**CHAROTAR UNIVERSITY OF SCIENCE &**

**TECHNOLOGY**

**DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH**

**Computer Science & Engineering**

**NAME: PARTH NITESHKUMAR PATEL**

**ID: 19DCS098**

**SUBJECT: DESIGN AND ANALYSIS OF**

**ALGORITHM**

**CODE: CS 351**

# PRACTICAL-1

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

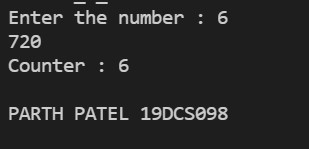
1. Factorial (Iterative and Recursive) of a number.

**PROGRAM CODE:**

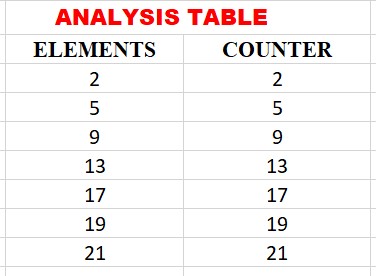
**1. ITERATIVE:**

|  |
| --- |
| #include<iostream> using namespace std; int main()  { int counter=0; double number; cout<<"Enter the number : "; // INPUT cin>>number;  double fact=1;  for(double i=number;i>=1;i--){ //MAIN LOGIC fact\*=i; counter++;  } cout<<fact<<endl; cout<<"Counter : "<<counter<<endl;  cout<<endl; |
| cout<<"PARTH PATEL "<<"19DCS098"<<endl;  return 0;  } |

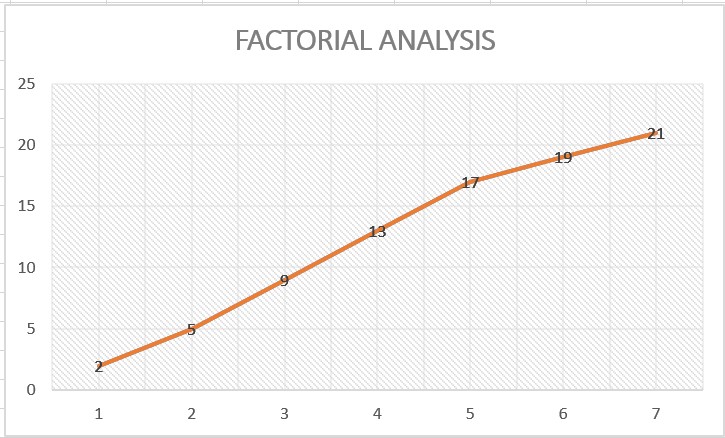
**OUTPUT:**



**TABLE:**



**GRAPH:**



|  |
| --- |
| [CS 351] DESIGN AND ANALYSIS OF ALGORITHM 19DCS098    **RECURSIVE:**    **PROGRAM CODE:**    #include<iostream> using namespace std;  static int counter=0; int factorial(int number){ if(number==1){ counter++; return 1;  } else{ counter++; return number\*factorial(number-1);  }  }  int main()  { int number; cout<<"Enter the number : "; cin>>number;  cout<<factorial(number)<<endl;  cout<<"COUNTER : "<<counter<<endl;  cout<<endl; cout<<"PARTH PATEL"<<"19DCS098"<<endl; return 0;  }          DEPSTAR CSE 5 |

CS 351] DESIGN AND ANALYSIS OF ALGORITHM

[

19

DCS

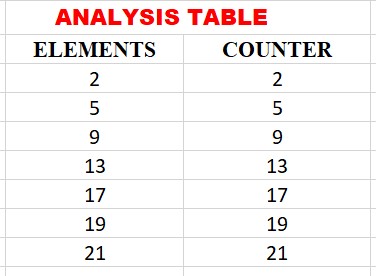
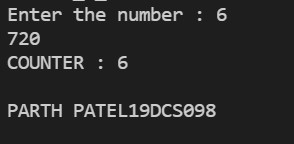
098

