PRACTICAL : 1

**AIM:** Implement and analyze algorithms given below

* 1. Factorial (Iterative and Recursive).

**PROGRAM(Iterative):** #include <iostream> using namespace std;

int main()

{

int i,fact=1,number,counter=0; cout<<"Enter any Number: "; cin>>number; for(i=1;i<=number;i++){

fact=fact\*i; counter++;

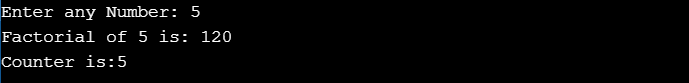
}

cout<<"Factorial of " <<number<<" is: "<<fact<<endl; cout<<"Counter is:"<<counter;

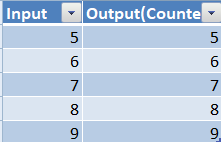
return 0;

}

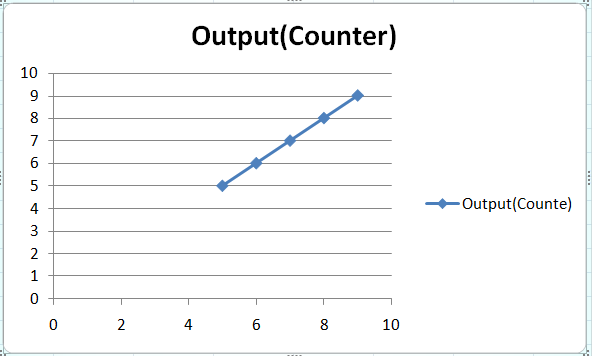
# OUTPUT:



**ANALYSIS TABLE:**



**GRAPH:**



**CONCLUSION:** I Implemented and analyzed algorithms given below 1 Factorial using iterative method.

# PROGRAM(Recursive):

#include<iostream> using namespace std; int ctr=0;

int factorial(int n)

{

if(n<0)

return(-1); /\*Wrong value\*/ if(n==0)

return(1); /\*Terminating condition\*/ else

{

ctr++; return(n\*factorial(n-1));

}

}

int main()

{

int fact,number,counter;

cout<<"Enter the number to find it's factorial: "; cin>>number;

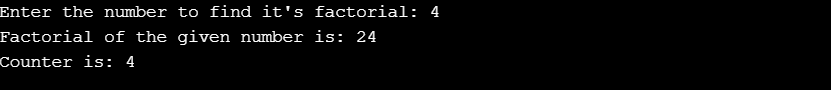
fact=factorial(number);

cout<<"Factorial of the given number is: "<<fact<<endl; cout<<"Counter is: "<<ctr<<endl;

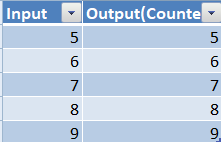
return 0;

}

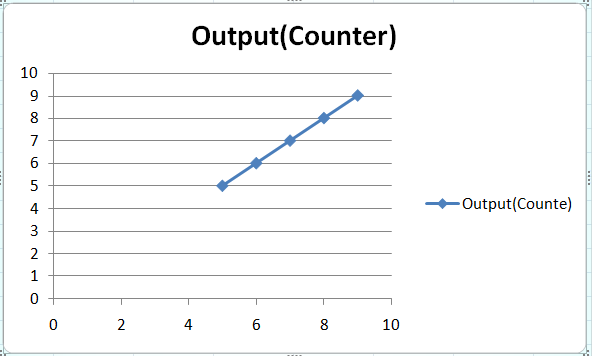
# OUTPUT:



**ANALYSIS TABLE:**



**GRAPH:**



**CONCLUSION:** I Implemented and analyzed algorithms given below 1 Factorial using Recursive method.

# Euclidean Algorithm

**PROGRAM:-**

#include<stdio.h>

int counter = 0; int gcd(int x,int y)

{

while(y!=0)

{

counter

++; int temp

= x%y;

x

= y; y

=

temp;

}

return x;

}

int main() {

int n1,n2,result; printf("Enter First

Integer: "); scanf("%d",&n1);

printf("\nEnter Second Integer: "); scanf("%d",&n2);

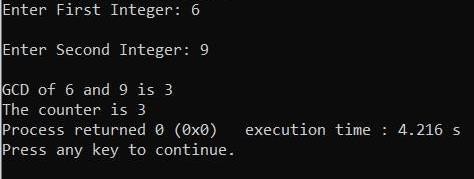
result = gcd(n1,n2); printf("\nGCD of %d and %d is

