TASK : 2
DATE :

IMPLEMENT CONDITIONAL, CONTROL & LOOPING STATEMENTS

2.1 FACTORIAL OF A NUMBER

AIM :-

Calculate factorial of a number using while loop.

ALGORITHMS :-

Step 1 :- Start
Step 2 :- Input a number num
Step 3 :- Initialize factorial = 1
Step 4 :- Repeat for i from 1 to num
- multiply factorial = factorial × i

Step 5 :- End the loop
Step 6 :- Display the value of factorial
Step 7 :- Stop

Sample Input :-

Enter a number : 5

Sample Output :-

Factorial of 5 is 120

RESULT :-

Thus a Python Program to calculate factorial of a number

Program :-

```
num = int( input ("Enter a number :"))
factorial = 1
i
    factorial * = i
    i + = 1
Print ("Factorial of ", num, "is", factorial)
```

```
i = 1
while i <= num :
    factorial * = i
    i + = 1
Print (" Factorial of ", num, "is", factorial)
```

2.2 Counting The Non-Repeated Digits In A Given Number

Aim:-

Write a program to find the count of non-repeated digits in a given number N. The number will be passed to the program as an input of type int.

Algorithm:-

Step 1:- Start

Step 2:- Input a number num from the user.

Step 3:- Convert the number into a string num_str to easily access each digit.

Step 4:- Initialize a variable count = 0 to store the number of non-repeated digits.

Step 5:- Repeat for each digit in num_str:

- check if the count of digit in num_str is equal to 1.
- If yes, increment count by 1.

Step 6:- After loop ends, display count as the count of non-repeated digits.

Step 7:- Stop

Sample Input:-

Enter a number : 12341

Sample Output:-

Count of non-repeated digits : 3

Result:-

Thus a Python Program to count the non-repeated digits is compiled and executed successfully.

PROGRAMS :-

```
num = int(input ("Enter a number :"))
num_str = str(num)
count = 0
for digit in num_str:
    if num_str.count(digit) == 1 :
        count += 1
print(" Count of non-repeated digits :", count)
```

```
n = int (input ( " Enter the number:"))
count = [0] * 10
for d in str(n):
    count [ int(d)] + = 1
non_repeated = 0
for i in count:
    if i = = 1:
        non_repeated + = 1
Print (non_repeated)
```

2.3 MULTIPLICATION TABLE GENERATOR

AIM :- Write a program to print multiplication table of a number up to 10.

ALGORITHM :-

Step 1 :- Start
Step 2 :- Input a number num from the user.
Step 3 :- Display a message : "Multiplication table of num".
Step 4 :- Initialize a loop variable i = 1.
Step 5 :- Repeat steps 6 - 7 while i ≤ 10:

- calculate product = num × i
- print the result in the format num × i = product
- Increment i by 1

Step 6 :- End Loop
Step 7 :- Stop.

Sample Input :-

Enter a number : 5

Sample Output :-

Multiplication table of 5

5 × 1 = 5
5 × 2 = 10
5 × 3 = 15
5 × 4 = 20
5 × 5 = 25
5 × 6 = 30
5 × 7 = 35
5 × 8 = 40
5 × 9 = 45
5 × 10 = 50

RESULT :- Thus a Python Program to print multiplication table of a number up to 10 is compiled and executed successfully.

PROGRAM:-

```
num = int(input("Enter a number:"))
print("Multiplication table of", num)
for i in range(1,11):
    print(num, "x", i, "=", num*i)
```

```
n = int(input())
count = [0]*10
for d in str(n):
    count[int(d)] += 1
non_repeated = 0
for i in count:
    if i == 1:
        non_repeated += 1
print(non_repeated)
```

2.3

AIM:-

table

ALGOR

2.4 AUTOMORPHIC NUMBER

AIM :-

An Write a program to display if the number is a number whose square ends with the number itself.

ALGORITHM :-

Step 1 :- Start

Step 2 :- Input a number num from the user.

Step 3 :- Calculate the square of the number :

- Square = num x num

Step 4 :- Convert both square & num to strings

Step 5 :- check if the string of square ends with the string of num :

- if true, display "Automorphic".
- Else, display "Not Automorphic".

Step 6 :- Stop

Input :-

Enter a number : 25

Output :-

Automorphic

RESULT :-

Thus a Program to display if a number is automorphic or not is compiled and executed successfully.

PROGRAMS :-

```
num = int( input(" Enter a number : "))
square = num * num
if(          num) % (10 ** len(
temp = num
power = 1
while temp > 0:
    power = power * 10
    temp = temp // 10
if square % power == num:
    print (" Automorphic")
else:
    Print ("Not Automorphic")
```

2.4

AIM :-

An
number
number

ALGOR

St

strin

In

O

2.5 Counting the Number of Prime Numbers in a Specified Range.

AIM:- Write a program to find the count of the number of prime numbers in a specified range

ALGORITHM:-

Step 1:- Start
Step 2:- Input the starting number start
Step 3:- Input the ending number end
Step 4:- Set prime_count = 0
Step 5:- Repeat for each number num from start to end:
a) if num > 1, then check if it is prime:
- Set a flag to true
- Repeat for each i from 2 to √num:
  - if num % i == 0; it is not prime
  - break the loop
- If no divisor is found, increase prime_count by 1

Step 6:- After the loop ends, display prime_count

Step 7:- Stop

VELTECH	
EX No.	2
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	20
SIGN WITH DATE	

Sample Input:-

Enter the starting number : 10
Enter the Ending number : 20

Sample Output:-

Number of prime numbers in the range : 4

RESULT:- Thus a program to count the number of prime numbers is compiled & executed successfully.

PROGRAMS:-

```
start = int(input(" Enter the starting number:"))
end = int(input(" Enter the ending number:"))
Prime_count = 0
for num in range(start, end +1):
    if num > 1:
        for i in range(2, int(num ** 0.5) +1):
            if num % i == 0:
                break
            else:
                Prime_count += 1
Print(" Number of Prime numbers in the range:"
    Prime_count)
```