PROGRAM NO.1

```
/*
PROGRAM NAME= WRITE A PROGRAM TO SORT POSITIVE  INTEGER
NUMBERS IN NON-DECREASING ORDER USING INSERTION SORT.
ROLL NO. = 17103011
DATE = 09/01/2019   */

#include<iostream>
using namespace std;
int main()
{
   int testcase;
   cout<<"enter no. of testcases\n";
   cin>>testcase;
   while(testcase--)
   {
   int i,j,flag=0,n,key,loop_counter=0;
   cout<<"enter size of array\n";
   cin>>n;
   int a[n];
   cout<<"enter array\n";
   for(i=0;i<n;i++)
   {
      cin>>a[i];
      if(a[i]<0)
         flag=1;
   }
   if(flag)
      cout<<"invalid input\n";
   else
   {
   for(i=0;i<n;i++)
   {
      loop_counter++;
      key=a[i+1];
      for(j=i;j>=0;j--)
      {
         loop_counter++;
         if(key<a[j])
         a[j+1]=a[j];
         else
            break;
      }
     a[j+1]=key;
   }
   cout<<"sorted array is \n";
   for(i=0;i<n;i++)
   {
```

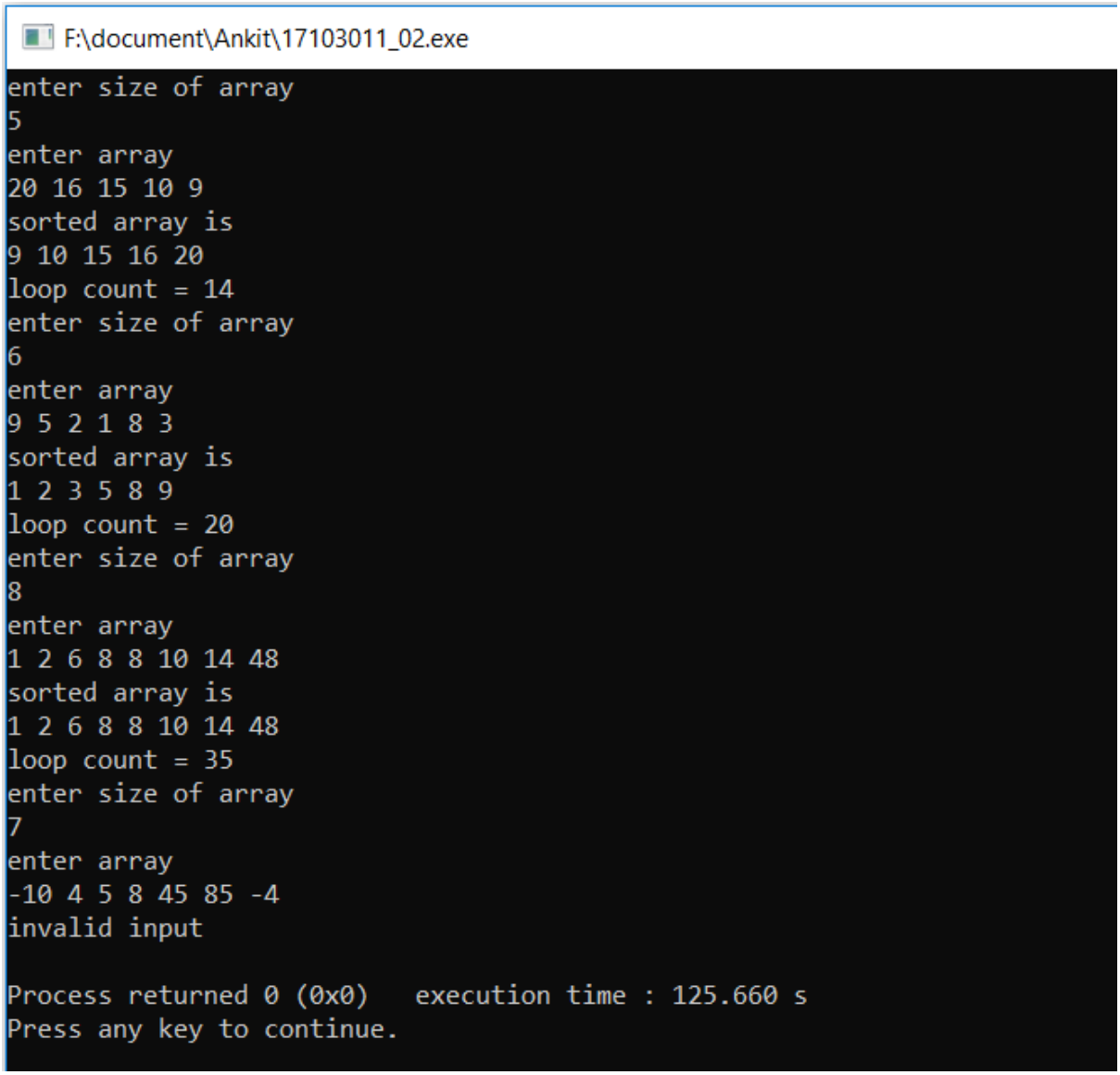
```
    cout<<a[i]<<" ";
  }
  cout<<"\nloop count = "<<loop_counter<<"\n";}
  }
  }
```

```
F:\document\Ankit\17103011_01.exe
enter size of array
7
enter array
9 5 2 1 4 8 3
sorted array is
1 2 3 4 5 8 9
loop count = 24
enter size of array
8
enter array
1 2 2 8 6 4 4 7
sorted array is
1 2 2 4 4 6 7 8
loop count = 22
enter size of array
5
enter array
20 16 15 14 13
sorted array is
13 14 15 16 20
loop count = 16
enter size of array
7
enter array
-10 45 -5 3 4 1 9
invalid input

Process returned 0 (0x0)   execution time : 113.324 s
Press any key to continue.
```

PROGRAM NO.2

```
/*
PROGRAM NAME= WRITE A PROGRAM TO SORT POSITIVE INTEGER
NUMBERS  IN NON-DECREASING ORDER USING SELECTION SORT.
ROLL NO. = 17103011
DATE = 09/01/2019   */

#include<iostream>
using namespace std;
int main()
{
   int testcase;
   cout<<"enter no. of testcases\n";
   cin>>testcase;
   while(testcase--)
   {
   int i,j,flag=0,k,n,minimum,loop_counter=0;
   cout<<"enter size of array\n";
   cin>>n;
   int a[n];
   cout<<"enter array\n";
   for(i=0;i<n;i++)
   {
      cin>>a[i];
      if(a[i]<0)
         flag=1;
   }
   if(flag)
      cout<<"invalid input\n";
   else
   {
   for(i=0;i<n-1;i++)
   {
      loop_counter++;
      k=i;
      for(j=i+1;j<n;j++)
      {
         loop_counter++;
         if(a[j]<a[k])
         {
            k=j;
         }
      }
      j=a[k];
      a[k]=a[i];
      a[i]=j;
   }
   cout<<"sorted array is \n";
   for(i=0;i<n;i++)
   {
```

```
        cout<<a[i]<<" ";
    }
    cout<<"\nloop count = "<<loop_counter<<"\n";
    }
    }
}
```



