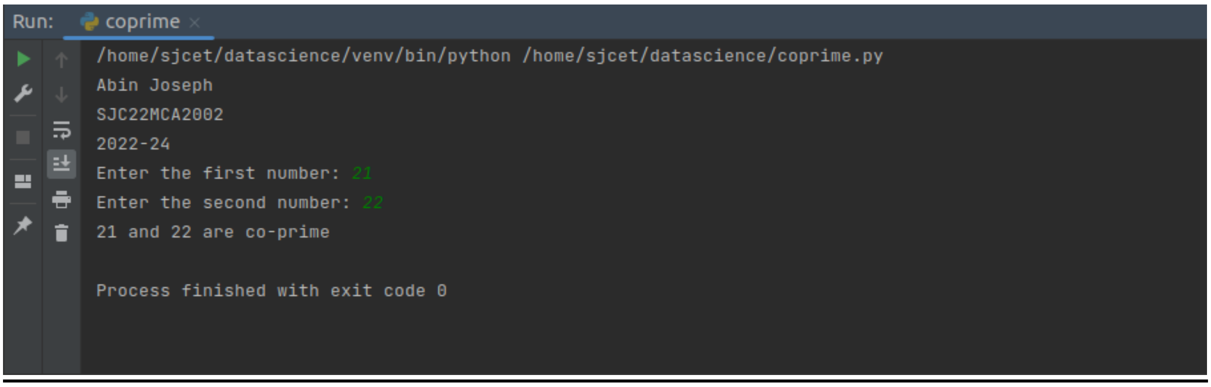


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/triangle.py
Abin Joseph
SJC22MCA2002
2022-24
Enter side a: 10
Enter side b: 20
Enter side c: 10
The triangle is Isosceles.

Process finished with exit code 0
```

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

Code

```
import math
def coprime(a, b):
    gcd = math.gcd(a, b)
    if gcd == 1:
        return True
    else:
        return False
print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
if coprime(a,b):
    print(f"{a} and {b} are co-prime")
else:
    print(f"{a} and {b} are not co-prime")
```

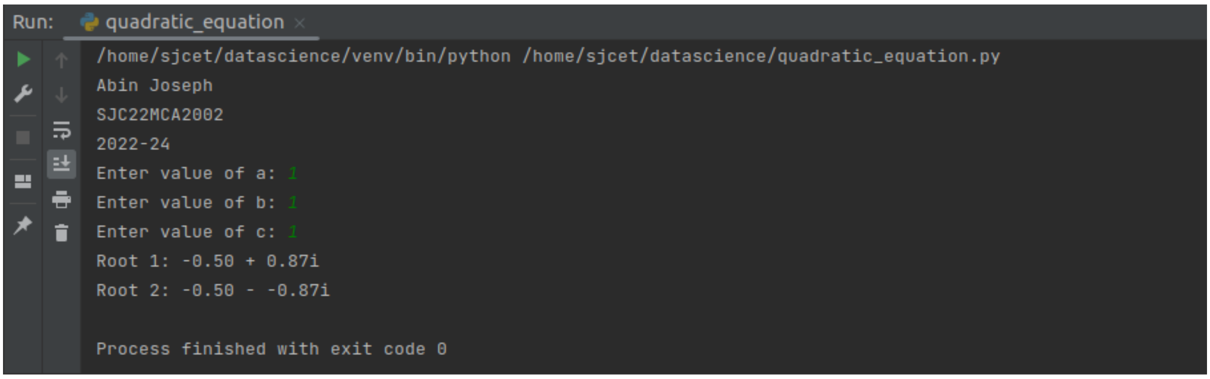


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/coprime.py
Abin Joseph
SJC22MCA2002
2022-24
Enter the first number: 21
Enter the second number: 22
21 and 22 are co-prime

Process finished with exit code 0
```

