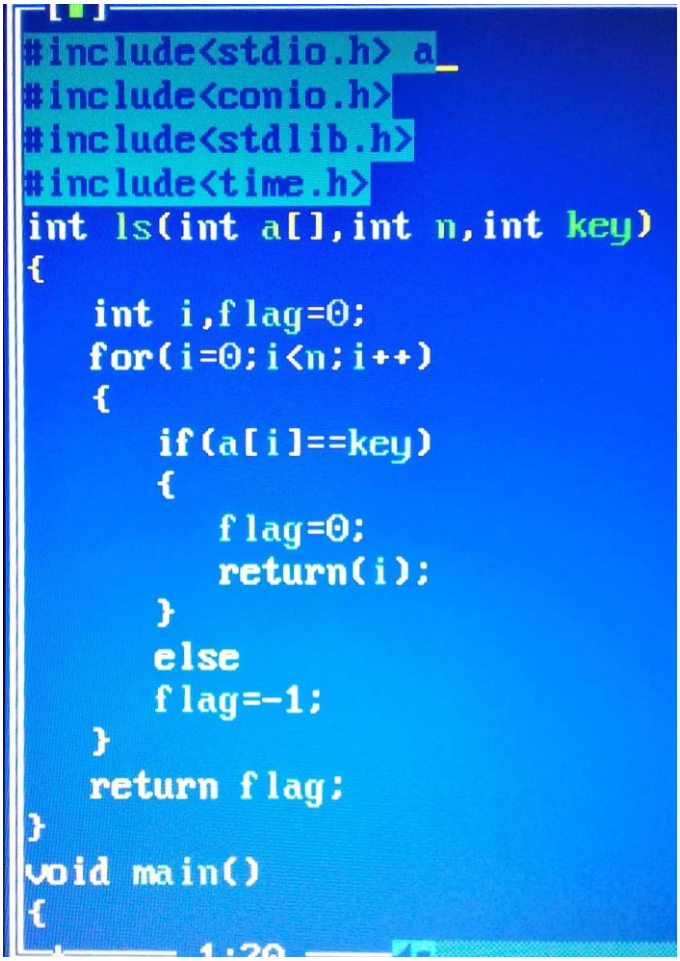
1. Write a program to search an elememt in a array using linear search method. Find the time complexity.



```
#include<stdio.h> a
#include<conio.h>
#include<stdlib.h>
#include<time.h>
int ls(int a[],int n,int key)
{
    int i,flag=0;
    for(i=0;i<n;i++)
    {
        if(a[i]==key)
        {
            flag=0;
            return(i);
        }
        else
        flag=-1;
    }
    return flag;
}
void main()
{
```

```
void main()
{
    int *a,i,n,key,flag;
    clock_t start,end;
    clrscr();
    printf("Enter the size\n");
    scanf("%d",&n);
    a=(int*)calloc(n,sizeof(int));
    printf("Elements are\n");
    for(i=0;i<n;i++)
    {
        a[i]=rand();
        printf("%d\t",a[i]);
    }
    printf("\nEnter the search element\n");
    scanf("%d",&key);
    start=clock();
    delay(110);
    flag=ls(a,n,key);
    end=clock();
    if(flag==-1)
```

```
scanf("%d",&n);
a=(int*)calloc(n,sizeof(int));
printf("Elements are\n");
for(i=0;i<n;i++)
{
   a[i]=rand();
   printf("%d\t",a[i]);
}
printf("\nEnter the search element\n");
scanf("%d",&key);
start=clock();
delay(110);
flag=ls(a,n,key);
end=clock();
if(flag==-1)
printf("%d not found\n",key);
else
printf("%d is at location %d\n",key,flag+1);
printf("Time taken = %f\n",(end-start)/CLK_TCK);
getch();
}
```