DEPSTAR CSE

6

**OUTPUT:**

**TABLE:**



CS 351] DESIGN AND ANALYSIS OF ALGORITHM

[

19

DCS

098

DEPSTAR CSE

7

**GRAPH:**

**CONCLUSION:**

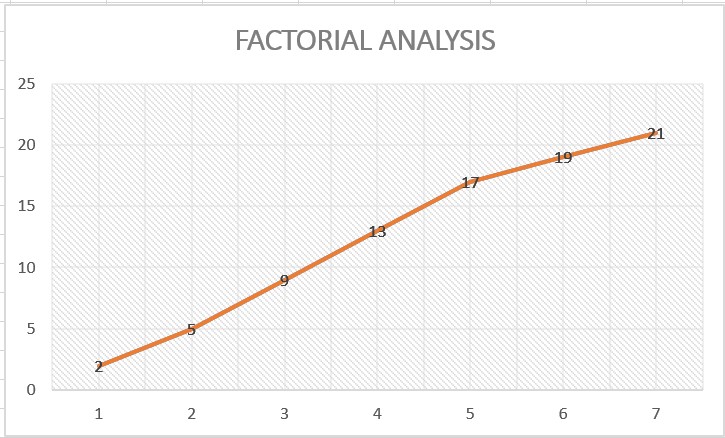
From the above practical, we can analyze that the program of factorial performed

by either Iterative method or by recursion has same time complexity i.e. O(n

)

as

it is directly proportional to n.



|  |
| --- |
| [CS 351] DESIGN AND ANALYSIS OF ALGORITHM 19DCS098      **2.Euclidean algorithm.**    **PROGRAM CODE:**    #include<iostream> using namespace std;  static int counter=0; int gcd(int a,int b){  if(a==0) { counter++; return b;  } else { counter++;  return gcd(b%a,a);  } }  int main()  { int a,b;  cout<<"Enter the numbers : (b>a) : "; cin>>a>>b;  cout<<"GCD : "<<gcd(a,b)<<endl;  cout<<"COUNTER : "<<counter<<endl;  cout<<endl;  cout<<"PARTH PATEL "<<"19DCS098"<<endl;    return 0;  DEPSTAR CSE 8 |

[

CS 351] DESIGN AND ANALYSIS OF ALGORITHM

DCS

19

098

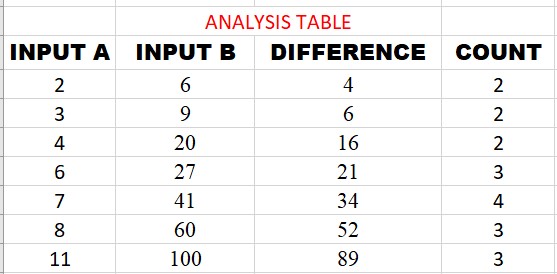
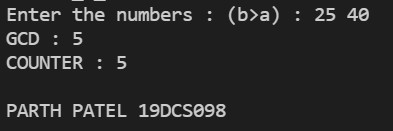
DEPSTAR CSE