%d",n1,n2,result); printf("\nThe counter is

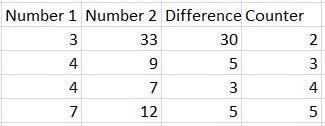
%d",counter);

}

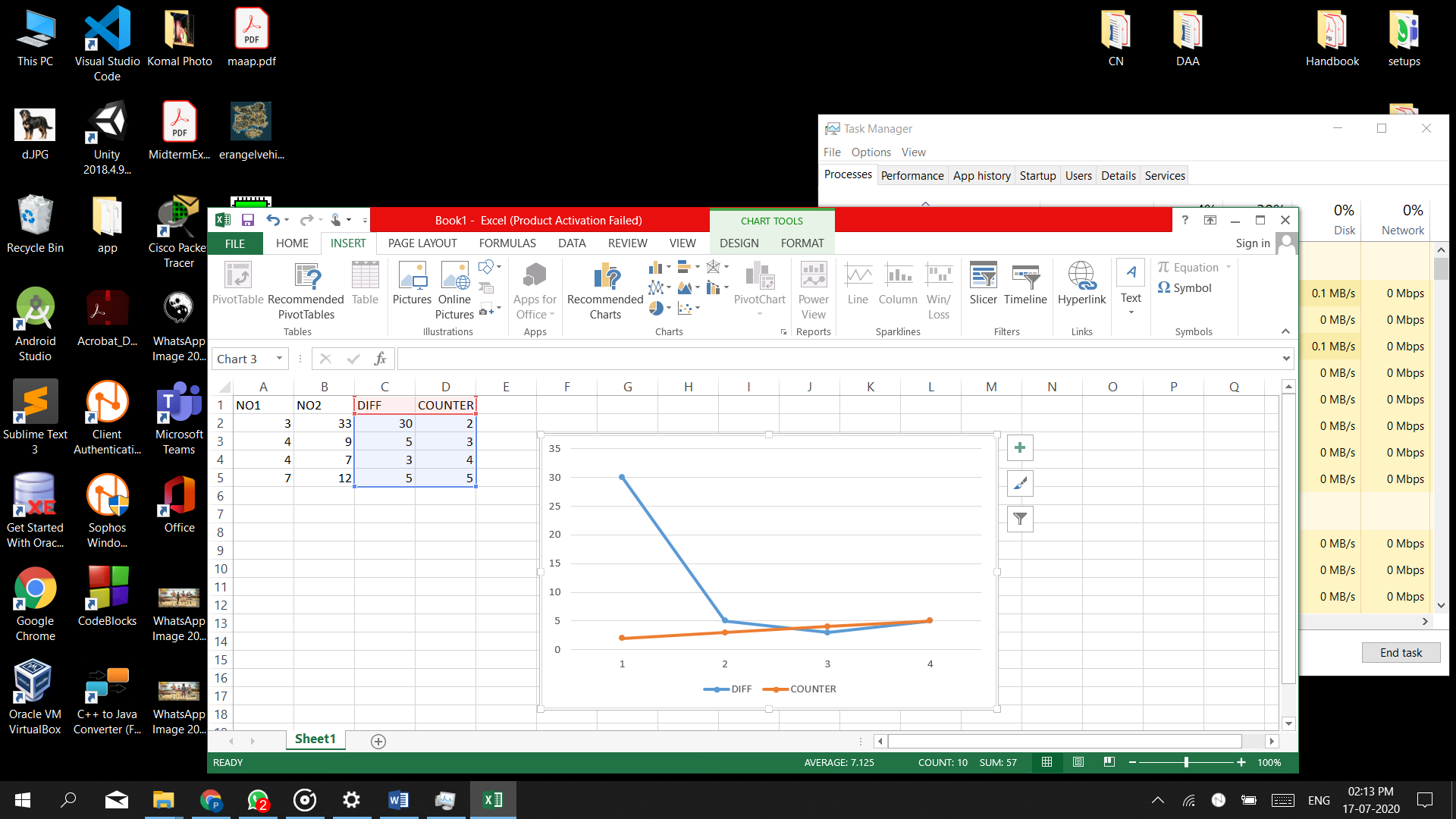
# OUTPUT:



**ANALYSIS TABLE:-**



**GRAPH:-**

****

**CONCLUSION:-**

From this practical, I learnt how to code GCD for Euclidean algorithm and find its count it

requires to find the solution. I was also able to do analysis and develop a graph for the same.