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

Code

```
import math
print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")

a = float(input("Enter value of a: "))
b = float(input("Enter value of b: "))
c = float(input("Enter value of c: "))
discri = b**2 - 4*a*c

if discri > 0:
   root1 = (-b + math.sqrt(discri)) / (2*a)
   root2 = (-b - math.sqrt(discri)) / (2*a)
   print(f"Root 1: {round(root1, 2)}")
   print(f"Root 2: {round(root2, 2)}")
elif discri == 0:
   root = -b / (2*a)
   print(f"Root: {round(root, 2)}")
else:
   real_part = -b / (2*a)
   img_part = math.sqrt(-discri) / (2*a)
   root1 = complex(real_part, img_part)
   root2 = complex(real_part, -img_part)
   print(f"Root 1: {root1.real:.2f} + {root1.imag:.2f}i")
   print(f"Root 2: {root2.real:.2f} - {root2.imag:.2f}i")
```

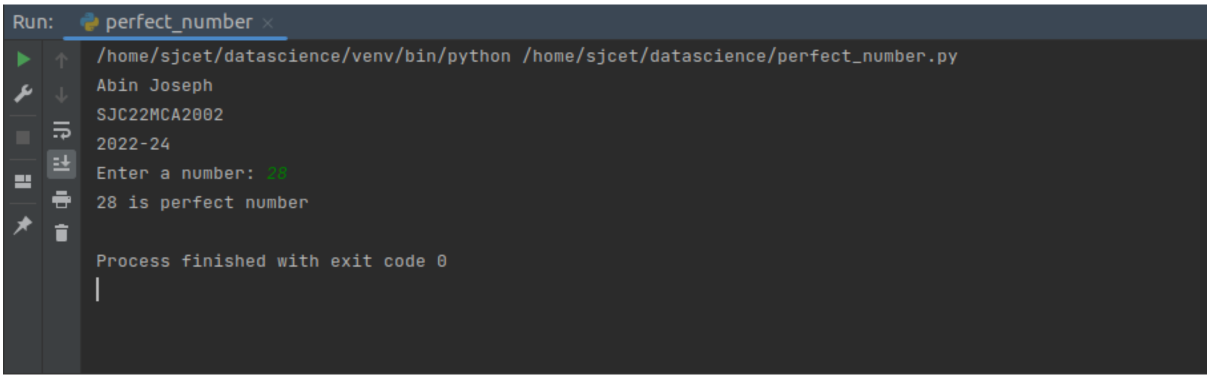


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/quadratic_equation.py
Abin Joseph
SJC22MCA2002
2022-24
Enter value of a: 1
Enter value of b: 1
Enter value of c: 1
Root 1: -0.50 + 0.87i
Root 2: -0.50 - -0.87i

Process finished with exit code 0
```

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

Code

```
def perfect_number(num):
    if num <= 0:
        return False
    sum_of_factor = 0
    for i in range(1, num):
        if num % i == 0:
            sum_of_factor += i

    return sum_of_factor == num
num = int(input("Enter a number: "))
if perfect_number(num):
    print(f"{num} is perfect number")
else:
    print(f"{num} is not perfect number")
```

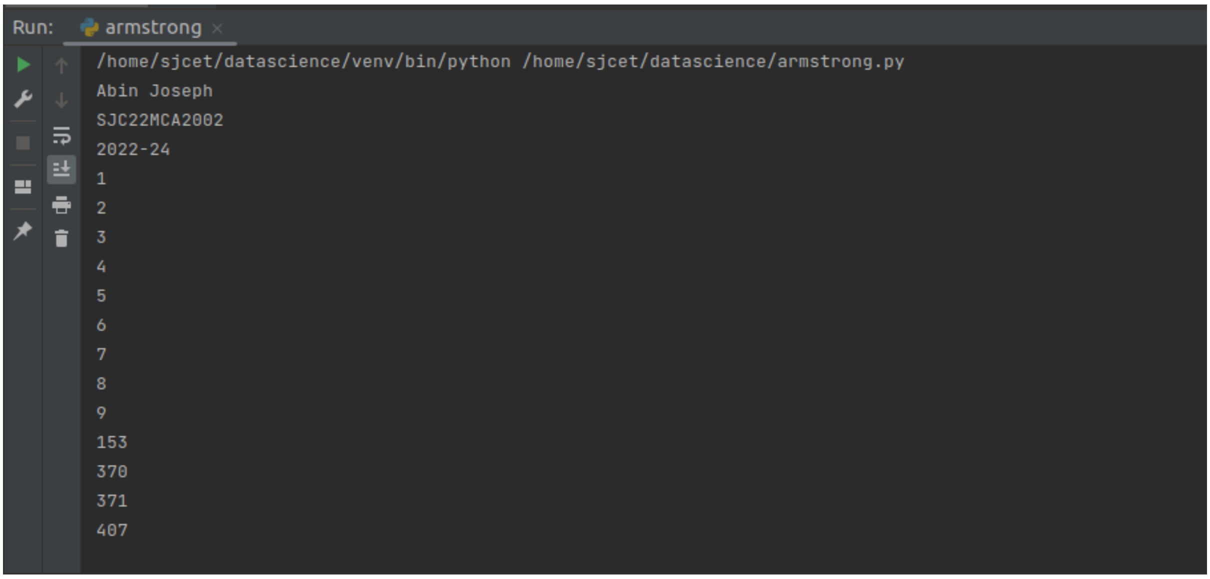


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/perfect_number.py
Abin Joseph
SJC22MCA2002
2022-24
Enter a number: 28
28 is perfect number

Process finished with exit code 0
```

