VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Analysis and Design of Algorithms

Submitted by

KOSHAL S GOYAL (1BM20CS073)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
May-2022 to July-2022

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**Analysis and Design of Algorithms**" carried out by **Koshal S Goyal(1BM20CS073),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **Analysis and Design of Algorithms - (19CS4PCADA)** work prescribed for the said degree.

Name of the Lab-In charge:

Dr Rajeshwari Bs
Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

`

Index Sheet

	and {1,8}. A suitable message is to be displayed if the given problem instance doesn't have a solution.	
18	Implement "N-Queens Problem" using Backtracking.	**82-84**

Course Outcome

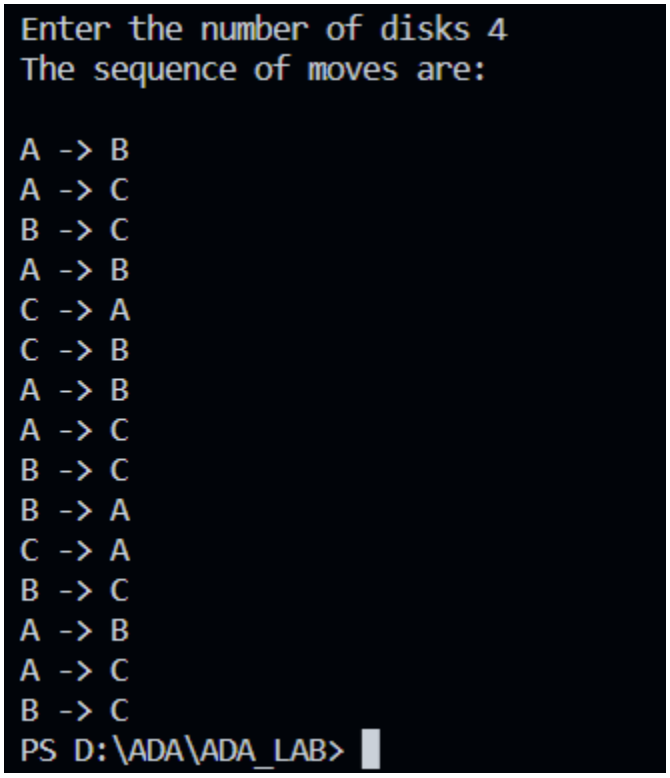
CO1	Ability to **analyze** time complexity of Recursive and Non-Recursive algorithms using asymptotic notations.
CO2	Ability to **design** efficient algorithms using various design techniques.
CO3	Ability to **apply** the knowledge of complexity classes P, NP, and NP-Complete and prove certain problems are NP-Complete
CO4	Ability to **conduct** practical experiments to solve problems using an appropriate designing method and find time efficiency.

1.Write a recursive program to Solve:

a)Tower of Hanoi:

```
#include<stdio.h>
void toh(int n,char src, char dest, char aux){
   if(n==1){
      printf("\n%c -> %c",src,dest);
      return;
   }
   else{
      toh(n-1,src,aux,dest);
      printf("\n%c -> %c",src,dest);
      toh(n-1,aux,dest,src);
   }
}
int main(){
   int n;
   printf("\nEnter the number of disks");
   scanf("%d",&n);
   printf("The sequence of moves are:\n");
   toh(n,'A','C','B');
}
```

Output:



b)To find GCD:

```
#include<stdio.h>
int gcd(int m,int n){
   if(n==0){
      return m;
   }
   else{
      return(gcd(n,(m%n)));
```

```
    }
}
int main(){
    int n,m;
    int GCD;
    printf("Enter the values:");
    scanf("%d %d",&m,&n);
    GCD=gcd(m,n);
    printf("The gcd of %d and %d is %d",m,n,GCD);
}
```

Output:

1.

```
Enter the values:36 48
The gcd of 36 and 48 is 12
PS D:\ADA\ADA_LAB>
```

2.

```
Enter the values:38 92
The gcd of 38 and 92 is 2
PS D:\ADA\ADA_LAB>
```

2.Implement Recursive Binary search and Linear search and determine the time required to search an element. Repeat the experiment for different values of N and plot a graph of the time taken versus N.

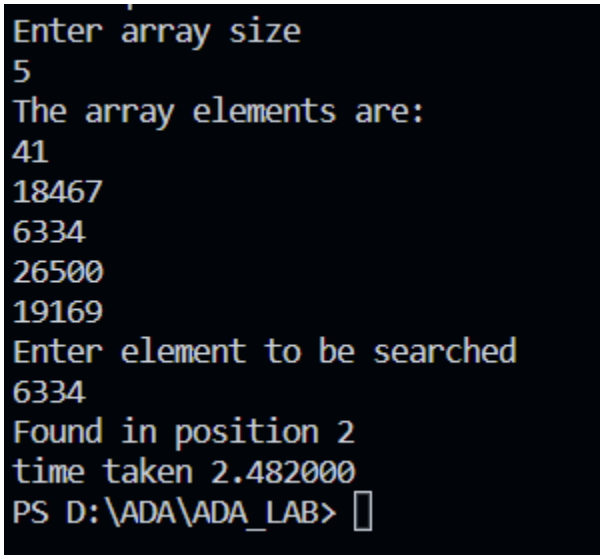
Binary search:

```
#include <stdio.h>
#include <time.h>
int binary(int element,int arr[], int start_index, int end_index){
  if (end_index >= start_index){
    int middle = start_index + (end_index - start_index )/2;
    if (arr[middle] == element)
      return middle;
    if (arr[middle] > element)
      return binary( element,arr, start_index, middle-1);
    return binary(element,arr, middle+1, end_index);
  }
  return -1;
}
int main()
{
   clock_t start,end;
   int n;
   int s;
```

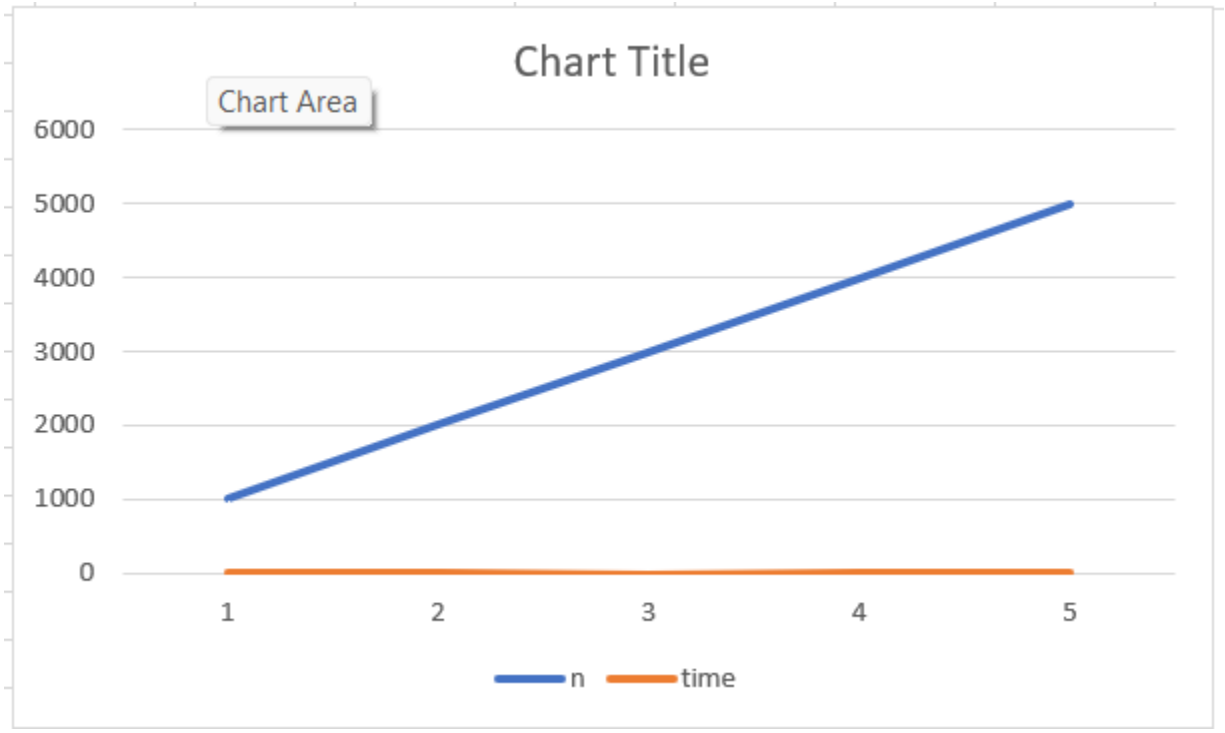
```
printf("Enter array size\n");
scanf("%d",&n);
int arr[n];
for(int i=0;i<n;i++){
   arr[i] = rand();
}
printf("The array elements are:");
for(int i=0;i<n;i++){
   printf("\n%d",arr[i]);
}
printf("\nEnter element to be searched\n");
scanf("%d",&s);
start=clock();
int res= binary(s,arr,0,n-1);
if(res==-1)
{
   printf("Element not found");
}
else
printf("Found in position %d", res);
for(int i=0;i<1000;i++){
   for(int j=0;j<1000000;j++){
```

```
        }
      }
      end=clock();
      printf("\ntime taken %f ", difftime(end,start)/CLOCKS_PER_SEC);
    }
```

Output:



Graph:



Linear Search:

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
void main()
{
   int n;
   printf("Enter size of array:\n");
   scanf("%d",&n);
   int a[n];
   time_t st,ed;
```

```
int ele,flag = 0;

for(int i = 0;i<n;i++)
{
   a[i] = rand();
}

for(int k = 0;k<500;k++)
{
   printf("%d,",a[k]);
}

printf("\n");
printf("ENTER ELEMENT TO SEARCH \n");
scanf("%d",&ele);
st = time(NULL);

for(int j = 0;j<n;j++)
{
   for(int p = 0;p<10000000;p++);
   if(a[j] == ele)
   {
      printf("\n ELEMENT FOUND");
```

```
            flag = 1;
            break;
        }
    }

    if(flag == 0)
    {
        printf("\n ELEMENT NOT FOUND");
    }
    ed = time(NULL);

    printf("\n TIME TAKEN = %f", difftime(ed,st));
    return 0;
}
```

Output:

```
Enter size of array:
1000
41,18467,6334,26500,19169,15724,11478,29358,26962,24464,5705,28145,23281,16827,9961,49
1,2995,11942,4827,5436,32391,14604,3902,153,292,12382,17421,18716,19718,19895,5447,217
26,14771,11538,1869,19912,25667,26299,17035,9894,28703,23811,31322,30333,17673,4664,15
141,7711,28253,6868,25547,27644,32662,32757,20037,12859,8723,9741,27529,778,12316,3035
,22190,1842,288,30106,9040,8942,19264,22648,27446,23805,15890,6729,24370,15350,15006,3
1101,24393,3548,19629,12623,24084,19954,18756,11840,4966,7376,13931,26308,16944,32439,
24626,11323,5537,21538,16118,2082,22929,16541,4833,31115,4639,29658,22704,9930,13977,2
306,31673,22386,5021,28745,26924,19072,6270,5829,26777,15573,5097,16512,23986,13290,91
61,18636,22355,24767,23655,15574,4031,12052,27350,1150,16941,21724,13966,3430,31107,30
191,18007,11337,15457,12287,27753,10383,14945,8909,32209,9758,24221,18588,6422,24946,2
7506,13030,16413,29168,900,32591,18762,1655,17410,6359,27624,20537,21548,6483,27595,40
41,3602,24350,10291,30836,9374,11020,4596,24021,27348,23199,19668,24484,8281,4734,53,1
999,26418,27938,6900,3788,18127,467,3728,14893,24648,22483,17807,2421,14310,6617,22813
,9514,14309,7616,18935,17451,20600,5249,16519,31556,22798,30303,6224,11008,5844,32609,
14989,32702,3195,20485,3093,14343,30523,1587,29314,9503,7448,25200,13458,6618,20580,19
796,14798,15281,19589,20798,28009,27157,20472,23622,18538,12292,6038,24179,18190,29657
,7958,6191,19815,22888,19156,11511,16202,2634,24272,20055,20328,22646,26362,4886,18875
,28433,29869,20142,23844,1416,21881,31998,10322,18651,10021,5699,3557,28476,27892,2438
9,5075,10712,2600,2510,21003,26869,17861,14688,13401,9789,15255,16423,5002,10585,24182
,10285,27088,31426,28617,23757,9832,30932,4169,2154,25721,17189,19976,31329,2368,28692
,21425,10555,3434,16549,7441,9512,30145,18060,21718,3753,16139,12423,16279,25996,16687
,12529,22549,17437,19866,12949,193,23195,3297,20416,28286,16105,24488,16282,12455,2573
4,18114,11701,31316,20671,5786,12263,4313,24355,31185,20053,912,10808,1832,20945,4313,
27756,28321,19558,23646,27982,481,4144,23196,20222,7129,2161,5535,20450,11173,10466,12
044,21659,26292,26439,17253,20024,26154,29510,4745,20649,13186,8313,4474,28022,2168,14
018,18787,9905,17958,7391,10202,3625,26477,4414,9314,25824,29334,25874,24372,20159,118
33,28070,7487,28297,7518,8177,17773,32270,1763,2668,17192,13985,3102,8480,29213,7627,4
802,4099,30527,2625,1543,1924,11023,29972,13061,14181,31003,27432,17505,27593,22725,13
031,8492,142,17222,31286,13064,7900,19187,8360,22413,30974,14270,29170,235,30833,19711
,25760,18896,4667,7285,12550,140,13694,2695,21624,28019,2125,26576,21694,22658,26302,1
7371,22466,4678,22593,23851,25484,1018,28464,21119,23152,2800,18087,31060,1926,9010,47
57,32170,20315,9576,30227,12043,22758,7164,5109,7882,17086,29565,3487,29577,14474,2625
,25627,5629,31928,25423,28520,6902,14962,123,24596,3737,13261,10195,32525,
ENTER ELEMENT TO SEARCH
32170

 ELEMENT FOUND
 TIME TAKEN = 12.000000
PS D:\ADA\ADA LAB>
```

Graph:

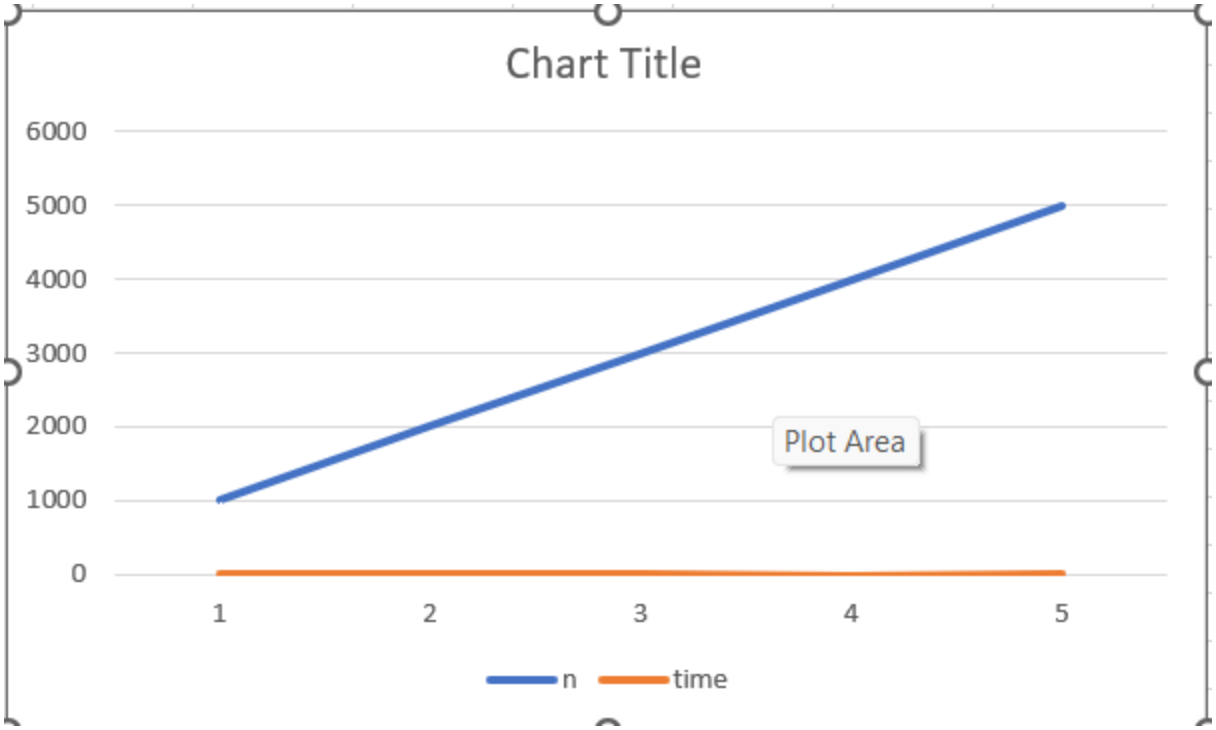


Chart Title
6000
5000
4000
3000
2000
1000
0
1
2
3
4
5
Plot Area
n
time

3)Sort a given set of N integer elements using Selection Sort technique and compute its time taken. Run the program for different values of N and record the time taken to sort.

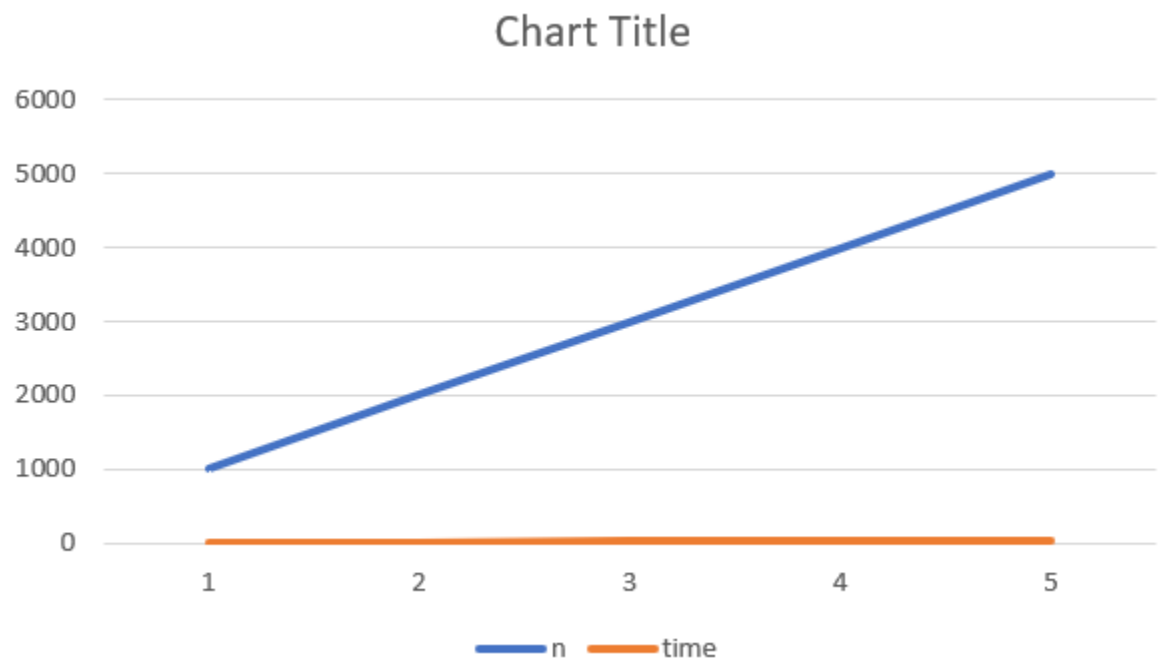
```
#include <stdio.h>
#include <time.h>
int main()
{
   int a[100], n, i, position, swap, j;
   clock_t start, end;
   printf("Enter the number of elements");
   scanf("%d", &n);
   printf("Enter %d numbers", n);
   for (i=0; i<n; i++)
   {
      scanf("%d", &a[i]);
   }
   start = clock();
   for (i=0; i<n-1; i++)
   {
      position = i;
      for (j=i+1; j<n; j++)
      {
```

```
            if(a[position]>a[j])
            {position = j;}
        }
        if (position != j)
        {
            swap = a[i];
            a[i] = a[position];
            a[position] = swap;
        }
    }
    end = clock();
    printf("Sorted Array\n");
    for (i=0; i<n; i++)
    {
        printf("%d ", a[i]);
    }
    printf("Time is %f", difftime(end, start)/CLOCKS_PER_SEC);
    return 0;
}
```

Output:

```
Enter the number of elements5
Enter 5 numbers4
2
1
6
3
Sorted Array
1 2 3 4 6 Time is 0.000000
PS D:\ADA\ADA_LAB>
```

Graph:



4)Write program to do the following:

a) Print all the nodes reachable from a given starting node in a digraph using BFS method.

b) Check whether a given graph is connected or not using DFS method.

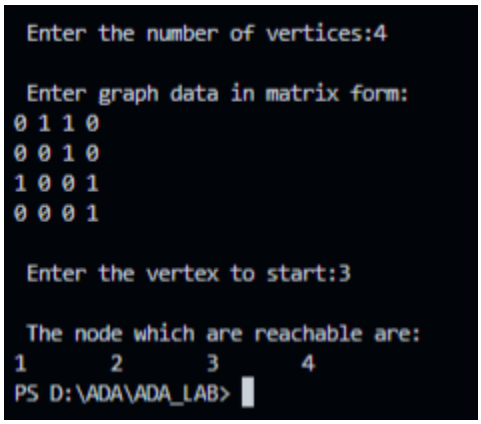
4.a)

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

int a[20][20],s[20];

int visited[20],n,i,j,f=0,r=-1;

void bfs(int v)
{
   for(i=1; i<=n; i++)
      if(a[v][i] && !visited[i])
         s[++r]=i;
   if(f<=r)
   {
      visited[s[f]]=1;
      bfs(s[f++]);
```

```
    }
}
void main()
{
    int v;
    printf("\n Enter the number of vertices:");
    scanf("%d",&n);
    for(i=1; i<=n; i++)
    {
        s[i]=0;
        visited[i]=0;
    }
    printf("\n Enter graph data in matrix form:\n");
    for(i=1; i<=n; i++)
        for(j=1; j<=n; j++)
            scanf("%d",&a[i][j]);
    printf("\n Enter the vertex to start:");
    scanf("%d",&v);
    bfs(v);
    printf("\n The node which are reachable are:\n");
    for(i=1; i<=n; i++)
        if(visited[i])
            printf("%d\t",i);
```

}

Output:



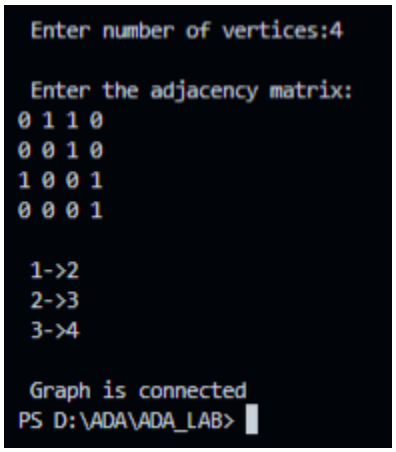
4.b)

```
#include<conio.h>
#include<stdio.h>
int a[20][20],reach[20],n;
```

```
void dfs(int v)
{
   int i;
   reach[v]=1;
   for(i=1; i<=n; i++)
      if(a[v][i] && !reach[i])
      {
         printf("\n %d->%d",v,i);
         dfs(i);
      }
}
void main()
{
   int i,j,count=0;
   printf("\n Enter number of vertices:");
   scanf("%d",&n);
   for(i=1; i<=n; i++)
   {
      reach[i]=0;
      for(j=1; j<=n; j++)
         a[i][j]=0;
   }
   printf("\n Enter the adjacency matrix:\n");
```

```
    for(i=1; i<=n; i++)
        for(j=1; j<=n; j++)
            scanf("%d",&a[i][j]);
    dfs(1);
    printf("\n");
    for(i=1; i<=n; i++)
    {
        if(reach[i])
            count++;
    }
    if(count==n)
        printf("\n Graph is connected");
    else
        printf("\n Graph is not connected");
}
```

Output:



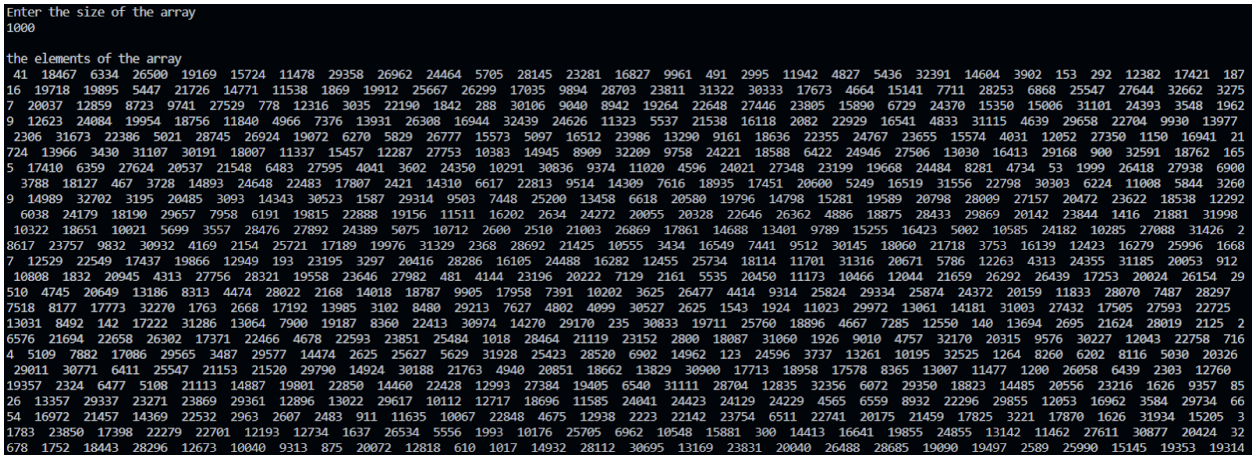
5)Sort a given set of N integer elements using Insertion Sort technique and compute its time taken.

```
#include <stdio.h>
#include <time.h>
void insertionSort(int arr[], int n)
{
   int i, key, j;
   for (i = 1; i < n; i++) {
      key = arr[i];
      j = i - 1;
      while (j >= 0 && arr[j] > key) {
         arr[j + 1] = arr[j];
         j = j - 1;
      }
      arr[j + 1] = key;
   }
}

int main()
{
   int n;
   clock_t start,end;
```

```
   printf("Enter the size of the array\n");
   scanf("%d",&n);

   int arr[n];
   for(int i=0;i<n;i++){
      arr[i]=rand();
   }
   printf("\nthe elements of the array\n");
   for(int i=0;i<n;i++){
      printf(" %d ",arr[i]);
   }

start=clock();
   insertionSort(arr, n);
   end=clock();

   printf("\Sorted array: ");
   for (int j = 0; j < n; j++)
      printf("%d ", arr[j]);
   printf("\n");
printf("\ntime taken %f ", difftime(end,start));

   return 0;
```

}

Output:


Enter the size of the array
1000
the elements of the array
41 18467 6334 26500 19169 15724 11478 29358 26962 24464 5705 28145 23281 16827 9961 491 2995 11942 4827 5436 32391 14604 3902 153 292 12382 17421 187
16 19718 19895 5447 21726 14771 11538 1869 19912 25667 26299 17035 9894 28703 23811 31322 30333 17673 4664 15141 7711 28253 6868 25547 27644 32662 3275
7 20037 12859 8723 9741 27529 778 12316 3035 22190 1842 288 30106 9040 8942 19264 22648 27446 23805 15890 6729 24370 15350 15006 31101 24393 3548 1962
9 12623 24084 19954 18756 11840 4966 7376 13931 26308 16944 32439 24626 11323 5537 21538 16118 2082 22929 16541 4833 31115 4639 29658 22704 9930 13977
2306 31673 22386 5021 28745 26924 19072 6270 5829 26777 15573 5097 16512 23986 13290 9161 18636 22355 24767 23655 15574 4031 12052 27350 1150 16941 21
724 13966 3430 31107 30191 18007 11337 15457 12287 27753 10383 14945 8909 32209 9758 24221 18588 6422 24946 27506 13030 16413 29168 900 32591 18762 165
5 17410 6359 27624 20537 21548 6483 27595 4041 3602 24350 10291 30836 9374 11020 4596 24021 27348 23199 19668 24484 8281 4734 53 1999 26418 27938 6900
3788 18127 467 3728 14893 24648 22483 17807 2421 14310 6617 22813 9514 14309 7616 18935 17451 20600 5249 16519 31556 22798 30303 6224 11008 5844 3260
9 14989 32702 3195 20485 3093 14343 30523 1587 29314 9503 7448 25200 13458 6618 20580 19796 14798 15281 19589 20798 28009 27157 20472 23622 18538 12292
6038 24179 18190 29657 7958 6191 19815 22888 19156 11511 16202 2634 24272 20055 20328 22646 26362 4886 18875 28433 29869 20142 23844 1416 21881 31998
10322 18651 10021 5699 3557 28476 27892 24389 5075 10712 2600 2510 21003 26869 17861 14688 13401 9789 15255 16423 5002 10585 24182 10285 27088 31426 2
8617 23757 9832 30932 4169 2154 25721 17189 19976 31329 2368 28692 21425 10555 3434 16549 7441 9512 30145 18060 21718 3753 16139 12423 16279 25996 1668
7 12529 22549 17437 19866 12949 193 23195 3297 20416 28286 16105 24488 16282 12455 25734 18114 11701 31316 20671 5786 12263 4313 24355 31185 20053 912
10808 1832 20945 4313 27756 28321 19558 23646 27982 481 4144 23196 20222 7129 2161 5535 20450 11173 10466 12044 21659 26292 26439 17253 20024 26154 29
510 4745 20649 13186 8313 4474 28022 2168 14018 18787 9905 17958 7391 10202 3625 26477 4414 9314 25824 29334 25874 24372 20159 11833 28070 7487 28297
7518 8177 17773 32270 1763 2668 17192 13985 3102 8480 29213 7627 4802 4099 30527 2625 1543 1924 11023 29972 13061 14181 31003 27432 17505 27593 22725
13031 8492 142 17222 31286 13064 7900 19187 8360 22413 30974 14270 29170 235 30833 19711 25760 18896 4667 7285 12550 140 13694 2695 21624 28019 2125 2
6576 21694 22658 26302 17371 22466 4678 22593 23851 25484 1018 28464 21119 23152 2800 18087 31060 1926 9010 4757 32170 20315 9576 30227 12043 22758 716
4 5109 7882 17086 29565 3487 29577 14474 2625 25627 5629 31928 25423 28520 6902 14962 123 24596 3737 13261 10195 32525 1264 8260 6202 8116 5030 20326
29011 30771 6411 25547 21153 21520 29790 14924 30188 21763 4940 20851 18662 13829 30900 17713 18958 17578 8365 13007 11477 1200 26058 6439 2303 12760
19357 2324 6477 5108 21113 14887 19801 22850 14460 22428 12993 27384 19405 6540 31111 28704 12835 32356 6072 29350 18823 14485 20556 23216 1626 9357 85
26 13357 29337 23271 23869 29361 12896 13022 29617 10112 12717 18696 11585 24041 24423 24129 24229 4565 6559 8932 22296 29855 12053 16962 3584 29734 66
54 16972 21457 14369 22532 2963 2607 2483 911 11635 10067 22848 4675 12938 2223 22142 23754 6511 22741 20175 21459 17825 3221 17870 1626 31934 15205 3
1783 23850 17398 22279 22701 12193 12734 1637 26534 5556 1993 10176 25705 6962 10548 15881 300 14413 16641 19855 24855 13142 11462 27611 30877 20424 32
678 1752 18443 28296 12673 10040 9313 875 20072 12818 610 1017 14932 28112 30695 13169 23831 20040 26488 28685 19090 19497 2589 25990 15145 19353 19314

```
  18651  26740  22044  11258  335  8759  11192  7605  25264  12181  28503  3829  23775  20608  29292  5997  17549  29556  25561  31627  6467  29541  26129  31240  27813  29174
20601  6077  20215  8683  8213  23992  25824  5601  23392  15759  2670  26428  28027  4084  10075  18786  15498  24970  6287  23847  32604  503  21221  22663  5706  2363  9010
22171  27489  18240  12164  25542  7619  20913  7591  6704  31818  9232  750  25205  4975  1539  303  11422  21098  11247  13584  13648  2971  17864  22913  11075  21545  28712
 17546  18678  1769  15262  8519  13985  28289  15944  2865  18540  23245  25508  28318  27870  9601  28323  21132  24472  27152  25087  28570  29763  29901  17103  14423  3527
 11600  26969  14015  5565  28  21543  25347  2088  2943  12637  22409  26463  5049  4681  1588  11342  608  32060  21221  1758  29954  20888  14146  690  7949  12843  21430  25
620  748  27067  4536  20783  18035  32226  15185  7038  9853  25629  11224  15748  19923  3359  32257  24766  4944  14955  23318  32726  25411  21025  20355  31001  22549  9496
  18584  9515  17964  23342  8075  17913  16142  31196  21948  25072  20426  14606  26173  24429  32404  6705  20626  29812  19375  30093  16565  16036  14736  29141  30814  599
4  8256  6652  23936  30838  20482  1355  21015  1131  18230  17841  14625  2011  32637  4186  19690  1650  5662  21634  10893  10353  21416  13452  14008  7262  22233  5454  16
303  16634  26303  14256  148  11124  12317  4213  27109  24028  29200  21080  21318  16858  24050  24155  31361  15264  11903  3676  29643  26909  14902  3561  28489  24948  12
82  13653  30674  2220  5402  6923  3831  19369  3878  20259  19008  22619  23971  30003  21945  9781  26504  12392  32685  25313  6698  5589  12722  5938  19037  6410  31461  6
234  12508  9961  3959  6493  1515  25269  24937  28869  58  14700  13971  26264  15117  16215  24555  7815  18330  3039  30212  29288  28082  1954  16085  20710  24484  24774
8380  29815  25951  6541  18115  1679  17110  25898  23073  788  23977  18132  29956  28689  26113  10008  12941  15790  1723  21363  28  25184  24778  7200  5071  1885  21974
1071  11333  22867  26153  14295  32168  20825  9676  15629  28650  2598  3309  4693  4686  30080  10116  12249
Sorted array: 28 28 41 53 58 123 140 142 148 153 193 235 288 292 300 303 335 467 481 491 503 608 610 690 748 750 778 788 875 900 911 912 1017 1018 1071 1131 1150 1200 1264 1282
1355 1416 1515 1539 1543 1587 1588 1626 1626 1637 1650 1655 1679 1723 1752 1758 1763 1769 1832 1842 1869 1885 1924 1926 1954 1993 1999 2011 2082 2088 2125 2154 2161 2168 2220 22
23 2303 2306 2324 2363 2368 2421 2483 2510 2589 2598 2600 2607 2625 2625 2634 2668 2670 2695 2800 2865 2943 2963 2971 2995 3035 3039 3093 3102 3195 3221 3297 3309 3359 3430 3434
 3487 3527 3548 3557 3561 3584 3602 3625 3676 3728 3737 3753 3788 3829 3831 3878 3902 3959 4031 4041 4084 4099 4144 4169 4186 4213 4313 4313 4414 4474 4536 4565 4596 4639 4664 4
667 4675 4678 4681 4686 4693 4734 4745 4757 4802 4827 4833 4886 4940 4944 4966 4975 5002 5021 5030 5049 5071 5075 5097 5108 5109 5249 5402 5436 5447 5454 5535 5537 5556 5565 558
9 5601 5629 5662 5699 5705 5706 5786 5829 5844 5938 5994 5997 6038 6072 6077 6191 6202 6224 6234 6270 6287 6334 6359 6410 6411 6422 6439 6467 6477 6483 6493 6511 6540 6541 6559
6617 6618 6652 6654 6698 6704 6705 6729 6868 6900 6902 6923 6962 7038 7129 7164 7200 7262 7285 7376 7391 7441 7448 7487 7518 7591 7605 7616 7619 7627 7711 7815 7882 7900 7949 79
58 8075 8116 8177 8213 8256 8260 8281 8313 8360 8365 8380 8480 8492 8519 8526 8683 8723 8759 8909 8932 8942 9010 9010 9040 9161 9232 9313 9314 9357 9374 9496 9503 9512 9514 9515
 9576 9601 9676 9741 9758 9781 9789 9832 9853 9894 9905 9930 9961 9961 10008 10021 10040 10067 10075 10112 10116 10176 10195 10202 10285 10291 10322 10353 10383 10466 10548 1055
5 10585 10712 10808 10893 11008 11020 11023 11075 11124 11173 11192 11224 11247 11258 11323 11333 11337 11342 11422 11462 11477 11478 11511 11538 11585 11600 11635 11701 11833 1
1840 11903 11942 12043 12044 12052 12053 12164 12181 12193 12249 12263 12287 12292 12316 12317 12382 12392 12423 12455 12508 12529 12550 12623 12637 12673 12717 12722 12734 1276
0 12818 12835 12843 12859 12896 12938 12941 12949 12993 13007 13022 13030 13031 13061 13064 13142 13169 13186 13261 13290 13357 13401 13452 13458 13584 13648 13653 13694 13829 1
3931 13966 13971 13977 13985 13985 14008 14015 14018 14146 14181 14256 14270 14295 14309 14310 14343 14369 14413 14423 14460 14474 14485 14604 14606 14625 14688 14700 14736 1477
1 14798 14887 14893 14902 14924 14932 14945 14955 14962 14989 15006 15117 15141 15145 15185 15205 15255 15262 15264 15281 15350 15457 15498 15573 15574 15629 15724 15748 15759 1
5790 15881 15890 15944 16036 16085 16105 16118 16139 16142 16202 16215 16279 16282 16303 16413 16423 16512 16519 16541 16549 16565 16634 16641 16687 16827 16858 16941 16944 1696
2 16972 17035 17086 17103 17110 17189 17192 17222 17253 17371 17398 17410 17421 17437 17451 17505 17546 17549 17578 17673 17713 17773 17807 17825 17841 17861 17864 17870 17913 1
7958 17964 18007 18035 18060 18087 18114 18115 18127 18132 18190 18230 18240 18330 18443 18467 18538 18540 18584 18588 18636 18651 18651 18662 18678 18696 18716 18756 18762 1878
6 18787 18823 18875 18896 18935 18958 19008 19037 19072 19090 19156 19169 19187 19264 19314 19353 19357 19369 19375 19405 19497 19558 19589 19629 19668 19690 19711 19718 19796 1
9801 19815 19855 19866 19895 19912 19923 19954 19976 20024 20037 20040 20053 20055 20072 20142 20159 20175 20215 20222 20259 20315 20326 20328 20355 20416 20424 20426 20450 2047
2 20482 20485 20537 20556 20580 20600 20601 20608 20626 20649 20671 20710 20783 20798 20825 20851 20888 20913 20945 21003 21015 21025 21080 21098 21113 21119 21132 21153 21221 2
1221 21318 21363 21416 21425 21430 21457 21459 21520 21538 21543 21545 21548 21624 21634 21659 21694 21718 21724 21726 21763 21881 21945 21948 21974 22044 22142 22171 22190 2223
3 22279 22296 22355 22386 22409 22413 22428 22466 22483 22532 22549 22549 22593 22619 22646 22648 22658 22663 22701 22704 22725 22741 22758 22798 22813 22848 22850 22867 22888 2
2913 22929 23073 23152 23195 23196 23199 23216 23245 23271 23281 23318 23342 23392 23622 23646 23655 23754 23757 23775 23805 23811 23831 23844 23847 23850 23851 23869 23936 2397
1 23977 23986 23992 24021 24028 24041 24050 24084 24129 24155 24179 24182 24221 24229 24272 24350 24355 24370 24372 24389 24393 24423 24429 24464 24472 24484 24484 24488 24555 2
4596 24626 24648 24766 24767 24774 24778 24855 24937 24946 24948 24970 25072 25087 25184 25200 25205 25264 25269 25313 25347 25411 25423 25484 25508 25542 25547 25547 25561 2562
0 25627 25629 25667 25705 25721 25734 25760 25824 25824 25874 25898 25951 25990 25996 26058 26113 26129 26153 26154 26173 26264 26292 26299 26302 26303 26308 26362 26418 26428 2
6439 26463 26477 26488 26500 26504 26534 26576 26740 26777 26869 26909 26924 26962 26969 27067 27088 27109 27152 27157 27348 27350 27384 27432 27446 27489 27506 27529 27593 2759
5 27611 27624 27644 27753 27756 27813 27870 27892 27938 27982 28009 28019 28022 28027 28070 28082 28112 28145 28253 28286 28289 28296 28297 28318 28321 28323 28433 28464 28476 2
8489 28503 28520 28570 28617 28650 28685 28689 28692 28703 28704 28712 28745 28869 29011 29141 29168 29170 29174 29200 29213 29288 29292 29314 29334 29337 29350 29358 29361 2951
0 29541 29556 29565 29577 29617 29643 29657 29658 29734 29763 29790 29812 29815 29855 29869 29901 29954 29956 29972 30003 30080 30093 30106 30145 30188 30191 30212 30227 30303 3
0333 30523 30527 30674 30695 30771 30814 30833 30836 30838 30877 30900 30932 30974 31001 31003 31060 31101 31107 31111 31115 31185 31196 31240 31286 31316 31322 31329 31361 3142
6 31461 31556 31627 31673 31783 31818 31928 31934 31998 32060 32168 32170 32209 32226 32257 32270 32356 32391 32404 32439 32525 32591 32604 32609 32637 32662 32678 32685 32702 3
2726 32757

time taken 0.001000
PS D:\ADA\ADA_LAB>
```