9

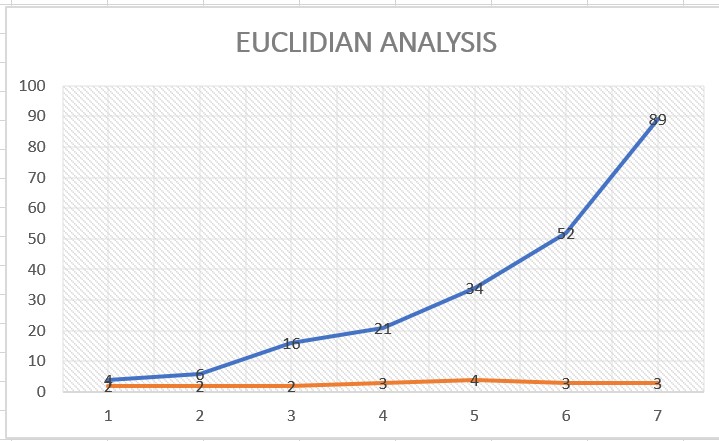
}

**OUTPUT:**

**TABLE:**



**GRAPH:**

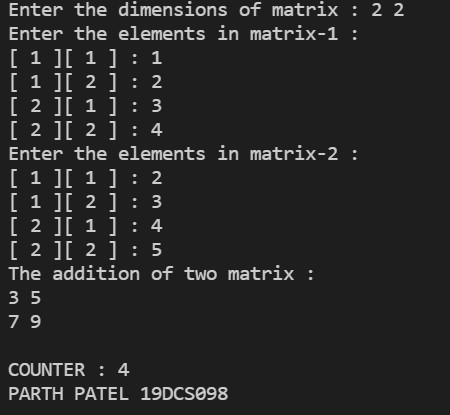


**CONCLUSION:**

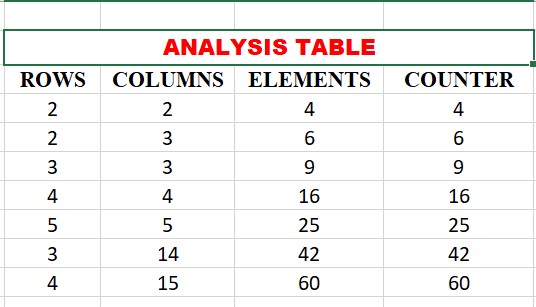
From this practical, we can analyze that for the Euclidian program the graph of difference vs count which is non-linear.

|  |  |
| --- | --- |
| [CS 351] DESIGN AND ANALYSIS OF ALGORITHM 19DCS098      **3.Matrix Addition and Matrix Multiplication:**    **MATRIX ADDITION:**    #include<iostream> using namespace std;  static int counter=0; int main()  { int m,n; cout<<"Enter the dimensions of matrix : "; cin>>m>>n; int matrix1[m][n],matrix2[m][n],sum[m][n];  cout<<"Enter the elements in matrix-1 :"<<endl;  for(int i=0;i<m;i++){ for(int j=0;j<n;j++)  { cout<<"[ "<<i+1<<" ]"<<"[ "<<j+1<<" ] : "; cin>>matrix1[i][j];  }  } cout<<"Enter the elements in matrix-2 :"<<endl;  for(int i=0;i<m;i++){ for(int j=0;j<n;j++)  { cout<<"[ "<<i+1<<" ]"<<"[ "<<j+1<<" ] : "; cin>>matrix2[i][j];  }  }  for(int i=0;i<m;i++){  DEPSTAR CSE 11 | |
| for(int j=0;j<n;j++){ sum[i][j]=matrix1[i][j]+matrix2[i][j];  counter++;}  } cout<<"The addition of two matrix : "<<endl; for(int i=0;i<m;i++){ for(int j=0;j<n;j++){ cout<<sum[i][j]<<" ";  }  cout<<endl;  } cout<<endl; cout<<"COUNTER : "<<counter<<endl; cout<<"PARTH PATEL 19DCS098"<<endl; return 0;    } | |