7. Program to display amstrong numbers upto 1000

Code

```
def armstrong(a):
    num_digits = len(str(a))
    sum_digits = 0
    temp = a

    while temp > 0:
        digit = temp % 10
        sum_digits += digit ** num_digits
        temp //= 10

    if a == sum_digits:
        return True
    else:
        return False

print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")
for num in range(1, 1001):
    if armstrong(num):
        print(num)
```

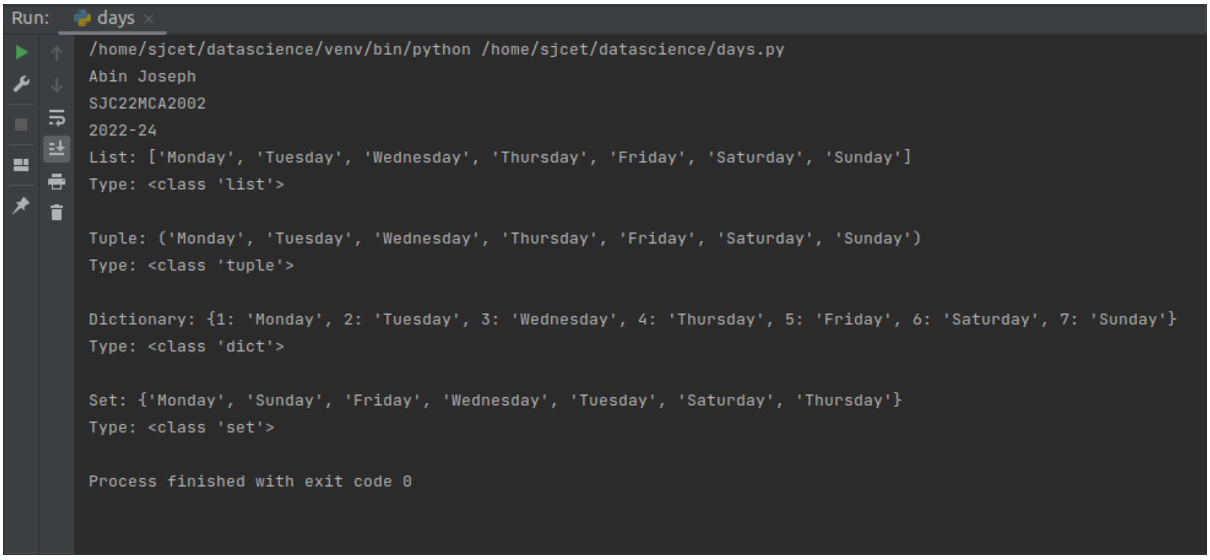


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/armstrong.py
Abin Joseph
SJC22MCA2002
2022-24
1
2
3
4
5
6
7
8
9
153
370
371
407
```

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("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")

days_list = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
print("List:", days_list)
print("Type:", type(days_list))

days_tuple = ('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday')
print("\nTuple:", days_tuple)
print("Type:", type(days_tuple))

days_dict = {
  1: 'Monday',
  2: 'Tuesday',
  3: 'Wednesday',
  4: 'Thursday',
  5: 'Friday',
  6: 'Saturday',
  7: 'Sunday'
}
print("\nDictionary:", days_dict)
print("Type:", type(days_dict))

days_set = {'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'}
print("\nSet:", days_set)
print("Type:", type(days_set))
```