PROGRAM NO.3

```
/*
PROGRAM NAME = WRITE A PROGRAM TO SORT INTEGER NUMBERS IN
NON-DECREASING ORDER USING MERGE SORT
ROLL NO. = 17103011
DATE = 16/01/2019   */

#include<iostream>
using namespace std;
int counter;
void merge(int arr[],int lower,int mid,int uper)
{
   int size=uper-lower+1;
   int size_left=mid-lower+1;
   int size_right=uper-mid;
   int array_left[size_left+1],array_right[size_right+1];
   int i,j,k;
   for(i=0;i<size_left;i++)
      array_left[i]=arr[lower+i];
   for(j=0;j<size_right;j++)
      array_right[j]=arr[mid+1+j];
   array_left[i]=INT_MAX;
   array_right[j]=INT_MAX;
   i=j=0;
   for(k=0;k<size;k++)
   {
      counter++;
      if(array_left[i]<=array_right[j])
      {
         arr[lower+k]=array_left[i];
         i++;
      }
      else
      {
         arr[lower+k]=array_right[j];
         j++;
      }
   }
}
void merge_sort(int arr[],int l_bound,int u_bound)
{
   if(l_bound<u_bound)
   {
      int mid=(l_bound+u_bound)/2;
      merge_sort(arr,l_bound,mid);
      merge_sort(arr,mid+1,u_bound);
      merge(arr,l_bound,mid,u_bound);
   }
}
int main()
```

```
{
   int testcases;
   cout<<"enter no. of testcases\n";
   cin>>testcases;
   while(testcases--)
   {
      int size,i=0,j,flag=0;
      cout<<"enter size of array\n";
      cin>>size;
      int arr[size];
      cout<<"enter array\n";
      cin>>arr[i];
      i++;
      if(arr[0]>=0)
      {
         flag=1;
        while(i<size)
       {
          cin>>arr[i];
          if(arr[i]<0)
            flag=3;
          i++;
       }
      }
      else
      {
         flag=2;
         while(i<size)
         {
            cin>>arr[i];
            if(arr[i]>=0)
               flag=3;
            i++;
         }
      }
      if(flag==3)
         cout<<"invalid input\n";
      else
      {
         counter=0;
         merge_sort(arr,0,size-1);
         i=0;
         cout<<"sorted array is \n";
         while(i<size)
         {
            cout<<arr[i]<<" ";
            i++;
         }
         cout<<"\nno. of comparison = "<<counter;
         cout<<"\n";
```

```
    }
  }
}
```

F:\document\Ankit\17103011_03.exe

```
enter no. of testcases
6
enter size of array
5
enter array
-5 -9 -8 -45 -1
sorted array is
-45 -9 -8 -5 -1
no. of comparison = 12
enter size of array
5
enter array
1 6 9 45 58
sorted array is
1 6 9 45 58
no. of comparison = 12
enter size of array
6
enter array
-5 8 4 -10 9 45
invalid input
enter size of array
7
enter array
11 22 33 11 45 78 98
sorted array is
11 11 22 33 45 78 98
no. of comparison = 20
enter size of array
4
enter array
20 15 11 8
sorted array is
8 11 15 20
no. of comparison = 8
enter size of array
5
enter array
14 45 65 78 89
sorted array is
14 45 65 78 89
no. of comparison = 12

Process returned 0 (0x0)   execution time : 212.927 s
Press any key to continue.
```

PROGRAM NO.4

```
/*
PROGRAM NAME = WRITE A PROGRAM TO SORT NON REPEATING
INTEGERS IN NON DECREASING ORDER USING QUICK SORT
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 23/01/2019     */

#include<bits/stdc++.h>
#include<map>
using namespace std;
int partition(int[],int,int);
void quick_sort(int array[],int first,int last)
{
   if(first<last)
   {
      int pivot=partition(array,first,last);
      quick_sort(array,first,pivot-1);
      quick_sort(array,pivot+1,last);
   }
}
int comparision;
int partition(int array[],int low,int high)
{
   int i,j,k;
   i=low-1;
   int pivot=array[high];
   for(j=low;j<high;j++)
   {
      comparision++;
      if(array[j]<pivot)
      {
         i++;
         k=array[j];
         array[j]=array[i];
         array[i]=k;
      }
   }
   k=array[i+1];
   array[i+1]=array[high];
   array[high]=k;
   return (i+1);
}

int main()
{
   int testcases;
   cout<<"enter number of testcases\n";
   cin>>testcases;
   while(testcases--)
```

```
    {
       comparision=0;
    int i,j,size,flag=0;
    map<int,int> m;
    cout<<"enter size of array\n";
    cin>>size;
    int array[size];
    i=0;
    cout<<"enter array\n";
    while(i<size)
    {
       cin>>array[i];
       m[array[i]]++;
       if(m[array[i]]>1)
       {
          flag=1;
       }
       i++;
    }
    if(flag==1)
       cout<<"invalid input\n";
    else
    {
       quick_sort(array,0,size-1);
       cout<<"sorted array is\n";
       for(i=0;i<size;i++)
          cout<<array[i]<<" ";
       cout<<"\nno. of comparisions = "<<comparision<<"\n";
    }
    }
}
```

"F:\document\Ankit\17103011_04(quick sort).exe"

```
enter number of testcases
5
enter size of array
5
enter array
1 3 5 9 10
sorted array is
1 3 5 9 10
no. of comparisions = 10
enter size of array
5
enter array
-8 -4 -10 -3 -15
sorted array is
-15 -10 -8 -4 -3
no. of comparisions = 8
enter size of array
5
enter array
4 9 9 10 56
invalid input
enter size of array
5
enter array
45 25 13 9 4
sorted array is
4 9 13 25 45
no. of comparisions = 10
enter size of array
5
enter array
-8 -5 -8 -2 -1
invalid input

Process returned 0 (0x0)   execution time : 914.669 s
Press any key to continue.
```