OUTPUT :-

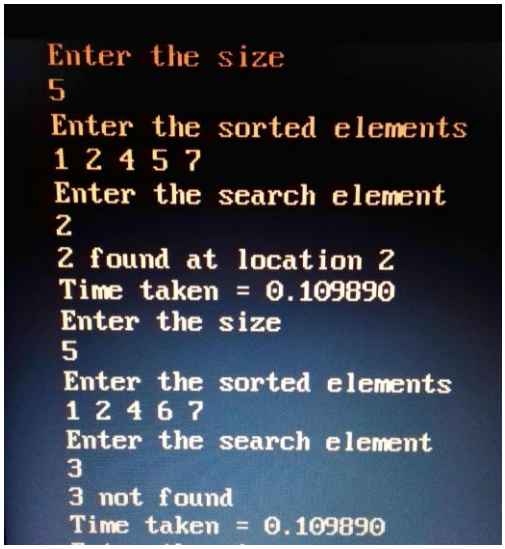
```
Enter the size
5
Elements are
346     130     10982   1090    11656
Enter the search element
130
130 is at location 2
Time taken = 0.109890
Enter the size
5
Elements are
346     130     10982   1090    11656
Enter the search element
1
1 not found
Time taken = 0.109890
```

2. Write a program to search an elememt in a array using binary search method. Find the time complexity.

```
#include<stdaio.h>
#include<conio.h>
#include<time.h>
int bs(int a[],int key,int first,int last)
{
   int mid;
   while(first<=last)
   {
      mid=(first+last)/2;
      if(key==a[mid])
      return mid+1;
      else if(key<a[mid])
      last=mid-1;
      else
      first=mid+1;
   }
   return -1;
}
void main()
{
   int a[100],n,i,j,key,first,last,res;
   clock_t start,end;
   clrscr();
   printf("Enter the size\n");
   scanf("%d",&n);
   printf("Enter the sorted elements\n");
   for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }
   printf("Enter the search element\n");
   scanf("%d",&key);
   first=0;
   last=n-1;
   start=clock();
   delay(110);
   res=bs(a,key,first,last);
   end=clock();
   if(res==-1)
```

```
      scanf("%d",&a[i]);
   }
   printf("Enter the search element\n");
   scanf("%d",&key);
   first=0;
   last=n-1;
   start=clock();
   delay(110);
   res=bs(a,key,first,last);
   end=clock();
   if(res==-1)
   {
      printf("%d not found\n",key);
   }
   else
   {
      printf("%d found at location %d\n",key,res);
   }
   printf("Time taken = %f\n",(end-start)/CLK_TCK);
   getch();
}
```

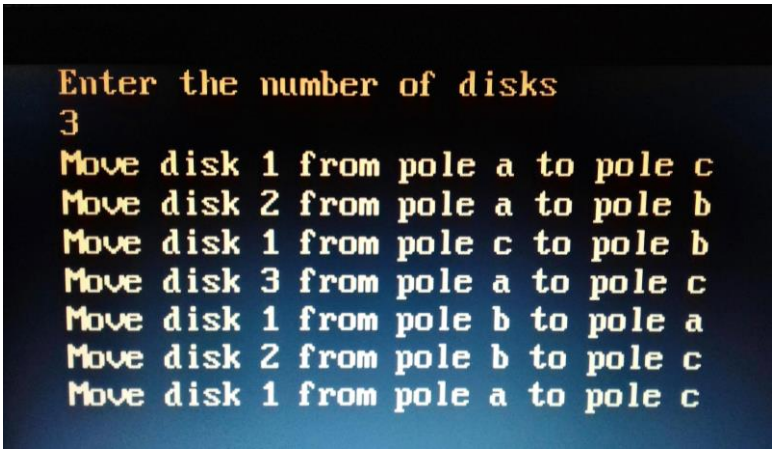
OUTPUT :-



3. Write a program for tower of hanoi using recursion and find the time complexity.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void toh(int n,char a,char c,char b)
{
   if(n==1)
   {
      printf("Move disk 1 from pole %c to pole %c\n",a,c);
      return;
   }
   toh(n-1,a,b,c);
   printf("Move disk %d from pole %c to pole %c\n",n,a,c);
   toh(n-1,b,c,a);
}
void main()
{
   int n;
   clrscr();
   printf("Enter the number of disks\n");
   scanf("%d",&n);
   toh(n,'a','c','b');
   getch();
}
```

OUTPUT :-



4. Write a program to sort elements using selection sort.