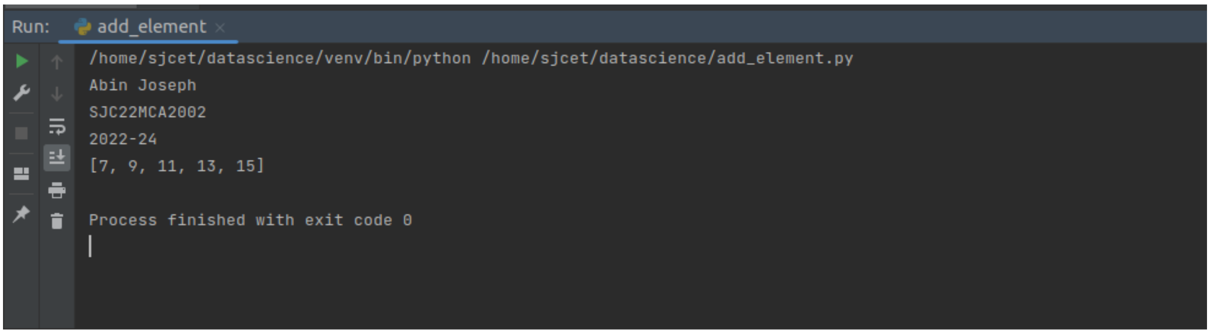


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/days.py
Abin Joseph
SJC22MCA2002
2022-24
List: ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
Type: <class 'list'>

Tuple: ('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday')
Type: <class 'tuple'>

Dictionary: {1: 'Monday', 2: 'Tuesday', 3: 'Wednesday', 4: 'Thursday', 5: 'Friday', 6: 'Saturday', 7: 'Sunday'}
Type: <class 'dict'>

Set: {'Monday', 'Sunday', 'Friday', 'Wednesday', 'Tuesday', 'Saturday', 'Thursday'}
Type: <class 'set'>

Process finished with exit code 0
```

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

Code

```
def add_lists(list1,list2):
   if len(list1) != len(list2):
      return "Lists must have same length"
   result = []
   for i in range(len(list1)):
      result.append(list1[i] + list2[i])
   return result

print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")

list1 = [1, 2, 3, 4, 5]
list2 = [6, 7, 8, 9, 10]
result_list = add_lists(list1, list2)
print(result_list)
```

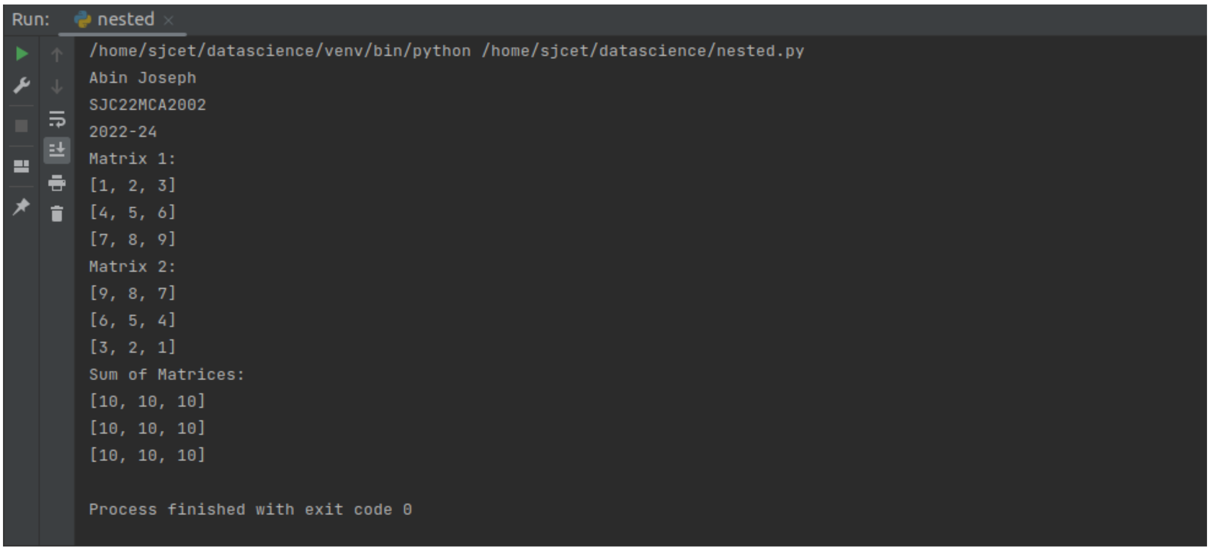


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/add_element.py
Abin Joseph
SJC22MCA2002
2022-24
[7, 9, 11, 13, 15]

Process finished with exit code 0
```