PROGRAM NO.5

```
/*
PROGRAM NAME = WRITE A PROGRAM TO SORT INTEGERS IN NON DECREASING ORDER USING HEAP SORT
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 30/01/2019  */

#include<bits/stdc++.h>
using namespace std;
void max_heapify(int array[],int i,int size)
{
   int j,left,right,largest;
   left=2*i+1;
   right=2*i+2;
   largest=i;
   if(array[largest]<array[left]&&left<size)
   {
      largest=left;
   }
   if(array[largest]<array[right]&&right<size)
   {
      largest=right;
   }
   if(largest!=i)
   {
      j=array[largest];
      array[largest]=array[i];
      array[i]=j;
      max_heapify(array,largest,size);
   }
}
int loop_count;
void heap_sort(int array[],int size)
{
   int i,j;
   for(i=size/2-1;i>=0;i--)
   {
      loop_count++;
      max_heapify(array,i,size);
   }
   for(i=size-1;i>=0;i--)
   {
      loop_count++;
      j=array[0];
      array[0]=array[i];
      array[i]=j;
      max_heapify(array,0,i);
   }
}
```

```
int main()
{
   int testcases;
   cout<<"enter no. of testcases\n";
   cin>>testcases;
   while(testcases--)
   {
      int i,j,k,size;
      loop_count=0;
      cout<<"enter no. of elements\n";
      cin>>size;
      int array[size];
      i=0;
      cout<<"enter array\n";
      while(i<size)
      {
         cin>>array[i];
         i++;
      }
      heap_sort(array,size);
      cout<<"sorted array is \n";
      i=0;
      while(i<size)
      {
         cout<<array[i]<<" ";
         i++;
      }
      cout<<"\ntotal no. of loops = "<<loop_count<<"\n";
   }
}
```

"F:\document\Ankit\17103011_05(heap sort).exe"

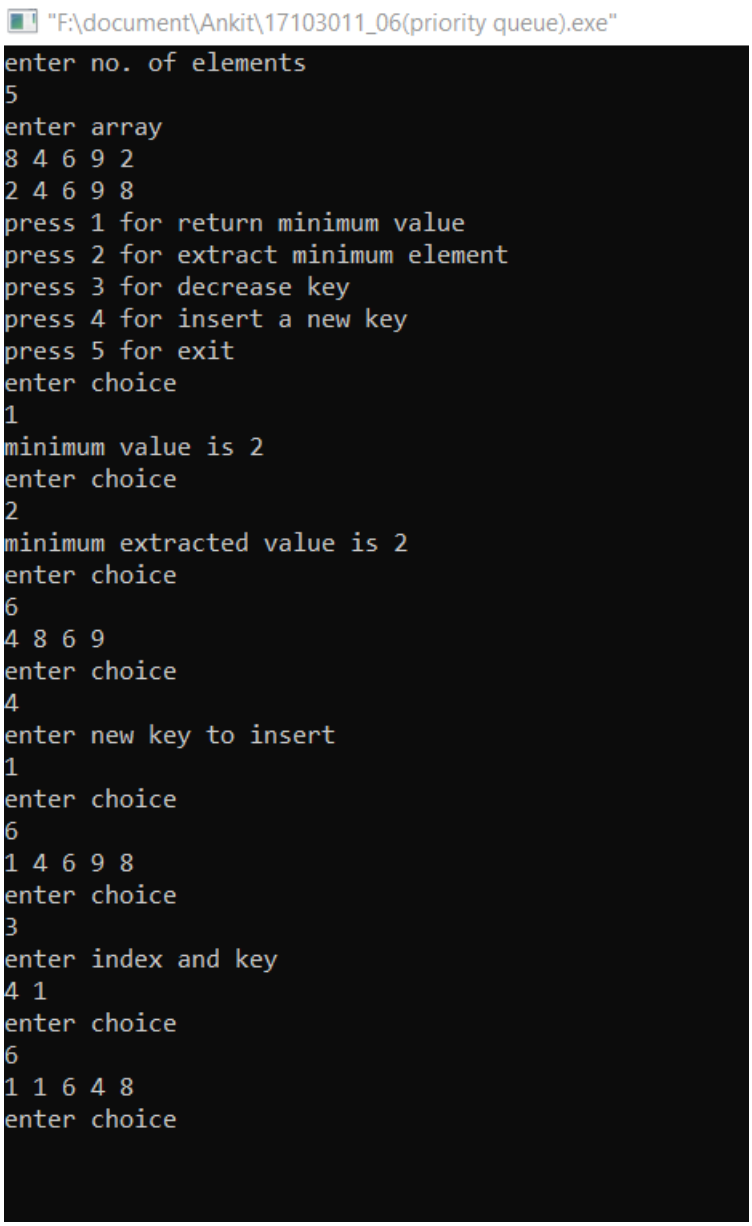
```
enter no. of testcases
5
enter no. of elements
5
enter array
15 25 65 18 6
sorted array is
6 15 18 25 65
total no. of loops = 7
enter no. of elements
6
enter array
58 -89 -1 25 26 5
sorted array is
-89 -1 5 25 26 58
total no. of loops = 9
enter no. of elements
5
enter array
56 42 35 15 8
sorted array is
8 15 35 42 56
total no. of loops = 7
enter no. of elements
5
enter array
52 45 95 45 25
sorted array is
25 45 45 52 95
total no. of loops = 7
enter no. of elements
4
enter array
-8 -45 -24 -10
sorted array is
-45 -24 -10 -8
total no. of loops = 6

Process returned 0 (0x0)   execution time : 147.070 s
Press any key to continue.
```

PROGRAM NO.6

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT PRIORITY QUEUE USING MIN HEAP
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 13/02/2019    */

#include<bits/stdc++.h>
using namespace std;
void min_heapify(int array[],int i,int size)
{
   int j,left,right,least;
   left=2*i+1;
   right=2*i+2;
   least=i;
   if(array[least]>array[left]&&left<size)
      least=left;
   if(array[least]>array[right]&&right<size)
       least=right;
   if(least!=i)
   {
      j=array[least];
      array[least]=array[i];
      array[i]=j;
      min_heapify(array,least,size);
   }
}
int minimum(int array[])
{
   return array[0];
}
int extract_min(int array[],int &size)
{
   int minimum=array[0];
   array[0]=array[size-1];
   size=size-1;
   min_heapify(array,0,size);
   return minimum;
}
void decrease_key(int array[],int index,int key)
{
   if(key>array[index])
      cout<<"new key is larger than previous key\n";
   else
   {
      array[index]=key;
      while(index>0&&key<array[(index-1)/2])
      {
         int j=array[index];
```

```
            array[index]=array[(index-1)/2];
            array[(index-1)/2]=j;
            index=(index-1)/2;
        }
    }
}
void insert(int array[],int key,int &size)
{
    size=size+1;
    array[size-1]=INT_MAX;
    decrease_key(array,size-1,key);
}
void build_minheap(int array[],int size)
{
    int i;
    for(i=size/2-1;i>=0;i--)
        min_heapify(array,i,size);
}
int main()
{
    int i=0,temp,size;
    cout<<"enter no. of elements\n";
    cin>>size;
    int array[size];
    cout<<"enter array\n";
    while(i<size)
    {
        cin>>array[i];
        i++;
    }
    build_minheap(array,size);
    i=0;
    while(i<size)
    {
        cout<<array[i]<<" ";
        i++;
    }
    cout<<"\npress 1 for return minimum value\npress 2 for extract minimum element\n";
    cout<<"press 3 for decrease key\npress 4 for insert a new key\npress 6 to print queue\n";
    cout<<"press 5 for exit\nenter choice\n";
    int choice,index,key;
    cin>>choice;
    while(choice!=5)
    {
        switch(choice)
        {
        case 1:
            temp=minimum(array);
            cout<<"minimum value is "<<temp<<"\n";
            break;
```