Graph:

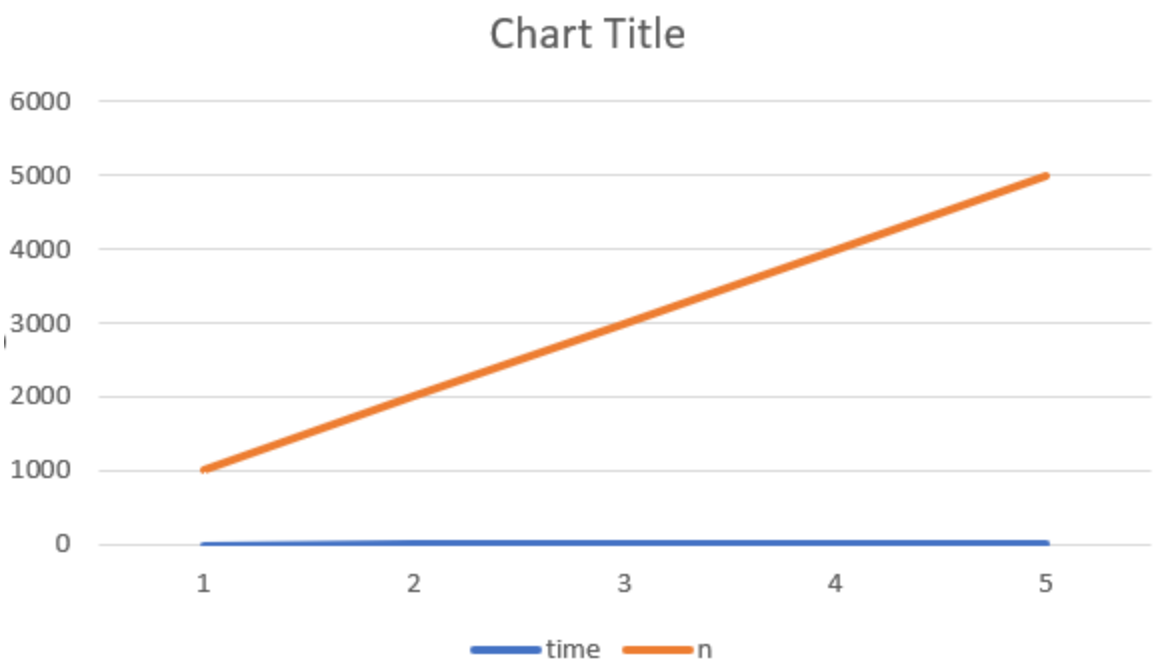


Chart Title
6000
5000
4000
3000
2000
1000
0
1
2
3
4
5
time
n

6)Write program to obtain the Topological ordering of vertices in a given graph

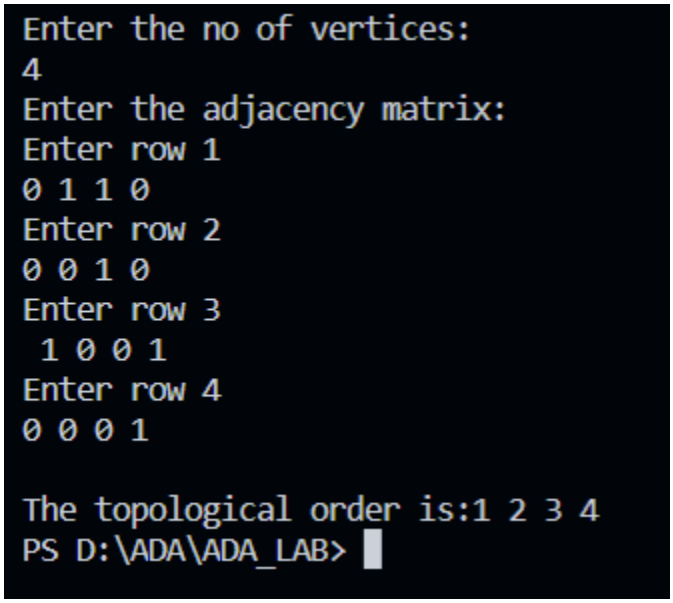
```
#include<stdio.h>
#include<conio.h>
int main()
{

   int i,j,k,n,a[10][10],indeg[10],flag[10],count=0;

   printf("Enter the no of vertices:\n");

   scanf("%d",&n);

   printf("Enter the adjacency matrix:\n");

   for(i=0; i<n; i++)
   {

      printf("Enter row %d\n",i+1);

      for(j=0; j<n; j++)
```

```
        scanf("%d",&a[i][j]);

}
for(i=0; i<n; i++)
{

   indeg[i]=0;

   flag[i]=0;
}

for(i=0; i<n; i++)

   for(j=0; j<n; j++)

        indeg[i]=indeg[i]+a[j][i];

printf("\nThe topological order is:");

while(count<n)
{

   for(k=0; k<n; k++)
```

```
{

   if((indeg[k]==0) && (flag[k]==0))
   {

      printf("%d ",(k+1));

      flag [k]=1;

   }

   for(i=0; i<n; i++)
   {

      if(a[i][k]==1)

         indeg[k]--;

   }
}

count++;
```

```
}
}
```

Output:


Enter the no of vertices:
4
Enter the adjacency matrix:
Enter row 1
0 1 1 0
Enter row 2
0 0 1 0
Enter row 3
1 0 0 1
Enter row 4
0 0 0 1
The topological order is:1 2 3 4
PS D:\ADA\ADA_LAB>

7)Implement Johnson Trotter algorithm to generate permutations.

```
#include <stdio.h>
#include <stdlib.h>
int flag = 0;
int swap(int *a,int *b)
{
    int t = *a;
   *a = *b;
   *b = t;
}

int search(int arr[],int num,int mobile)
{
   int g;
   for(g=0;g<num;g++)
   {
      if(arr[g] == mobile)
      {
          return g+1;
```

```
        }
        else
        {
         flag++;
        }
    }
    return -1;
}
int find_Moblie(int arr[],int d[],int num)
{
    int mobile = 0;
    int mobile_p = 0;
    int i;
    for(i=0;i<num;i++)
    {
        if((d[arr[i]-1] == 0) && i != 0)
        {
            if(arr[i]>arr[i-1] && arr[i]>mobile_p)
            {
                mobile = arr[i];
                mobile_p = mobile;
            }
            else
```

```
      {
         flag++ ;    }
   }
   else if((d[arr[i]-1] == 1) & i != num-1)
   {
      if(arr[i]>arr[i+1] && arr[i]>mobile_p)
      {
         mobile = arr[i];
         mobile_p = mobile;
      }
      else
      {
         flag++;
      }
   }
   else
      {
         flag++;
      }
}
if((mobile_p == 0) && (mobile == 0))
   return 0;
else
```

```
        return mobile;
}
void permutations(int arr[],int d[],int num)
{
    int i;
    int mobile = find_Moblie(arr,d,num);
    int pos = search(arr,num,mobile);
    if(d[arr[pos-1]-1]==0)
        swap(&arr[pos-1],&arr[pos-2]);
    else
        swap(&arr[pos-1],&arr[pos]);
    for(int i=0;i<num;i++)
    {
        if(arr[i] > mobile)
        {
            if(d[arr[i]-1]==0)
                d[arr[i]-1] = 1;
            else
                d[arr[i]-1] = 0;
        }
    }
    for(i=0;i<num;i++)
    {
```

```
        printf(" %d ",arr[i]);
    }
}

int factorial(int k)
{
    int f = 1;
    int i = 0;
    for(i=1;i<k+1;i++)
    {
        f = f*i;
    }
    return f;
}
int main()
{
    int num = 0;
    int i;
    int j;
    int z = 0;
    printf("Johnson trotter algorithm to find all permutations of given
numbers \n");
    printf("Enter the number\n");
    scanf("%d",&num);
```

```
    int arr[num],d[num];
    z = factorial(num);
    printf("The total permutations are %d",z);
    printf("\nAll possible permutations are: \n");
    for(i=0;i<num;i++)
    {
        d[i] = 0;
        arr[i] = i+1;
        printf(" %d ",arr[i]);
    }
    printf("\n");
    for(j=1;j<z;j++)
    {
        permutations(arr,d,num);
        printf("\n");
    }
    return 0;
}
```

Output:

```
Johnson trotter algorithm to find all permutations of given numbers
Enter the number
4
The total permutations are 24
All possible permutations are:
 1  2  3  4
 1  2  4  3
 1  4  2  3
 4  1  2  3
 4  1  3  2
 1  4  3  2
 1  3  4  2
 1  3  2  4
 3  1  2  4
 3  1  4  2
 3  4  1  2
 4  3  1  2
 4  3  2  1
 3  4  2  1
 3  2  4  1
 3  2  1  4
 2  3  1  4
 2  3  4  1
 2  4  3  1
 4  2  3  1
 4  2  1  3
 2  4  1  3
 2  1  4  3
 2  1  3  4
PS D:\ADA\ADA_LAB>
```

8)Sort a given set of N integer elements using Merge Sort technique and compute its time taken. Run the program for different values of N and record the time taken to sort.

```
#include<stdlib.h>
#include<stdio.h>
#include<time.h>
void merge(int arr[], int l, int m, int r)
{
   int i, j, k;
   int n1 = m - l + 1;
   int n2 = r - m;
   int L[n1], R[n2];
   for (i = 0; i < n1; i++)
      L[i] = arr[l + i];
   for (j = 0; j < n2; j++)
      R[j] = arr[m + 1+ j];
   i = 0;
   j = 0;
   k = l;
   while (i < n1 && j < n2)
   {
```

```
    if (L[i] <= R[j])
    {
        arr[k] = L[i];
        i++;
    }
    else
    {
        arr[k] = R[j];
        j++;
    }
    k++;
}
while (i < n1)
{
    arr[k] = L[i];
    i++;
    k++;
}
while (j < n2)
{
    arr[k] = R[j];
    j++;
    k++;
```