```c
#include<stdio.h>
#include<conio.h>
#include<time.h>
void main()
{
    int *a,n,i,j,temp,min;
    clock_t start,end;
    clrscr();
    printf("Enter the size\n");
    scanf("%d",&n);
    a=(int*)calloc(n,sizeof(int));
    for(i=0;i<n;i++)
    {
        a[i]=rand();
        printf("%d\t",a[i]);
    }
    start=clock();
    delay(110);
    for(i=0;i<n;i++)
    {
        min=i;
```

```c
    for(j=i+1;j<n;j++)
    {
        if(a[min]>a[j])
        min=j;
        if(min!=i)
        {
            temp=a[i];
            a[i]=a[min];
            a[min]=temp;
        }
    }
}
end=clock();
printf("\nSorted array :- \n");
for(i=0;i<n;i++)
{
    printf("%d\t",a[i]);
}
printf("\nTime taken = %f",(end-start)/CLK_TCK);
getch();
}
```

OUTPUT :-

```
Enter the size
10
346     130     10982   1090    11656   7117    17595   6415    22948   31126

Sorted array :-
130     346     1090    6415    7117    10982   11656   17595   22948   31126

Time taken = 0.109890
```

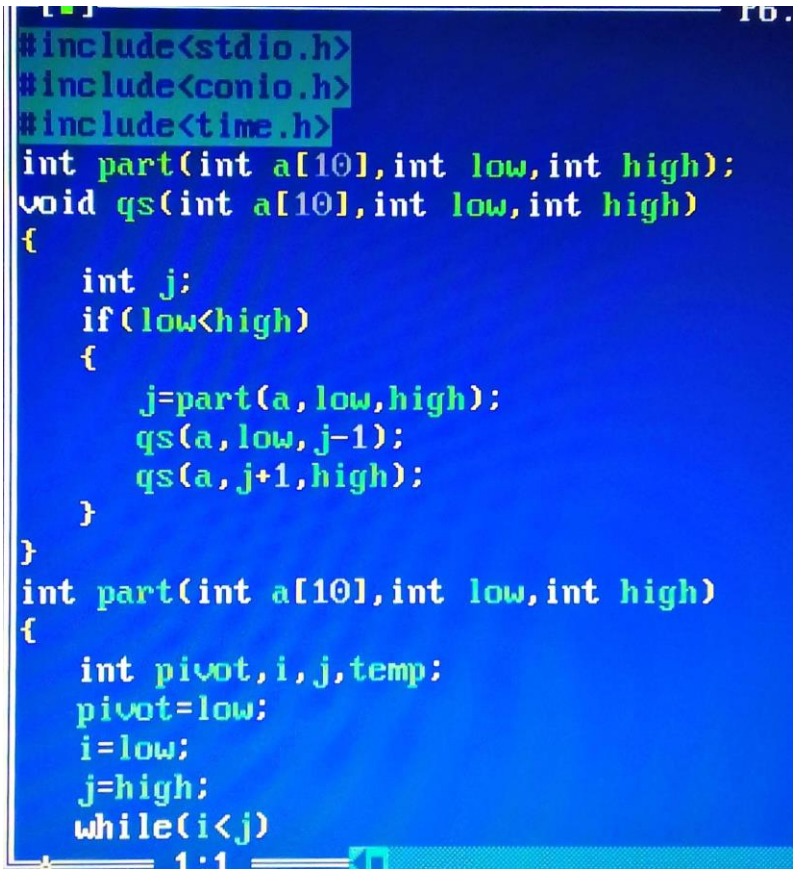
5. Write a program to find value of A using brute force based algorithm and divide and conquer based algorithm.