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

Code

```
def add_matrices(matrix1, matrix2):
    if len(matrix1) != len(matrix2) or len(matrix1[0]) != len(matrix2[0]):
        return "Matrices must have the same dimensions for addition."

    result = [[0 for _ in range(len(matrix1[0]))] for _ in range(len(matrix1))]

    for i in range(len(matrix1)):
        for j in range(len(matrix1[0])):
            result[i][j] = matrix1[i][j] + matrix2[i][j]

    return result

print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")

matrix1 = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]

matrix2 = [
    [9, 8, 7],
    [6, 5, 4],
    [3, 2, 1]
]

result_matrix = add_matrices(matrix1, matrix2)

if isinstance(result_matrix, str):
    print(result_matrix)
else:
    print("Matrix 1:")
    for row in matrix1:
        print(row)

    print("Matrix 2:")
    for row in matrix2:
        print(row)

    print("Sum of Matrices:")
    for row in result_matrix:
        print(row)
```

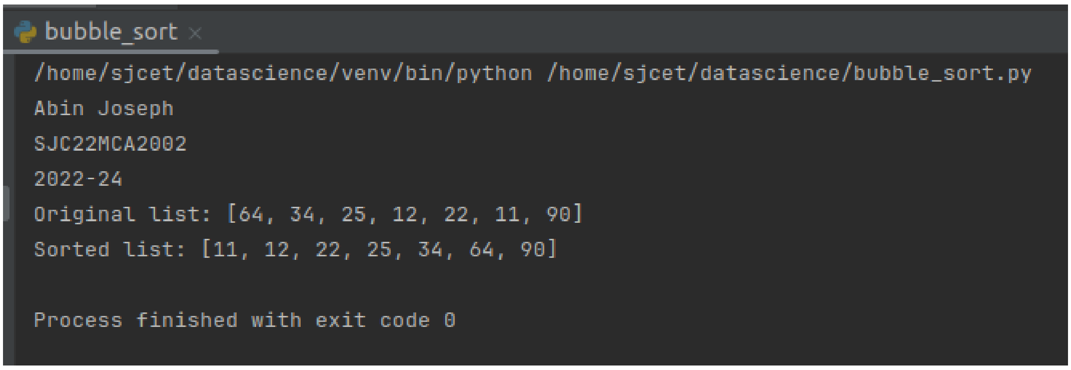


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/nested.py
Abin Joseph
SJC22MCA2002
2022-24
Matrix 1:
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
Matrix 2:
[9, 8, 7]
[6, 5, 4]
[3, 2, 1]
Sum of Matrices:
[10, 10, 10]
[10, 10, 10]
[10, 10, 10]

Process finished with exit code 0
```

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

Code

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]

if __name__ == "__main__":
    elements = [64, 34, 25, 12, 22, 11, 90]
    print("Original list:", elements)
    bubble_sort(elements)
    print("Sorted list:", elements)
```