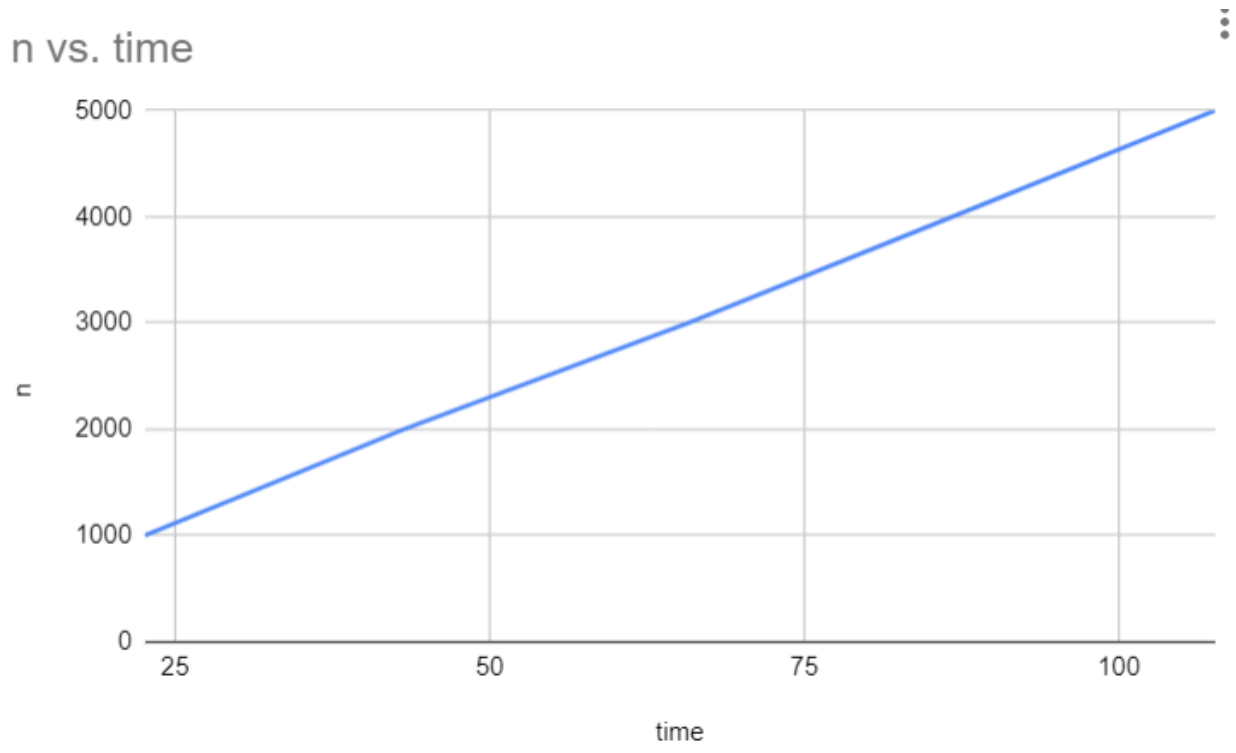
```
    }
}
void mergeSort(int arr[], int l, int r)
{
if (l < r)
{
int m = l+(r-l)/2;
for(int p=0;p<10000000;p++);
mergeSort(arr, l, m);
mergeSort(arr, m+1, r);
merge(arr, l, m, r);
}
}

void printArray(int A[], int size)
{
int i;
for (i=0; i < size; i++)
printf("%d ", A[i]);
printf("\n");
}

int main()
```

```
{
int n;
clock_t st,ed;
   printf("ENTER SIZE OF = ");
   scanf("%d",&n);
   int arr[n];
   printf("ENTER ARRAY ELEMENTS = ");
   for (int j = 0; j < n; j++)
      {
         arr[j] = (rand() % 1000) + 1;
         printf("%4d", arr[j]);
      }
    printf("\n");
    st = clock();
mergeSort(arr, 0, n - 1);
ed = clock();
printf("\n %lf",((double)(ed-st))/CLOCKS_PER_SEC);
printf("\nSORTED ARRAY IS\n");
printArray(arr, n);
return 0;
}
```

Output:

```
ENTER SIZE OF = 5
ENTER ARRAY ELEMENTS =   42 468 335 501 170

 0.083000
SORTED ARRAY IS
42 170 335 468 501
PS D:\ADA\ADA_LAB>
```

Graph:



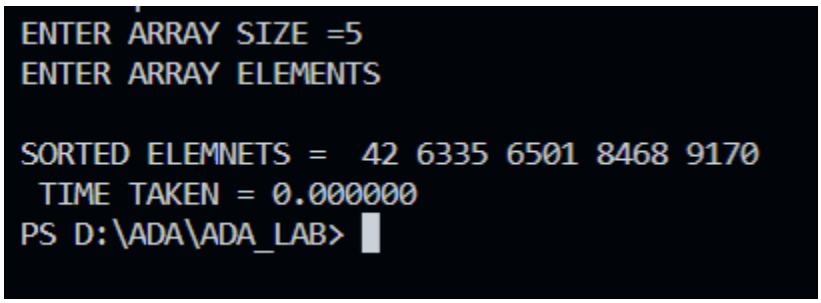
9)Sort a given set of N integer elements using Quick Sort technique and compute its time taken.

```
#include<stdio.h>
void quicksort(int arr[25],int first,int last)
{

   int i, j, pivot, temp;
   if(first<last)
   {
      pivot=first;
      i=first;
      j=last;
      while(i<j)
      {
         while(arr[i]<=arr[pivot]&&i<last)
            i++;
         while(arr[j]>arr[pivot])
            j--;
         if(i<j)
         {
            temp=arr[i];
```

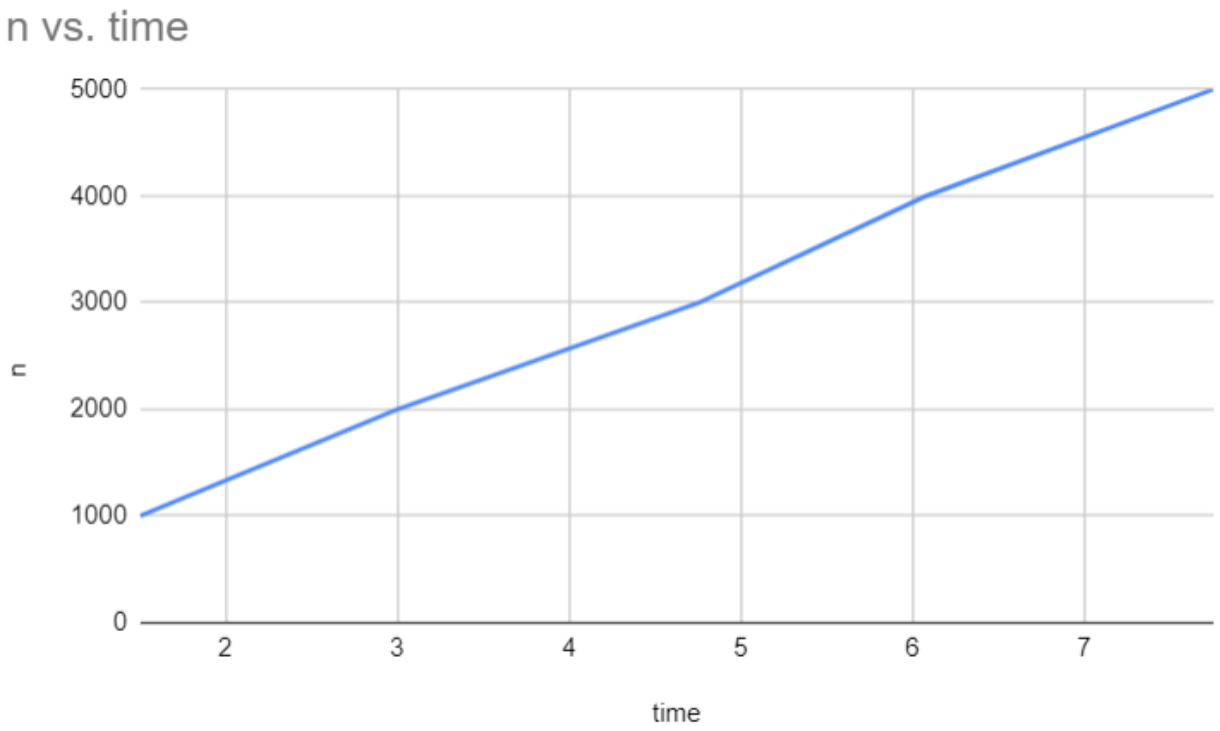
```
            arr[i]=arr[j];
            arr[j]=temp;
          }
        }
        temp=arr[pivot];
        arr[pivot]=arr[j];
        arr[j]=temp;
        for(int p = 0; p<1000000; p++);
        quicksort(arr,first,j-1);
        quicksort(arr,j+1,last);
    }
}
int main()
{
    int i, n;
    time_t st,ed;
    printf("ENTER ARRAY SIZE =");
    scanf("%d",&n);
    int arr[n];
    printf("ENTER ARRAY ELEMENTS");
    for (int j = 0; j < n; j++)
    {
        arr[j] = (rand() % 10000) + 1;
```

```
    }
    printf("\n");
    st = time(NULL);
    quicksort(arr,0,n-1);
    ed = time(NULL);

    printf("\nSORTED ELEMNETS = ");
    for(i=0; i<n; i++)
        printf(" %d",arr[i]);
    printf("\n TIME TAKEN = %f \n",difftime(ed,st));
    return 0;
}
```

Output:



Graph:



10)Sort a given set of N integer elements using Heap Sort technique and compute its time taken.

```
#include <stdio.h>
#include<stdlib.h>
#include<time.h>

void swap(int *a, int *b) {

  int temp = *a;

  *a = *b;

  *b = temp;

}
void heapify(int arr[], int n, int i) {
  int largest = i;

  int left = 2 * i + 1;

  int right = 2 * i + 2;

  if (left < n && arr[left] > arr[largest])
```

```
    largest = left;

  if (right < n && arr[right] > arr[largest])

    largest = right;

  if (largest != i) {

    swap(&arr[i], &arr[largest]);

    heapify(arr, n, largest);

  }

}

void heapSort(int arr[], int n) {

  for (int i = n / 2 - 1; i >= 0; i--)

    heapify(arr, n, i);
```

```
  for (int i = n - 1; i >= 0; i--) {

    swap(&arr[0], &arr[i]);

    heapify(arr, i, 0);

  }



}

void printArray(int arr[], int n)
{
  for (int i = 0; i < n; i++)
    printf("%d ", arr[i]);
  printf("\n");

}



int main()
{
  clock_t start,end;
```