```
        case 2:
           temp=extract_min(array,size);
           cout<<"minimum extracted value is "<<temp<<"\n";
           break;
        case 3:
           cout<<"enter index and key \n";
           cin>>index>>key;
           decrease_key(array,index-1,key);
           break;
        case 4:
           cout<<"enter new key to insert\n";
           cin>>key;
           insert(array,key,size);
           break;
        case 6:
           i=0;
            while(i<size)
            {
                  cout<<array[i]<<" ";
                   i++;
            }
            cout<<"\n";
            break;
        default:
           cout<<"invalid input\n";
        }
        cout<<"enter choice\n";
        cin>>choice;
    }
}
```


"F:\document\Ankit\17103011_06(priority queue).exe"
enter no. of elements
5
enter array
8 4 6 9 2
2 4 6 9 8
press 1 for return minimum value
press 2 for extract minimum element
press 3 for decrease key
press 4 for insert a new key
press 5 for exit
enter choice
1
minimum value is 2
enter choice
2
minimum extracted value is 2
enter choice
6
4 8 6 9
enter choice
4
enter new key to insert
1
enter choice
6
1 4 6 9 8
enter choice
3
enter index and key
4 1
enter choice
6
1 1 6 4 8
enter choice

PROGRAM NO.7

```
/*
PROGRAM NAME = WRITE A PROGRAM TO SORT IN NON DECREASING
ORDER USING COUNTING SORT
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 20/02/2019     */

#include<bits/stdc++.h>
using namespace std;
int main()
{
   int testcases;
   cout<<"enter no. of testcases\n";
   cin>>testcases;
   while(testcases--)
   {
      int size,i,max,min,range;
      cout<<"enter size of array\n";
      cin>>size;
      int array[size];
      cout<<"enter array\n";
      cin>>array[0];
      min=max=array[0];
      i=1;
      while(i<size)
      {
         cin>>array[i];
         if(array[i]>max)
            max=array[i];
         if(array[i]<min)
            min=array[i];
         i++;
      }
      range=max-min+1;
      int arr_range[range]={0};
      i=0;
      while(i<size)
      {
         arr_range[array[i]-min]+=1;
         i++;
      }
      i=0;
      int element=min-1,freq=0;
      while(i<size)
      {
         if(freq==0)
         {
            element++;
            freq=arr_range[element-min];
```

```
        }
        else
        {
           array[i]=element;
           freq--;
           i++;
        }
     }
     cout<<"sorted array is \n";
     i=0;
     while(i<size)
     {
        cout<<array[i]<<" ";
        i++;
     }
     cout<<"\n";
  }
}
```

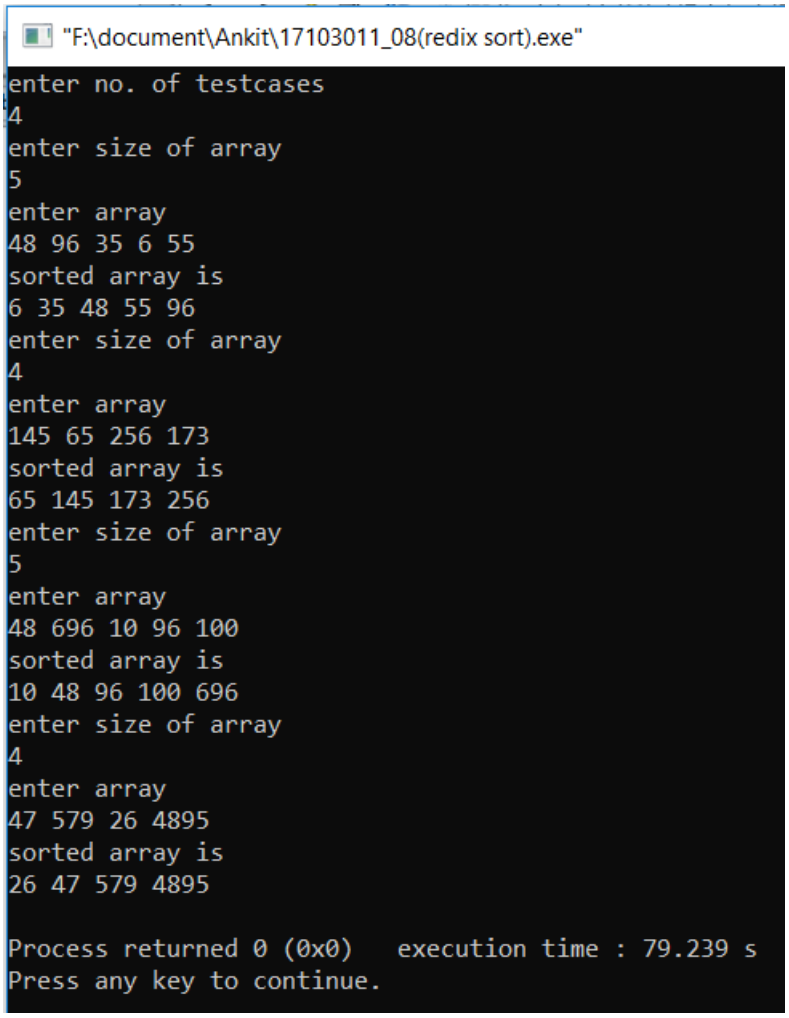
"F:\document\Ankit\17103011_07(counting sort).exe"

```
enter no. of testcases
5
enter size of array
5
enter array
1 6 89 5  4
sorted array is
1 4 5 6 89
enter size of array
8
enter array
8 4 6 92 1 9 65 16
sorted array is
1 4 6 8 9 16 65 92
enter size of array

5
enter array
8 4 6 -10 5
sorted array is
-10 4 5 6 8
enter size of array
5
enter array
-7 -5 -40 -16 -4
sorted array is
-40 -16 -7 -5 -4
enter size of array
4
enter array
5 6 9 12
sorted array is
5 6 9 12

Process returned 0 (0x0)   execution time : 380.877 s
Press any key to continue.
```

PROGRAM NO.8

```
/*
PROGRAM NAME = WRITE A PROGRAM TO SORT IN NON DECREASING ORDER USING RADIX SORT
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 20/02/2019  */