```c
#include<stdio.h>
#include<conio.h>
void main()
{
    int base,a,res;
    clrscr();
    printf("Enter the base\n");
    scanf("%d",&base);
    printf("Enter a positive number\n");
    scanf("%d",&a);
    res=pow(base,a);
    printf("%d pow %d = %d",base,a,res);
    getch();
}
int pow(int base,int a)
{
    if(a!=0)
    {
        return(base*pow(base,a-1));
    }
    else
    return 1;
}
```

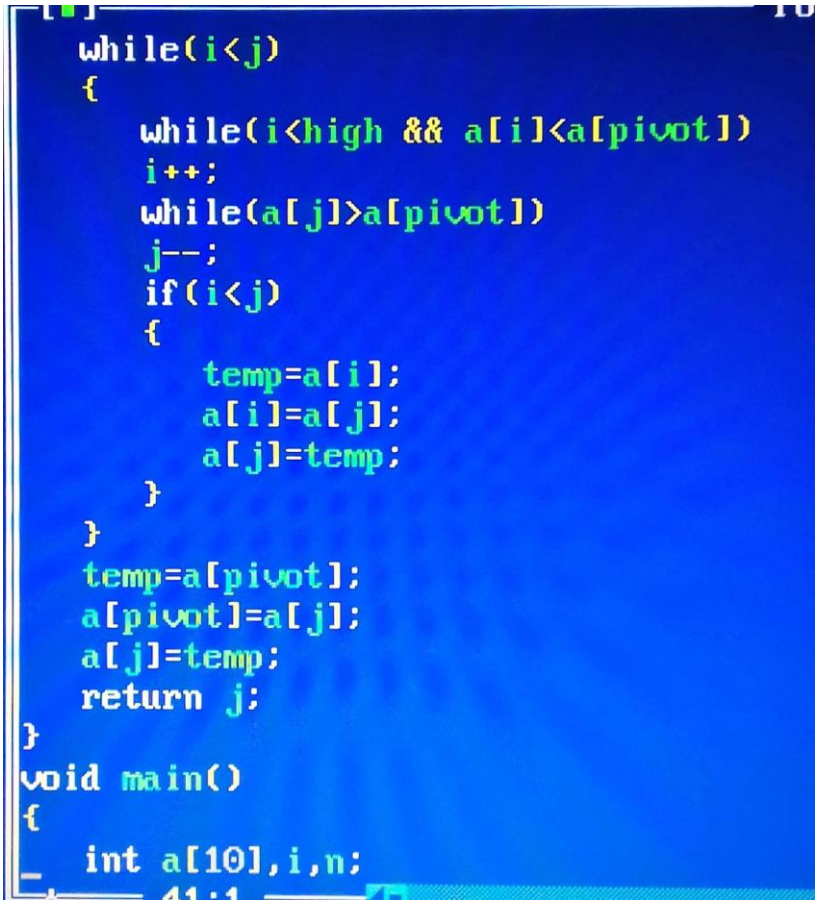
OUTPUT :-

```
Enter the base
2
Enter a positive number
3
2 pow 3 = 8_
```

6. Write a program to sort an elements in an array using quick sort. Find the time required to sort the elements.



```
#include<stdio.h>
#include<conio.h>
#include<time.h>
int part(int a[10],int low,int high);
void qs(int a[10],int low,int high)
{
   int j;
   if(low<high)
   {
      j=part(a,low,high);
      qs(a,low,j-1);
      qs(a,j+1,high);
   }
}
int part(int a[10],int low,int high)
{
   int pivot,i,j,temp;
   pivot=low;
   i=low;
   j=high;
   while(i<j)
```



```
   while(i<j)
   {
      while(i<high && a[i]<a[pivot])
      i++;
      while(a[j]>a[pivot])
      j--;
      if(i<j)
      {
         temp=a[i];
         a[i]=a[j];
         a[j]=temp;
      }
   }
   temp=a[pivot];
   a[pivot]=a[j];
   a[j]=temp;
   return j;
}
void main()
{
   int a[10],i,n;
```

```
clock_t start,end;
clrscr();
printf("Enter the size\n");
scanf("%d",&n);
printf("Enter the elements\n");
for(i=0;i<n;i++)
{
   scanf("%d",&a[i]);
}
start=clock();
delay(110);
qs(a,0,n-1);
end=clock();
printf("Sorted array :- \n");
for(i=0;i<n;i++)
{
   printf("%d\t",a[i]);
}
printf("\nTime taken = %f",(end-start)/CLK_TCK);
getch();
}
```

OUTPUT :-

```
Enter the size
5
Enter the elements
1 8 0 5 9
Sorted array :-
0       1       5       8       9
Time taken = 0.109890_
```

7. Write a program to find minimum cost spanning tree of a graph using prims algorithm.