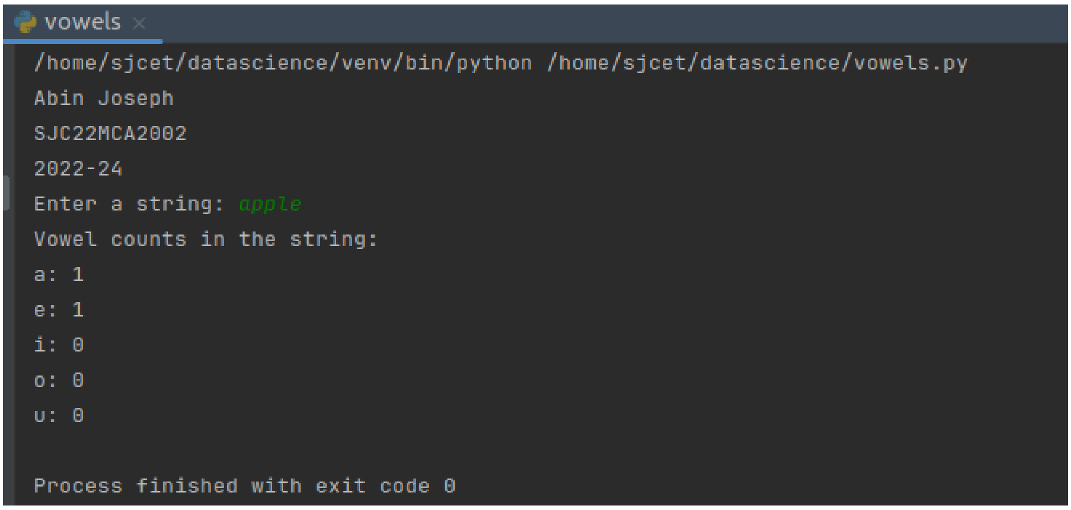


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/bubble_sort.py
Abin Joseph
SJC22MCA2002
2022-24
Original list: [64, 34, 25, 12, 22, 11, 90]
Sorted list: [11, 12, 22, 25, 34, 64, 90]

Process finished with exit code 0
```

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

Code

```
def count_vowels(string):
   vowel_counts = {'a': 0, 'e': 0, 'i': 0, 'o': 0, 'u': 0}
   string = string.lower()
   for char in string:
      if char in vowel_counts:
         vowel_counts[char] += 1
   return vowel_counts
print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")
input_string = input("Enter a string: ")
vowel_counts = count_vowels(input_string)
print("Vowel counts in the string:")
for vowel, count in vowel_counts.items():
   print(f"{vowel}: {count}")
```

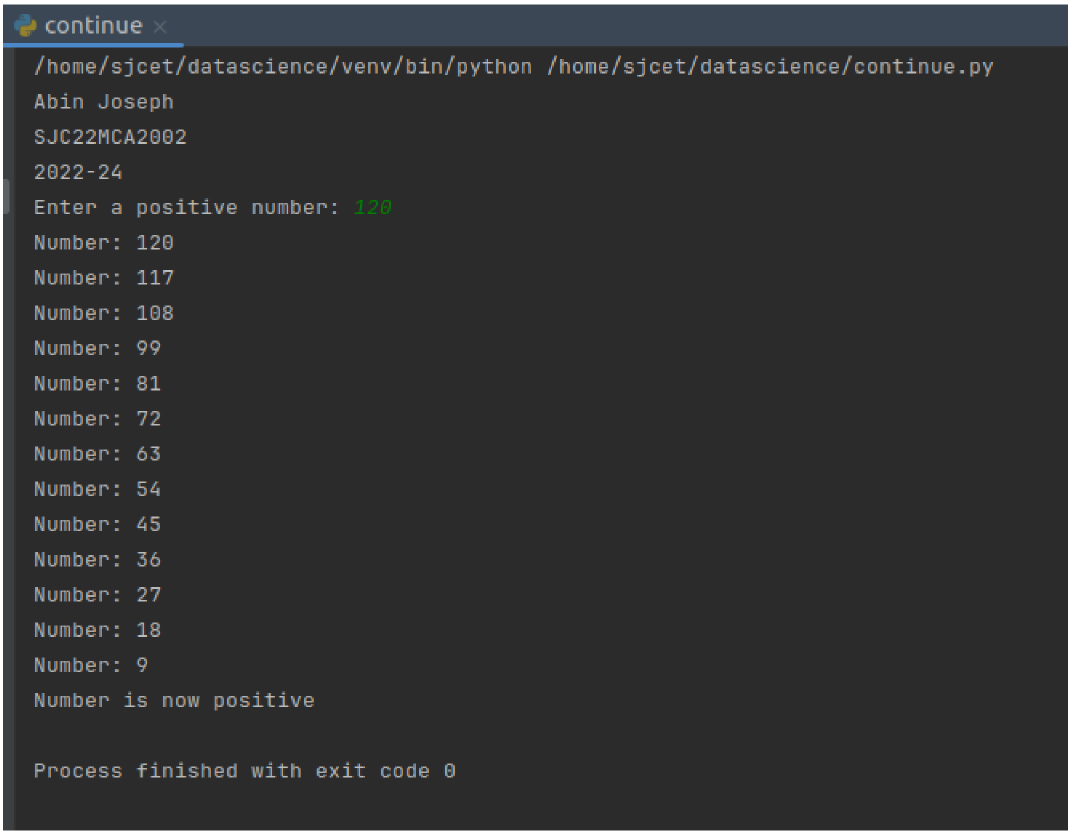


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/vowels.py
Abin Joseph
SJC22MCA2002
2022-24
Enter a string: apple
Vowel counts in the string:
a: 1
e: 1
i: 0
o: 0
u: 0

Process finished with exit code 0
```