#include<bits/stdc++.h>
using namespace std;
void counting_sort(int array[],int factor,int size)
{
   int i=0,count_array[10]={0};
   while(i<size)
   {
      count_array[(array[i]/factor)%10]++;
      i++;
   }
   i=1;
   while(i<10)
   {
      count_array[i]+=count_array[i-1];
      i++;
   }
   i=size-1;
   int temp_array[size];
   while(i>=0)
   {
      temp_array[count_array[(array[i]/factor)%10]-1]=array[i];
      count_array[(array[i]/factor)%10]--;
      i--;
   }
   i=0;
   while(i<size)
   {
      array[i]=temp_array[i];
      i++;
   }
}
int main()
{
   int testcases;
   cout<<"enter no. of testcases\n";
   cin>>testcases;
   while(testcases--)
   {
      int size,i,max;
      cout<<"enter size of array\n";
      cin>>size;
      int array[size];
```

```
    cout<<"enter array\n";
    cin>>array[0];
    max=array[0];
    i=1;
    while(i<size)
    {
       cin>>array[i];
       if(array[i]>max)
          max=array[i];
       i++;
    }
    int max_length=0;
    while(max>0)
    {
       max=max/10;
       max_length++;
    }
    int factor=1;
    while(max_length--)
    {
       counting_sort(array,factor,size);
       factor=factor*10;
    }
    cout<<"sorted array is \n";
    i=0;
    while(i<size)
    {
       cout<<array[i]<<" ";
       i++;
    }
    cout<<"\n";
  }
}
```


"F:\document\Ankit\17103011_08(redix sort).exe"
enter no. of testcases
4
enter size of array
5
enter array
48 96 35 6 55
sorted array is
6 35 48 55 96
enter size of array
4
enter array
145 65 256 173
sorted array is
65 145 173 256
enter size of array
5
enter array
48 696 10 96 100
sorted array is
10 48 96 100 696
enter size of array
4
enter array
47 579 26 4895
sorted array is
26 47 579 4895
Process returned 0 (0x0) execution time : 79.239 s
Press any key to continue.

PROGRAM NO.9

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT HASH FUNCTION
(DIVISION METHOD)
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 27/02/2019  */
#include<bits/stdc++.h>
using namespace std;
struct node
{
   int info;
   node *next;
};
int main()
{
   int size,i,choice,temp1,temp2 ,flag=0,count=0;;
   node *p1,*p2;
   cout<<"press 1 for insert element\npress 2 for searching element\n";
   cout<<"press 3 for exit\nenter choice\n";
   cin>>choice;
   node *array[100]={NULL};
   while(choice!=3)
   {
      switch(choice)
      {
      case 1:
         cout<<"enter number\n";
         cin>>temp1;
         p1=new node;
         p1->info=temp1;
         p1->next=NULL;
         temp1=temp1%100;
         if(array[temp1]==NULL)
            array[temp1]=p1;
         else
         {
            p2=array[temp1];
            while(p2->next!=NULL)
               p2=p2->next;
            p2->next=p1;
         }
         break;
      case 2:
         cout<<"enter number\n";
         cin>>temp1;
         temp2=temp1%100;
         p1=array[temp2];
         flag=0,count=0;
         while(p1!=NULL)
```

```
        {
            count++;
            if(p1->info==temp1)
            {
                flag=1;
                break;
            }
            p1=p1->next;
        }
        if(count>1)
            cout<<"collision occured\n";
        if(flag==1)
            cout<<temp1<<" is present in the table\n";
        else
            cout<<temp1<<" is not present in the table\n";
        break;
    }
    cout<<"enter choice\n";
    cin>>choice;
  }
}
```

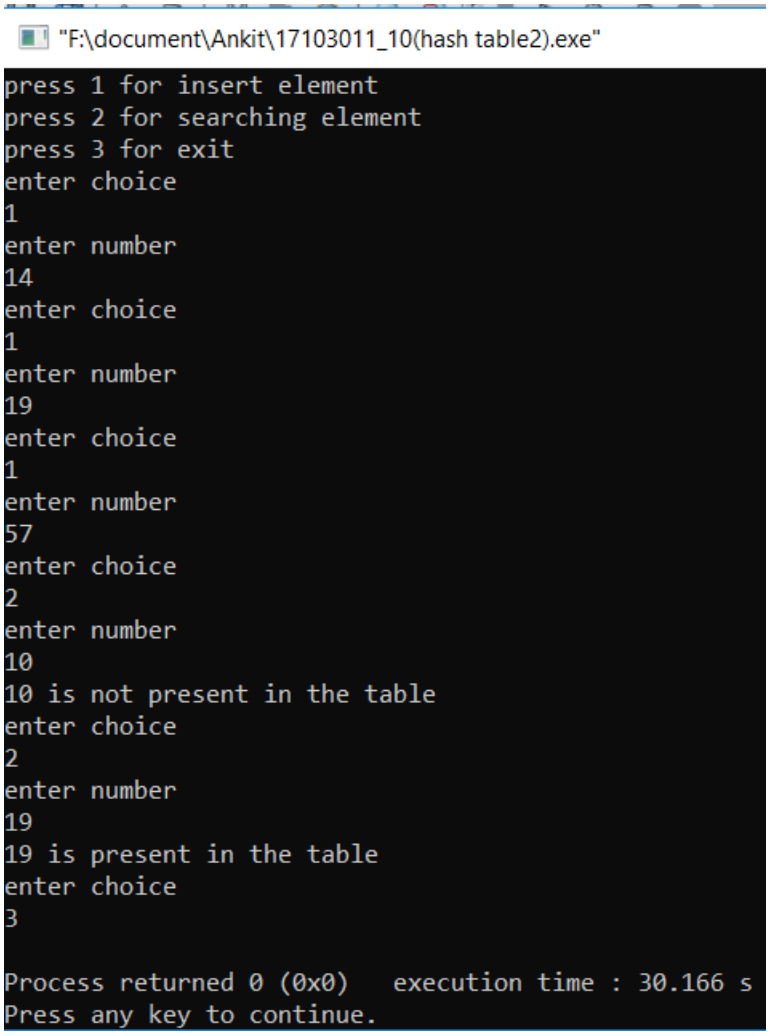
"F:\document\Ankit\17103011_09(hash table).exe"

```
press 1 for insert element
press 2 for searching element
press 3 for exit
enter choice
1
enter number
48
enter choice
1
enter number
56
enter choice
1
enter number
78
enter choice
1
enter number
148
enter choice
2
enter number
48
48 is present in the table
enter choice
2
enter number
148
collision occured
148 is present in the table
enter choice
3

Process returned 0 (0x0)   execution time : 316.053 s
Press any key to continue.
```

PROGRAM NO.10

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT HASH FUNCTION
(MULTIPLICATION METHOD)
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 27/02/2019  */