```
#include<stdio.h>
#include<conio.h>
int a,b,u,v,n,i,j,noe=1;
int visited[10],min,mincost=0,cost[10][10];
void main()
{
   clrscr();
   printf("Enter the number of vertices\n");
   scanf("%d",&n);
   printf("Enter the 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;
      }
   }
   for(i=2;i<=n;i++)
   {
```

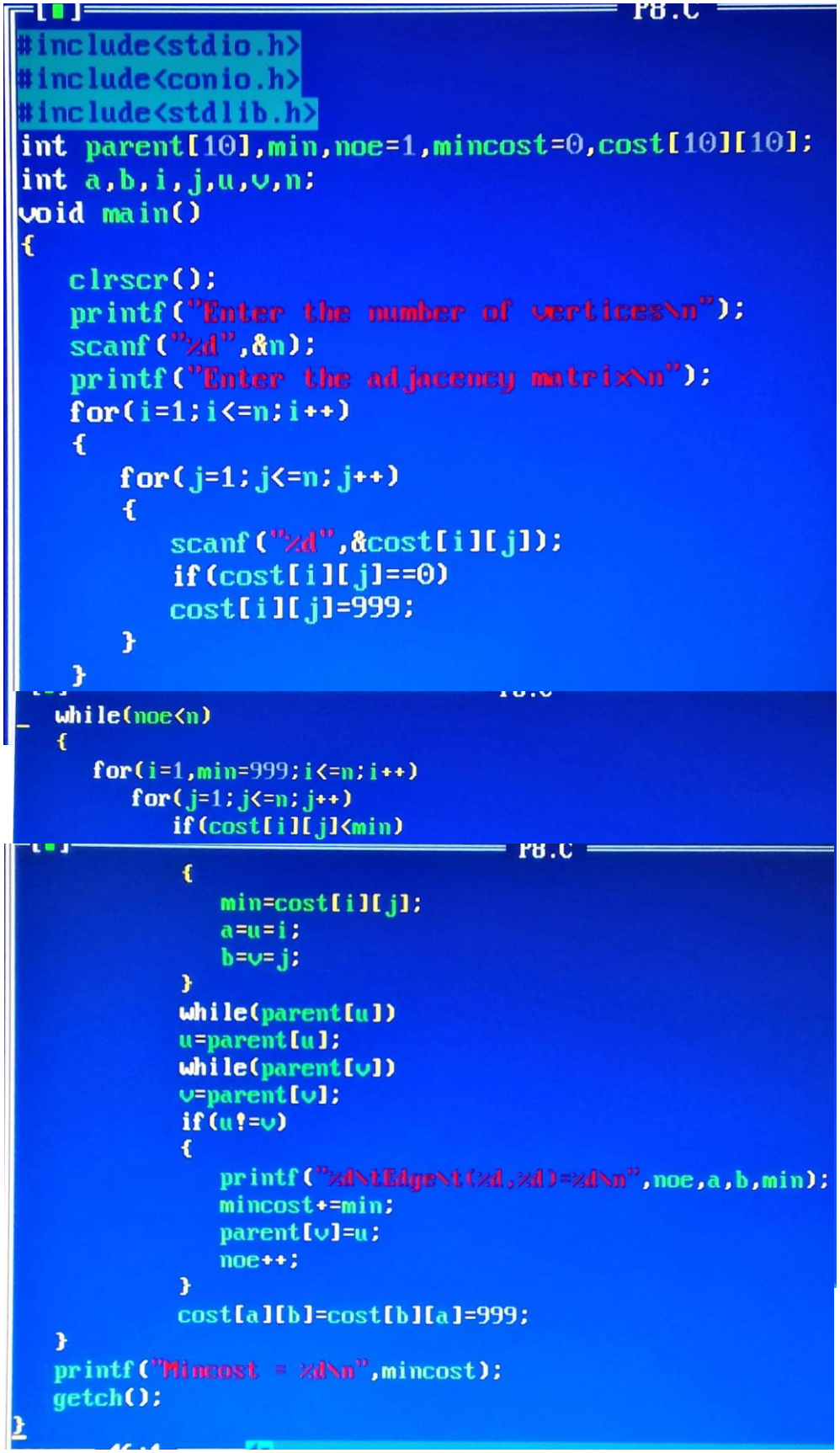
```
for(i=2;i<=n;i++)
{
   visited[i]=0;
}
printf("Edges of spanning tree\n");
visited[1]=1;
while(noe<n)
{
   for(i=1,min=999;i<=n;i++)
   {
      for(j=1;j<=n;j++)
      {
         if(cost[i][j]<min)
         if(visited[i]==0)
         continue;
         else
         {
            min=cost[i][j];
            a=u=i;
            b=v=j;
         }
```