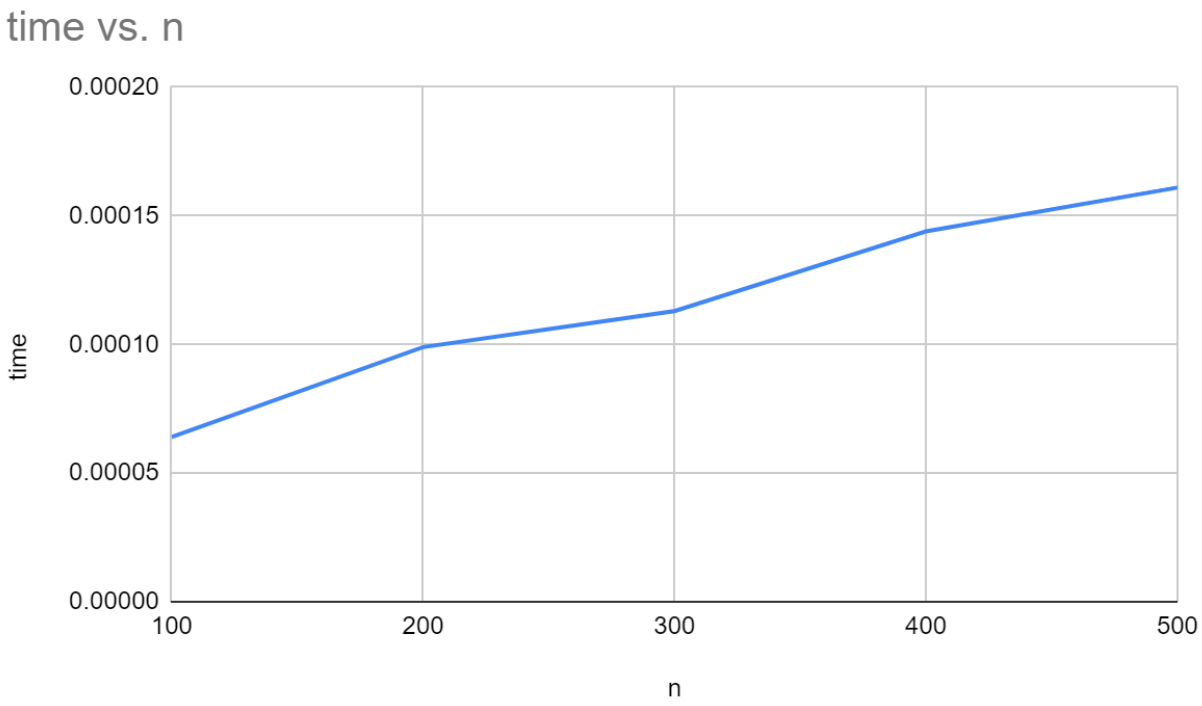
```
  int n;
  printf("Enter the number of elements of the array\n");
  scanf("%d",&n);

  int arr[n];
//   printf("Enter the elements of the array\n");
//   for(int i=0;i<n;i++){
//       scanf("%d",&arr[i]);
//   }

//   for random input
for(int i=0;i<n;i++){
    arr[i]=rand();
}


  start=clock();
  for(int i=0;i<9999;i++);
  heapSort(arr,n);
  end=clock();

//   printf("Sorted array is: ");
//   for(int i=0;i<n;i++){
```

```
//      printf("%d ",arr[i]);
//   }
 printf("\nTime taken is: %f \n",difftime(end,start)/CLOCKS_PER_SEC);
}
```

output:

```
Enter the number of elements of the array
100

Time taken is: 0.001000
PS D:\ADA\ADA_LAB>
```

Graph:



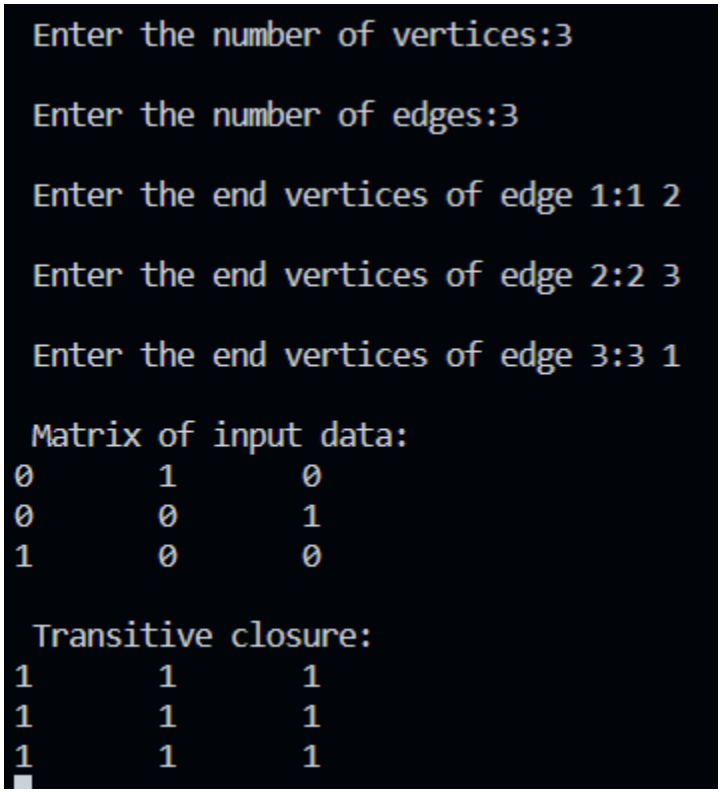
11)Implement Warshall’s algorithm using dynamic programming

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
int max(int,int);
void warshal(int p[10][10],int n) {
   int i,j,k;
   for (k=1;k<=n;k++)
     for (i=1;i<=n;i++)
      for (j=1;j<=n;j++)
       p[i][j]=max(p[i][j],p[i][k]&&p[k][j]);
}
int max(int a,int b) {
   ;
   if(a>b)
    return(a); else
    return(b);
}
void main() {
   int p[10][10]= {
      0
   }   ,n,e,u,v,i,j;
```

```
printf("\n Enter the number of vertices:");
scanf("%d",&n);
printf("\n Enter the number of edges:");
scanf("%d",&e);
for (i=1;i<=e;i++) {
   printf("\n Enter the end vertices of edge %d:",i);
   scanf("%d%d",&u,&v);
   p[u][v]=1;
}
printf("\n Matrix of input data: \n");
for (i=1;i<=n;i++) {
   for (j=1;j<=n;j++)
      printf("%d\t",p[i][j]);
   printf("\n");
}
warshal(p,n);
printf("\n Transitive closure: \n");
for (i=1;i<=n;i++) {
   for (j=1;j<=n;j++)
      printf("%d\t",p[i][j]);
   printf("\n");
}
```

```
    getch();
}
```

Output:



12)Implement 0/1 Knapsack problem using dynamic programming.

```
#include<stdio.h>
#include<conio.h>

int max(int a, int b)
 {
   if(a>b)
     return a;
   else return b;
 }

void knapsack(int w[],int v[], int s,int n)
{
  int k[n+1][s+1];
  int i,j,res=0;

  for(i=0;i<=n;i++)
     for(j=0;j<=s;j++)
        { if(i==0  || j==0)
            k[i][j]=0;
          else if(w[i - 1] <= j)
            k[i][j] = max(v[i-1]+k[i-1][j-w[i-1]],k[i-1][j]);
```

```
        else
          k[i][j] = k[i-1][j];
       }

  res=k[n][s];
  printf("\n\nMaximum Value that can be obtained is : %d",res);
  j=s;
  printf("\nAnd the objects with there respective Weights selected are :");
  for(i=n;i>0 && res>0; i--)
    {if (res == k[i - 1][j])
        continue;
      else
        {printf("%d ", w[i-1]);
         res =res-v[i-1];
         j = j-w[i-1];
        }
    }
}

int main()
{
  int w[10],v[10],s,n,i;
  printf("\nEnter the Number of objects : ");
```

```
  scanf("%d",&n);
  printf("\nEnter the Weights of the objects : ");
  for(i=0;i<n;i++)
    scanf("%d",&w[i]);
  printf("\nEnter the Values of the objects : ");
  for(i=0;i<n;i++)
    scanf("%d",&v[i]);
  printf("\nEnter the Size of the KnapSack : ");
  scanf("%d",&s);
  knapsack(w,v,s,n);
}
```

Output:

```
Enter the Number of objects : 3

Enter the Weights of the objects : 1 2 3

Enter the Values of the objects : 3 2 1

Enter the Size of the KnapSack : 5


Maximum Value that can be obtained is : 5
And the objects with there respective Weights selected are :2 1
PS D:\ADA\ADA_LAB>
```

13)Implement All Pair Shortest paths problem using Floyd's algorithm.

```
#include<stdio.h>
#include<conio.h>
int min(int,int);
void floyds(int p[10][10],int n) {
   int i,j,k;
   for (k=1;k<=n;k++)
     for (i=1;i<=n;i++)
      for (j=1;j<=n;j++)
       if(i==j)
        p[i][j]=0; else
        p[i][j]=min(p[i][j],p[i][k]+p[k][j]);
}
int min(int a,int b) {
   if(a<b)
     return(a); else
     return(b);
}
void main() {
   int p[10][10],w,n,e,u,v,i,j;

   printf("\n Enter the number of vertices:");
```

```
scanf("%d",&n);
printf("\n Enter the number of edges:\n");
scanf("%d",&e);
for (i=1;i<=n;i++) {
   for (j=1;j<=n;j++)
      p[i][j]=999;
}
for (i=1;i<=e;i++) {
   printf("\n Enter the end vertices of edge%d with its weight \n",i);
   scanf("%d%d%d",&u,&v,&w);
   p[u][v]=w;
}
printf("\n Matrix of input data:\n");
for (i=1;i<=n;i++) {
   for (j=1;j<=n;j++)
      printf("%d \t",p[i][j]);
   printf("\n");
}
floyds(p,n);
printf("\n Transitive closure:\n");
for (i=1;i<=n;i++) {
   for (j=1;j<=n;j++)
      printf("%d \t",p[i][j]);
```

```
        printf("\n");
    }
    printf("\n The shortest paths are:\n");
    for (i=1;i<=n;i++)
      for (j=1;j<=n;j++) {
        if(i!=j)
            printf("\n <%d,%d>=%d",i,j,p[i][j]);
    }
    getch();
}
```

Output:

```
 Enter the number of vertices:4

 Enter the number of edges:
4

 Enter the end vertices of edge1 with its weight
1 3 4

 Enter the end vertices of edge2 with its weight
1 2 5

 Enter the end vertices of edge3 with its weight
2 4 6

 Transitive closure:
0       5       4       11
999     0       999     6
999     999     0       999
999     7       999     0

 The shortest paths are:

 <1,2>=5
 <1,3>=4
 <1,4>=11
 <2,1>=999
 <2,3>=999
 <2,4>=6
 <3,1>=999
 <3,2>=999
 <3,4>=999
 <4,1>=999
 <4,2>=7
 <4,3>=999
```

14) Find Minimum Cost Spanning Tree of a given undirected graph using Prim's algorithm.