#include<bits/stdc++.h>
using namespace std;
struct node
{
   int info;
   node *next;
};
int main()
{
   int size,i,choice,temp1,temp2,hash;
   float temp;
   node *p1,*p2;
   cout<<"press 1 for insert element\npress 2 for searching element\n";
   cout<<"press 3 for exit\nenter choice\n";
   cin>>choice;
   node *array[100]={NULL};
   int flag=0,count=0;
   while(choice!=3)
   {
      switch(choice)
      {
      case 1:
         cout<<"enter number\n";
         cin>>temp1;
         p1=new node;
         p1->info=temp1;
         p1->next=NULL;
         temp=temp1*0.43;
         temp2=temp;
         hash=64*(temp-temp2);
         temp1=hash;
         if(array[temp1]==NULL)
            array[temp1]=p1;
         else
         {
            p2=array[temp1];
            while(p2->next!=NULL)
               p2=p2->next;
            p2->next=p1;
         }
         break;
      case 2:
```

```
        cout<<"enter number\n";
        cin>>temp1;
        temp=temp1*0.43;
        temp2=temp;
        hash=64*(temp-temp2);
        temp2=hash;
        flag=0,count=0;
        p1=array[temp2];
        while(p1!=NULL)
        {
          count++;
          if(p1->info==temp1)
          {
            flag=1;
            break;
          }
          p1=p1->next;
        }
        if(count>1)
          cout<<"collision occured\n";
        if(flag==1)
          cout<<temp1<<" is present in the table\n";
        else
          cout<<temp1<<" is not present in the table\n";
        break;
      default:
        cout<<"invalid choice\n";
      }
      cout<<"enter choice\n";
      cin>>choice;
    }
}
```


"F:\document\Ankit\17103011_10(hash table2).exe"
press 1 for insert element
press 2 for searching element
press 3 for exit
enter choice
1
enter number
14
enter choice
1
enter number
19
enter choice
1
enter number
57
enter choice
2
enter number
10
10 is not present in the table
enter choice
2
enter number
19
19 is present in the table
enter choice
3
Process returned 0 (0x0) execution time : 30.166 s
Press any key to continue.

PROGRAM NO.11

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT INSERTION IN RED BLACK TREE
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 06/03/2019  */

#include<bits/stdc++.h>
using namespace std;
struct node
{
   int info;
   char colour;
   node *left,*right,*parent;
};
node *root='\0';
void insert_node(node *&root,node *p1)
{
  if(root=='\0')
   root=p1;
  else
  {
      if(p1->info<root->info)
      {
         if(root->left=='\0')
         {
            p1->parent=root;
            root->left=p1;
         }
         else
            insert_node(root->left,p1);
      }
      else
      {
         if(root->right=='\0')
         {
            p1->parent=root;
            root->right=p1;
         }
         else
            insert_node(root->right,p1);
      }
  }
}
void rotate_left(node *t)
{
   node *child=t->left;
   t->left=child->right;
   if(t->left!='\0')
```

```
        t->left->parent=t;
    child->right=t;
    node *temp=t->parent;
    t->parent=child;
    if(temp=='\0')
    {
        child->parent='\0';
        root=child;
    }
    else if(temp->left==t)
    {
        temp->left=child;
        child->parent=temp;
    }
    else
    {
        temp->right=child;
        child->parent=temp;
    }
}
void rotate_right(node *t)
{
    node *child=t->right;
    t->right=child->left;
    if(t->right!='\0')
        t->right->parent=t;
    child->left=t;
    node *temp=t->parent;
    t->parent=child;
    if(temp=='\0')
    {
        child->parent='\0';
        root=child;
    }
    else if(temp->left==t)
    {
        temp->left=child;
        child->parent=temp;
    }
    else
    {
        temp->right=child;
        child->parent=temp;
    }
}
void set_node(node *&root,node *&t)
{
    if(t->parent=='\0')
    {
        t->colour='b';
```

```
        return;
    }
    while(t->parent!='\0'&&t->parent->colour!='b')
    {
        node *uncle,*par=t->parent,*g_par=t->parent->parent;
        if(g_par->left==par)
            uncle=g_par->right;
        else
            uncle=g_par->left;
        if(uncle!='\0'&&uncle->colour=='r')
        {
            par->colour='b';
            uncle->colour='b';
            g_par->colour='r';
            t=g_par;
        }
        else
        {
            if(g_par->left==par)
            {
                if(par->right==t)
                {
                    t=t->parent;
                    rotate_right(t);
                }
                t->parent->colour='b';
                g_par->colour='r';
                rotate_left(g_par);
            }
            else
            {
                if(par->left==t)
                {
                    t=t->parent;
                    rotate_left(t);
                }
                t->parent->colour='b';
                g_par->colour='r';
                rotate_right(g_par);
            }
        }
    }
    root->colour='b';
}
void print_tree(node *root)
{
    if(root->left!='\0')
        print_tree(root->left);
    cout<<root->info<<" "<<root->colour<<"\n";
    if(root->right!='\0')
```

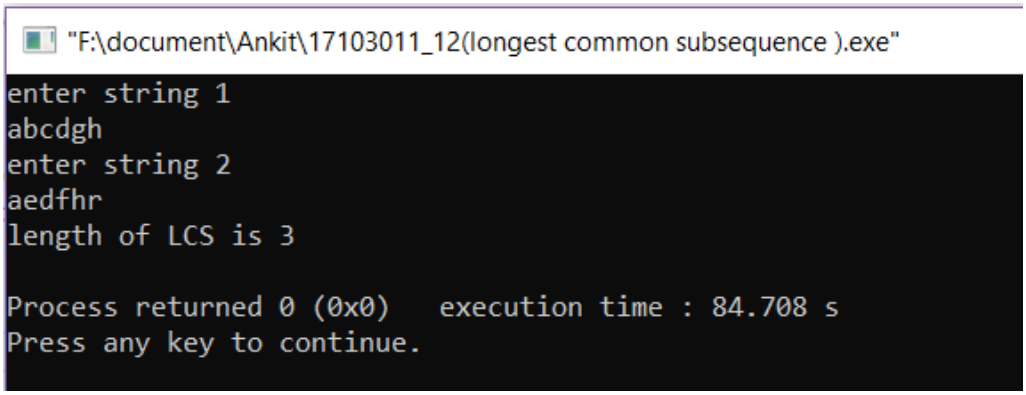
```
        print_tree(root->right);
}
int main()
{
   int i,choice,temp1,temp2,hash;
   node *p1,*p2;
   cout<<"press 1 for insert element\npress 2 for searching element\n";
   cout<<"press 4 for print tree\npress 5 for exit\nenter choice\n";
   cin>>choice;
   while(choice!=5)
   {
      switch(choice)
      {
      case 1:
         cout<<"enter number\n";
         cin>>temp1;
         p1=new node;
         p1->info=temp1;
         p1->left=p1->right=p1->parent='\0';
         p1->colour='r';
         insert_node(root,p1);
         set_node(root,p1);
         break;
      case 2:
         cout<<"enter no. to search\n";
         cin>>temp1;
         p1=root;
         while(p1!='\0'&&p1->info!=temp1)
         {
            if(p1->info<temp1)
               p1=p1->right;
            if(p1->info>temp1)
               p1=p1->left;
         }
         if(p1=='\0')
            cout<<temp1<<" is not present in the tree\n";
         else
            cout<<p1->info<<" "<<p1->colour<<" "<<p1->parent<<"\n";
         break;
      case 4:
         print_tree(root);
         cout<<"\n";
         break;
      default:
         cout<<"invalid choice\n";
      }
      cout<<"enter choice\n";
      cin>>choice;
   }
}
```