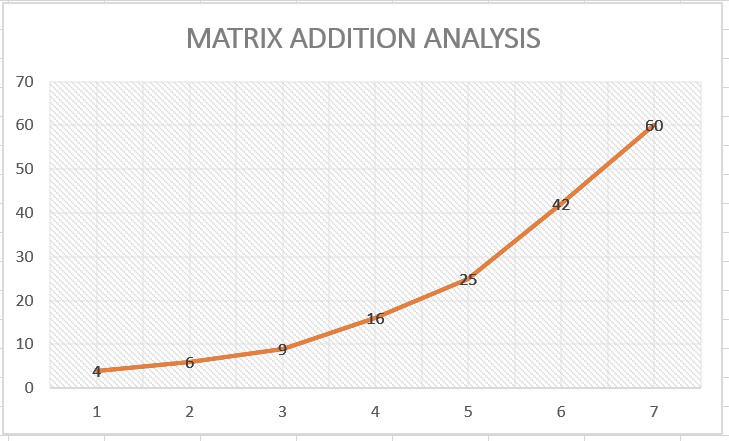
**OUTPUT:**



**TABLE:**



**GRAPH:**



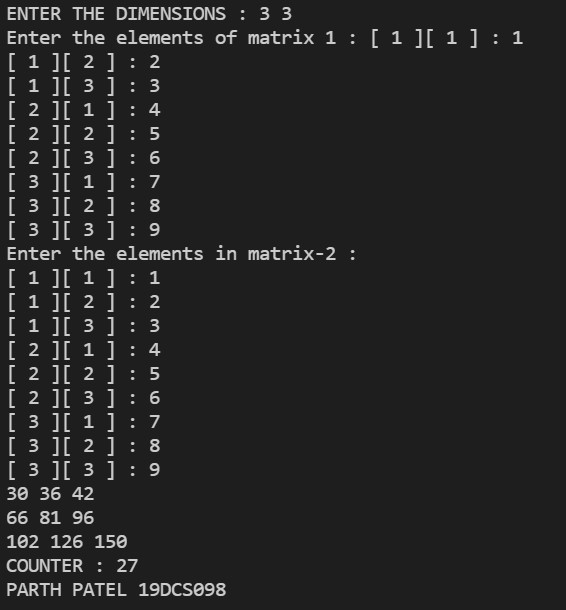
**CONCLUSION:**

It is clear from the practical that counter increases linearly with number of elements

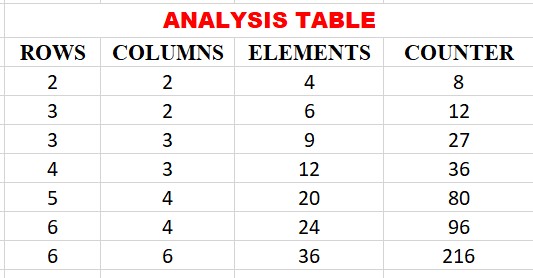
Time Complexity of matrix addition is O(n2) as we have to visit NxN elements exactly once.

|  |  |
| --- | --- |
| [CS 351] DESIGN AND ANALYSIS OF ALGORITHM 19DCS098    **MATRIX MULTIPLICATION:**    **PROGRAM CODE:**    #include<iostream> using namespace std;  static int counter=0; int main()  { int m,n; cout<<"ENTER THE DIMENSIONS : "; cin>>m>>n; int matrix1[m][n],matrix2[m][n],result[m][n];  cout<<"Enter the elements of matrix 1 : "; for(int i=0;i<m;i++){ for(int j=0;j<n;j++)  { cout<<"[ "<<i+1<<" ]"<<"[ "<<j+1<<" ] : "; cin>>matrix1[i][j];  }  } cout<<"Enter the elements in matrix-2 :"<<endl;  for(int i=0;i<m;i++){ for(int j=0;j<n;j++)  { cout<<"[ "<<i+1<<" ]"<<"[ "<<j+1<<" ] : "; cin>>matrix2[i][j];  }  }  for(int i=0;i<m;i++)  { for(int j=0;j<n;j++){  DEPSTAR CSE 16 | |
| result[i][j]=0; for(int k=0;k<n;k++)  { result[i][j]+=matrix1[i][k]\*matrix2[k][j];  counter++;  }  }  } for(int i=0;i<m;i++){ for(int j=0;j<n;j++) cout<<result[i][j]<<" ";  cout<<endl;  } cout<<"COUNTER : "<<counter<<endl; cout<<"PARTH PATEL 19DCS098"<<endl; return 0;  } | |