13. Write a Python program that accept a positive number and subtract from this number the sum of its digits and so on. Continues this operation until the number is
Positive

Code

```
def sum_of_digits(number):
    digit_sum = 0
    while number > 0:
        digit_sum += number % 10
        number //= 10
    return digit_sum

print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")
def main():
    try:
        num = int(input("Enter a positive number: "))
        if num <= 0:
            print("Please enter a positive number")
            return
        while num > 0:
            print(f"Number: {num}")
            digit_sum = sum_of_digits(num)
            num -= digit_sum
        print("Number is now positive")
    except ValueError:
        print("Invalid input.")
if __name__ == "__main__":
    main()
```

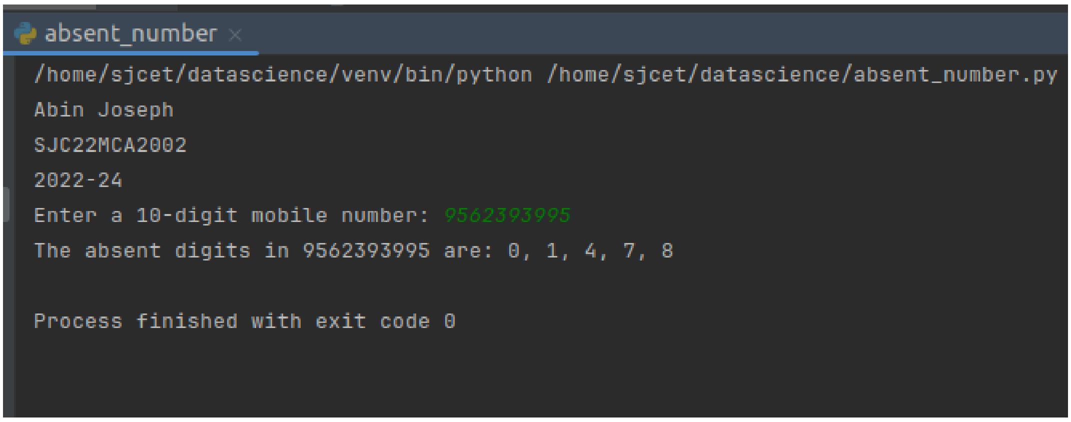


```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/continue.py
Abin Joseph
SJC22MCA2002
2022-24
Enter a positive number: 120
Number: 120
Number: 117
Number: 108
Number: 99
Number: 81
Number: 72
Number: 63
Number: 54
Number: 45
Number: 36
Number: 27
Number: 18
Number: 9
Number is now positive

Process finished with exit code 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

```
def find_absent_digits(mobile_number):
    all_digits = set(range(10))
    mobile_digits = set(int(digit) for digit in str(mobile_number) if digit.isdigit())
    absent_digits = all_digits - mobile_digits
    return absent_digits
print("Abin Joseph")
print("SJC22MCA2002")
print("2022-24")
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(f"The absent digits in {mobile_number} are: {', '.join(map(str, absent_digits))}")
    else:
        print(f"All digits are present in {mobile_number}.")
else:
    print("Invalid input. Please enter a valid 10-digit mobile number.")
```



```
/home/sjcet/datascience/venv/bin/python /home/sjcet/datascience/absent_number.py
Abin Joseph
SJC22MCA2002
2022-24
Enter a 10-digit mobile number: 9562393995
The absent digits in 9562393995 are: 0, 1, 4, 7, 8

Process finished with exit code 0
```