"F:\document\Ankit\17103011_11(red black tree).exe"

```
press 1 for insert element
press 2 for searching element
press 4 for print tree
press 5 for exit
enter choice
1
enter number
10
enter choice
1
enter number
8
enter choice
1
enter number
15
enter choice
1
enter number
24
enter choice
1
enter number
9
enter choice
1
enter number
4
enter choice
1
enter number
20
enter choice
4
4 r
8 b
9 r
10 b
15 r
20 b
24 r

enter choice
5

Process returned 0 (0x0)   execution time : 196.855 s
Press any key to continue.
```

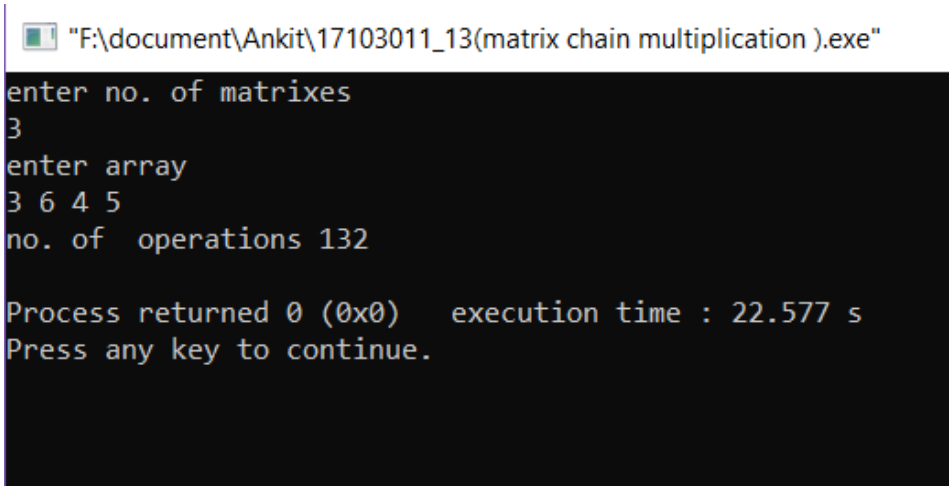
PROGRAM NO.12

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT LONGEST COMMON SUBSEQUENCE
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 13/03/2019  */

#include<bits/stdc++.h>
using namespace std;
int lcs(string s1,string s2,int length1,int length2)
{
   int m[length1+1][length2+1];
   int i,j;
   for(i=0;i<=length1;i++)
   {
      for(j=0;j<=length2;j++)
      {
         if(i==0||j==0)
            m[i][j]=0;
         else if(s1[i-1]==s2[j-1])
            m[i][j]=1+m[i-1][j-1];
         else
            m[i][j]=max(m[i-1][j],m[i][j-1]);
      }
   }
   return m[length1][length2];
}
int main()
{
   string s1,s2;
   cout<<"enter string 1\n";
   cin>>s1;
   cout<<"enter string 2\n";
   cin>>s2;
   int a=lcs(s1,s2,s1.length(),s2.length());
   cout<<"length of LCS is "<<a<<"\n";
}
```



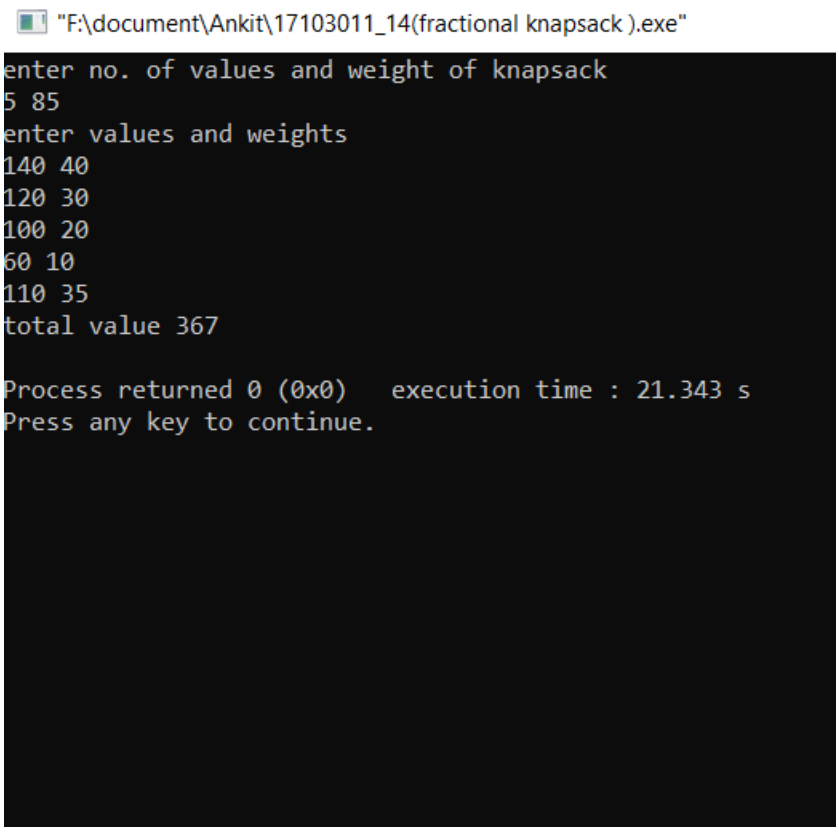
PROGRAM NO.13

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT MATRIX CHAIN MULTIPLICATION
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 03/04/2019  */

#include<bits/stdc++.h>
using namespace std;
int mcm(int array[],int size)
{
   int m[size][size];
   int i,j,k,cost;
   for(i=1;i<size;i++)
      m[i][i]=0;
   for(int l=2;l<size;l++)
   {
      for(i=1;i<size-l+1;i++)
      {
         j=i+l-1;
         m[i][j]=INT_MAX;
         for(k=i;k<j;k++)
         {
            cost=m[i][k]+m[k+1][j]+array[i-1]*array[k]*array[j];
            if(cost<m[i][j])
               m[i][j]=cost;
         }
      }
   }
   return m[1][size-1];
}
int main()
{
   int n;
   cout<<"enter no. of matrixes\n";
   cin>>n;
   int a[n+1];
   cout<<"enter array\n";
   for(int i=0;i<n+1;i++)
      cin>>a[i];
   cout<<"no. of  operations "<<mcm(a,n+1)<<"\n";
}
```


"F:\document\Ankit\17103011_13(matrix chain multiplication).exe"
enter no. of matrixes
3
enter array
3 6 4 5
no. of operations 132
Process returned 0 (0x0) execution time : 22.577 s
Press any key to continue.