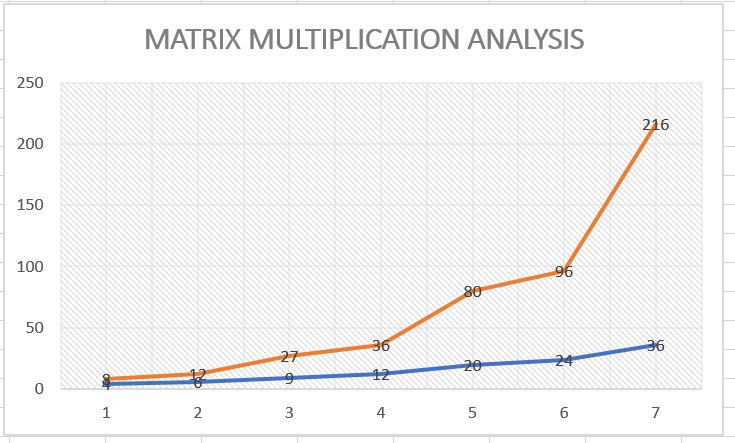
**OUTPUT:**



**TABLE:**



**GRAPH:**



**CONCLUSION:**

By Performing the above practical, it is clear that the counter increases disproportionately with number of elements.

Time Complexity will be almost O(n2.37)

**PRACTICAL-1.4**

**AIM:**

Find a subset of a given set S={s1,s2,.....,sn} of n positive integers whose sum is equal to a given positive integer d. For example, if S= {1, 2, 5, 6, 8} and d = 9 there are two solutions {1,2,6} and {1,8}.A suitable message is to be displayed if the given problem instance doesn't have a solution.

**PROGRAM CODE:**

#include <iostream>

using namespace std;

static int counter = 0;

bool isSubsetSum(int set[], int n, int sum)

{   counter++;

    if (sum == 0)

    return true;

    if (n == 0 && sum != 0)

        return false;

    if (set[n -1] > sum)

         return isSubsetSum(set, n -1, sum);

    return isSubsetSum(set, n -1, sum) || isSubsetSum(set, n -1, sum -set[n -1]);

    }

int main()

{

    int n;

    int sum;

    cout<<"Enter the size of array : ";

    cin>>n;

    int set[n];

    cout<<"Enter the elements : ";

    for(int i=0;i<n;i++)

    cin>>set[i];

    cout<<"Enter the sum : ";

    cin>>sum;

    if (isSubsetSum(set, n, sum) == true)

        cout <<"SUBSET EXISTS"<<endl;

    else

        cout << "NO SUBSET EXISTS"<<endl;

        cout <<"Counter : " << counter<<endl;

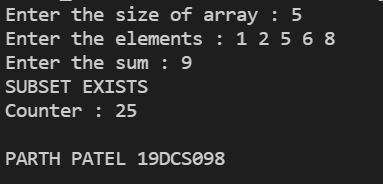
        cout <<endl;

        cout<<"PARTH PATEL 19DCS098"<<endl;

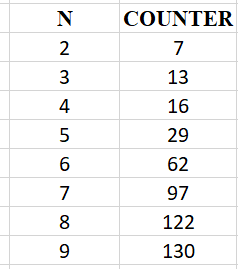
        return 0;

}

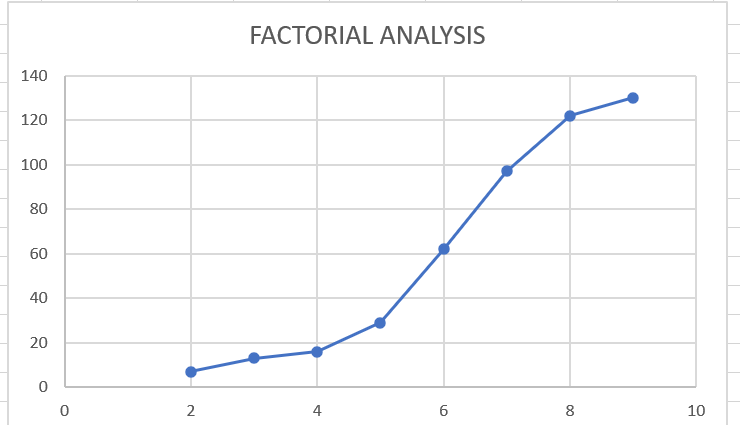
**OUTPUT:**



**TABLE:**



**GRAPH:**

****