```
         continue;
         else
         {
            min=cost[i][j];
            a=u=i;
            b=v=j;
         }
      }
   }
   if(visited[u]==0 || visited[v]==0)
   {
      printf("%d\tEdge\t(%d,%d)=%d\n",noe,a,b,min);
      mincost=mincost+min;
      visited[b]=1;
      noe=noe+1;
   }
   cost[a][b]=cost[b][a]=999;
}
printf("Mincost = %d\n",mincost);
getch();
}
```

OUTPUT :-

```
Enter the number of vertices
5
Enter the adjacency matrix
999 11 9 7 8
11 999 15 14 13
9 15 999 12 14
7 14 12 999 6
8 13 14 6 999
Edges of spanning tree
1       Edge    (1,4)=7
2       Edge    (4,5)=6
3       Edge    (1,3)=9
4       Edge    (1,2)=11
Mincost = 33
```

8. Write a program to find the minimum cost spanning tree of the undirected graph using kruskal algorithm.



```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
int parent[10],min,noe=1,mincost=0,cost[10][10];
int a,b,i,j,u,v,n;
void main()
{
   clrscr();
   printf("Enter the number of vertices\n");
   scanf("%d",&n);
   printf("Enter the 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;
      }
   }
  while(noe<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;
           }
           while(parent[u])
           u=parent[u];
           while(parent[v])
           v=parent[v];
           if(u!=v)
           {
              printf("%d\tEdge\t(%d,%d)=%d\n",noe,a,b,min);
              mincost+=min;
              parent[v]=u;
              noe++;
           }
           cost[a][b]=cost[b][a]=999;
  }
  printf("Mincost = %d\n",mincost);
  getch();
}
```

OUTPUT :-

```
Enter the number of vertices
6
Enter the adjacency matrix
0 5 10 999 999 999
5 0 999 45 55 20
10 999 0 999 999 25
999 45 999 0 35 50
999 55 999 35 0 45
999 20 25 50 45 0
1        Edge    (1,2)=5
2        Edge    (1,3)=10
3        Edge    (2,6)=20
4        Edge    (4,5)=35
5        Edge    (2,4)=45
Mincost = 115
```

9. Write a program to implement 0/1 knapsack problem using dynamic programming.

```
#include<stdio.h>
#include<conio.h>
int i,j,n,m,w[50],p[50],maxprofit;
int max(int x,int y)
{
   if(x>y)
   return x;
   else
   return y;
}
knapsack(int i,int c)
{
   if(i==n)
   return((c<w[n])?0:p[n]);
   if(c<w[i])
   return knapsack(i+1,c);
   return max(knapsack(i+1,c),knapsack(i+1,c-w[i])+p[i]);
}
void main()
{
   clrscr();
```

```
void main()
{
   clrscr();
   printf("Enter the number of objects\n");
   scanf("%d",&n);
   printf("Enter the weights\n");
   for(i=0;i<n;i++)
   {
      scanf("%d",&w[i]);
   }
   printf("Enter the profits\n");
   for(i=0;i<n;i++)
   {
      scanf("%d",&p[i]);
   }
   printf("Enter the capacity\n");
   scanf("%d",&m);
   maxprofit=knapsack(0,m);
   printf("Max profit = %d\n",maxprofit);
   getch();
}
```

OUTPUT :-

```
Enter the number of objects
3
Enter the weights
18 15 10
Enter the profits
25 24 15
Enter the capacity
20
Max profit = 25
```