```
#include<stdio.h>
#include<stdlib.h>

#define infinity 9999
#define MAX 20

int G[MAX][MAX],spanning[MAX][MAX],n;

int prims();

int main()
{
   int i,j,total_cost;
   printf("Enter no. of vertices:");
   scanf("%d",&n);
   printf("\nEnter the cost of  adjacency matrix:\n");
   for(i=0;i<n;i++)
      for(j=0;j<n;j++)
         scanf("%d",&G[i][j]);
   total_cost=prims();
   printf("\nspanning tree matrix:\n");
   for(i=0;i<n;i++)
```

```
    {
    printf("\n");
    for(j=0;j<n;j++)
    printf("%d\t",spanning[i][j]);
    }
    printf("\n\nTotal cost of spanning tree=%d",total_cost);
    return 0;
}

int prims()
{
    int cost[MAX][MAX];
    int u,v,min_distance,distance[MAX],from[MAX];
    int visited[MAX],no_of_edges,i,min_cost,j;
    //create cost[][] matrix,spanning[][]
    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
        {
            if(G[i][j]==0)
            cost[i][j]=infinity;
            else
            cost[i][j]=G[i][j];
            spanning[i][j]=0;
```

```
   }
//initialise visited[],distance[] and from[]
distance[0]=0;
visited[0]=1;
for(i=1;i<n;i++)
{
   distance[i]=cost[0][i];
   from[i]=0;
   visited[i]=0;
}
min_cost=0; //cost of spanning tree
no_of_edges=n-1; //no. of edges to be added
while(no_of_edges>0)
{

   //find the vertex at minimum distance from the tree
   min_distance=infinity;
   for(i=1;i<n;i++)
   if(visited[i]==0&&distance[i]<min_distance)
   {
      v=i;
      min_distance=distance[i];
   }
```

```
        u=from[v];
        //insert the edge in spanning tree
        spanning[u][v]=distance[v];
        spanning[v][u]=distance[v];
        no_of_edges--;
        visited[v]=1;
        //updated the distance[] array
        for(i=1;i<n;i++)
           if(visited[i]==0&&cost[i][v]<distance[i])
           {
              distance[i]=cost[i][v];
              from[i]=v;
           }
        min_cost=min_cost+cost[u][v];
    }
    return(min_cost);
}
```

Output:

```
Enter no. of vertices:6

Enter the cost of  adjacency matrix:
0 3 1 6 0 0
3 0 5 0 3 0
1 5 0 5 6 4
6 0 5 0 0 2
0 3 6 0 0 6
0 0 4 2 6 0

spanning tree matrix:

0       3       1       0       0       0
3       0       0       0       3       0
1       0       0       0       0       4
0       0       0       0       0       2
0       3       0       0       0       0
0       0       4       2       0       0

Total cost of spanning tree=13
PS D:\ADA\ADA_LAB>
```

15)Find Minimum Cost Spanning Tree of a given undirected graph using Kruskals algorithm

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
int i,j,k,a,b,u,v,n,ne=1;
int min,mincost=0,cost[9][9],parent[9];
int find(int);
int uni(int,int);
void main()
{
   printf("\nEnter the no. of vertices:");
   scanf("%d",&n);
   printf("\nEnter the cost adjacency matrix:\n");
   for(i=1;i<=n;i++)
   {
      for(j=1;j<=n;j++)
      {
         scanf("%d",&cost[i][j]);
         if(cost[i][j]==0)
            cost[i][j]=999;
      }
   }
   printf("The edges of Minimum Cost Spanning Tree are\n");
```

```
while(ne < n)
{
   for(i=1,min=999;i<=n;i++)
   {
      for(j=1;j <= n;j++)
      {
         if(cost[i][j] < min)
         {
            min=cost[i][j];
            a=u=i;
            b=v=j;
         }
      }
   }
   u=find(u);
   v=find(v);
   if(uni(u,v))
   {
      printf("%d edge (%d,%d) =%d\n",ne++,a,b,min);
      mincost +=min;
   }
   cost[a][b]=cost[b][a]=999;
}
```

```
    printf("\n\tMinimum cost = %d\n",mincost);
}
int find(int i)
{
    while(parent[i])
        i=parent[i];
    return i;
}
int uni(int i,int j)
{
    if(i!=j)
    {
    parent[j]=i;
    return 1;
    }
    return 0;
}
```

Output:

```
Enter the no. of vertices:6

Enter the cost adjacency matrix:
0 3 1 6 0 0
3 0 5 0 3 0
1 5 0 5 6 4
6 0 5 0 0 2
0 3 6 0 0 6
0 0 4 2 6 0
The edges of Minimum Cost Spanning Tree are
1 edge (1,3) =1
2 edge (4,6) =2
3 edge (1,2) =3
4 edge (2,5) =3
5 edge (3,6) =4

        Minimum cost = 13
PS D:\ADA\ADA_LAB>
```

16)From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.

```
#include<stdio.h>
#include<conio.h>
#define INFINITY 9999
#define MAX 10

void dijkstra(int G[MAX][MAX],int n,int startnode);

int main()
{
int G[MAX][MAX],i,j,n,u;
printf("Enter no. of vertices:");
scanf("%d",&n);
printf("\nEnter the adjacency matrix:\n");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
scanf("%d",&G[i][j]);
printf("\nEnter the starting node:");
scanf("%d",&u);
dijkstra(G,n,u);
return 0;
```

```
}

void dijkstra(int G[MAX][MAX],int n,int startnode)
{

int cost[MAX][MAX],distance[MAX],pred[MAX];
int visited[MAX],count,mindistance,nextnode,i,j;
//pred[] stores the predecessor of each node
//count gives the number of nodes seen so far
//create the cost matrix
for(i=0;i<n;i++)
for(j=0;j<n;j++)
if(G[i][j]==0)
cost[i][j]=INFINITY;
else
cost[i][j]=G[i][j];
//initialize pred[],distance[] and visited[]
for(i=0;i<n;i++)
{
distance[i]=cost[startnode][i];
pred[i]=startnode;
visited[i]=0;
```

```
}
distance[startnode]=0;
visited[startnode]=1;
count=1;
while(count<n-1)
{
mindistance=INFINITY;
//nextnode gives the node at minimum distance
for(i=0;i<n;i++)
if(distance[i]<mindistance&&!visited[i])
{
mindistance=distance[i];
nextnode=i;
}
//check if a better path exists through nextnode
visited[nextnode]=1;
for(i=0;i<n;i++)
if(!visited[i])
if(mindistance+cost[nextnode][i]<distance[i])
{
distance[i]=mindistance+cost[nextnode][i];
pred[i]=nextnode;
```

```
}
count++;
}

//print the path and distance of each node
for(i=0;i<n;i++)
if(i!=startnode)
{
printf("\nDistance of node%d=%d",i,distance[i]);
printf("\nPath=%d",i);
j=i;
do
{
j=pred[j];
printf("<-%d",j);
}while(j!=startnode);
}
}
```

Output:

```
Enter no. of vertices:5

Enter the adjacency matrix:
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0

Enter the starting node:0

Distance of node1=10
Path=1<-0
Distance of node2=50
Path=2<-3<-0
Distance of node3=30
Path=3<-0
Distance of node4=60
Path=4<-2<-3<-0
PS D:\ADA\ADA_LAB>
```

17)Implement “Sum of Subsets” using Backtracking. “Sum of Subsets” problem: 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.

```
#include<stdio.h>
int s[10] , x[10],d ;
void sumofsub ( int , int , int ) ;
void main ()
{
int n , sum = 0 ;
int i ;
printf ( " \n Enter the size of the set : " ) ;
scanf ( "%d" , &n ) ;
printf ( " \n Enter the set in increasing order:\n" ) ;
for ( i = 1 ; i <= n ; i++ )
scanf ("%d", &s[i] ) ;
printf ( " \n Enter the value of d : \n " ) ;
scanf ( "%d" , &d ) ;
for ( i = 1 ; i <= n ; i++ )
sum = sum + s[i] ;
if ( sum < d || s[1] > d )
printf ( " \n No subset possible : " ) ;
else
```

```
sumofsub ( 0 , 1 , sum ) ;
}
void sumofsub ( int m , int k , int r )
{
int i=1 ;
x[k] = 1 ;
if ( ( m + s[k] ) == d )
{
printf("Subset:");
for ( i = 1 ; i <= k ; i++ )
if ( x[i] == 1 )
printf ( "\t%d" , s[i] ) ;
printf ( "\n" ) ;
}
else
if ( m + s[k] + s[k+1] <= d )
sumofsub ( m + s[k] , k + 1 , r - s[k] ) ;
if ( ( m + r - s[k] >= d ) && ( m + s[k+1] <=d ) )
{
x[k] = 0;
sumofsub ( m , k + 1 , r - s[k] ) ;
}
}
```

Output:

```
 Enter the size of the set : 5

 Enter the set in increasing order:
1
2
3
4
5

 Enter the value of d :
 3
Subset: 1       2
Subset: 3
```

18)Implement “N-Queens Problem” using Backtracking.

```
#include<stdio.h>
#include<math.h>

int board[20],count;

int main()
{
   int n,i,j;
   void queen(int row,int n);

   printf(" - N Queens Problem Using Backtracking -");
   printf("\n\nEnter number of Queens:");
   scanf("%d",&n);
   queen(1,n);
   return 0;
}


void print(int n)
{
   int i,j;
   printf("\n\nSolution %d:\n\n",++count);
```

```
    for(i=1;i<=n;++i)
        printf("\t%d",i);

    for(i=1;i<=n;++i)
    {
        printf("\n\n%d",i);
        for(j=1;j<=n;++j)
        {
            if(board[i]==j)
                printf("\tQ");
            else
                printf("\t-");
        }
    }
}


int place(int row,int column)
{
    int i;
    for(i=1;i<=row-1;++i)
    {
    if(board[i]==column)
    return 0;
    else
    if(abs(board[i]-column)==abs(i-row))
```

```
        return 0;
    }

    return 1;
}



void queen(int row,int n)
{
    int column;
    for(column=1;column<=n;++column)
    {
    if(place(row,column))
    {
    board[row]=column;
    if(row==n)
        print(n);
        else
        queen(row+1,n);
    }
    }
}
```

Output:

```
Enter number of Queens:4

Solution 1:

        1       2       3       4

1       -       Q       -       -

2       -       -       -       Q

3       Q       -       -       -

4       -       -       Q       -

Solution 2:

        1       2       3       4

1       -       -       Q       -

2       Q       -       -       -

3       -       -       -       Q

4       -       Q       -       -
```