PROGRAM NO.14

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT FRACTIONAL KNAPSACK PROBLEM
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 10/04/2019  */

#include<bits/stdc++.h>
using namespace std;
struct node
{
   float ratio,value,weight;
};
bool compare(node n1,node n2)
{
   return n1.ratio>n2.ratio;
}
int fractional_knapsack(node array[],int w,int size)
{
   sort(array,array+size,compare);
   int i=0,total_value=0,current_w=0;
   while(i<size&&(current_w+array[i].weight)<=w)
   {
      current_w+=array[i].weight;
      total_value+=array[i].value;
      i++;
   }
   if(i<size&&w-current_w>0)
       total_value+=(w-current_w)*array[i].ratio;
   return total_value;
}
int main()
{
   int n,w;
   cout<<"enter no. of values and weight of knapsack\n";
   cin>>n>>w;
   node array[n];
   cout<<"enter values and weights\n";
   for(int i=0;i<n;i++)
   {
      cin>>array[i].value>>array[i].weight;
      array[i].ratio=array[i].value/array[i].weight;
   }
   cout<<"total value "<<fractional_knapsack(array,w,n)<<"\n";
}
```


"F:\document\Ankit\17103011_14(fractional knapsack).exe"
enter no. of values and weight of knapsack
5 85
enter values and weights
140 40
120 30
100 20
60 10
110 35
total value 367
Process returned 0 (0x0) execution time : 21.343 s
Press any key to continue.

PROGRAM NO.15

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT 0-1 KNAPSACK PROBLEM
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 10/04/2019  */

#include<bits/stdc++.h>
using namespace std;
int knapsack(int value[],int weight[],int w,int size)
{
   int i,j,k,m[size+1][w+1];
   for(i=0;i<size+1;i++)
   {
      for(j=0;j<w+1;j++)
      {
         if(i==0||j==0)
            m[i][j]=0;
         else if(weight[i-1]>j)
            m[i][j]=m[i-1][j];
         else
            m[i][j]=max(m[i-1][j],m[i-1][j-weight[i-1]]+value[i-1]);
      }
   }
   return m[size][w];
}
int main()
{
   int n,w;
   cout<<"enter no. of values and weight of knapsack\n";
   cin>>n>>w;
   int value[n],weight[n];
   cout<<"enter values and weights\n";
   for(int i=0;i<n;i++)
      cin>>value[i]>>weight[i];
   cout<<"total value "<<knapsack (value ,weight ,w ,n)<<"\n";
}
```

```
"F:\document\Ankit\17103011_15(0-1 knapsack ).exe"
enter no. of values and weight of knapsack
4 7
enter values and weights
1 1
4 3
5 4
7 5
total value 9

Process returned 0 (0x0)   execution time : 41.693 s
Press any key to continue.
```

PROGRAM NO.16

```
/*
PROGRAM NAME = WRITE A PROGRAM TO IMPLEMENT DEPTH FIRST
SEARCH AND BREADTH FIRST SEARCH IN GRAPH
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 24/04/2019  */

#include<bits/stdc++.h>
using namespace std;
struct node
{
   int value,status;
};
void dfs(node array[],int **matrix,int n,int edges)
{
   int i=0,j,k;
   node temp;
   stack <node> s;
   while(i<n)
   {
      if(array[i].status==0)
      {
         s.push(array[i]);
         while(!s.empty())
         {
            temp=s.top();
            s.pop();
            j=0;
            while(j<n)
            {
               if(temp.value==array[j].value)
                  break;
               j++;
            }
            array[j].status=2;
            cout<<array[j].value<<" ";
            k=0;
            while(k<n)
            {
               if(matrix[j][k]==1&&array[k].status==0)
               {
                  array[k].status=1;
                  s.push(array[k]);
               }
               k++;
            }
         }
      }
      i++;
```

```
    }
}
void bfs(node array[],int **matrix,int n,int edges)
{
    int i=0,j,k;
    node temp;
    queue <node> q;
    while(i<n)
    {
        if(array[i].status==0)
        {
            q.push(array[i]);
            while(!q.empty())
            {
                temp=q.front();
                q.pop();
                j=0;
                while(j<n)
                {
                    if(temp.value==array[j].value)
                        break;
                    j++;
                }
                array[j].status=2;
                cout<<array[j].value<<" ";
                k=0;
                while(k<n)
                {
                    if(matrix[j][k]==1&&array[k].status==0)
                    {
                        array[k].status=1;
                        q.push(array[k]);
                    }
                    k++;
                }
            }
        }
        i++;
    }
}
int main()
{
    int n,edges;
    cout<<"enter no. of nodes and edges\n";
    cin>>n>>edges;
    cout<<"enter nodes\n";
    node array[n];
    int i=0,j;
    while(i<n)
    {
```

```
        cin>>array[i].value;
        array[i].status=0;
        i++;
    }
    cout<<"enter edges\n";
    int **matrix=(int**)malloc(sizeof(int*)*n);
    for(i=0;i<n;i++)
        matrix[i]=(int*)malloc(sizeof(int)*n);
     for(i=0;i<n;i++)
        for(j=0;j<n;j++)
            matrix[i][j]=0;
    int temp1,temp2;
    i=0;
    while(i<edges)
    {
        cin>>temp1>>temp2;
        matrix[temp1][temp2]=1;
        i++;
    }
    cout<<"DFS = ";
    dfs(array,matrix,n,edges);
    cout<<"\n";
    for(j=0;j<n;j++)
        array[j].status=0;
    cout<<"BFS = ";
    bfs(array,matrix,n,edges);
}
```

```
"F:\document\Ankit\17103011_16(DFS and BFS).exe"
enter no. of nodes and edges
6 8
enter nodes
3 7 11 15 10 8
enter edges
0 1
1 2
3 2
3 4
4 5
2 4
2 5
0 5
DFS = 3 8 7 11 10 15
BFS = 3 7 8 11 10 15
Process returned 0 (0x0)   execution time : 9.369 s
Press any key to continue.
```

PROGRAM NO.17

```
/*
PROGRAM NAME = WRITE A PROGRAM TO FIND MINIMUM SPANNING
TREE USING PRIM'S ALGORITHM
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 24/04/2019  */

#include<bits/stdc++.h>
using namespace std;
int find_min(int value[],int flag[],int n)
{
   int i=0,min=INT_MAX,index=-1;
   while(i<n)
   {
      if(flag[i]==0&&value[i]<min)
      {
         min=value[i];
         index=i;
      }
      i++;
   }
   return index;
}
void prim(int** matrix,int n)
{
   int value[n],par[n],flag[n],count=0,i=0,index;
   while(i<n)
   {
      value[i]=INT_MAX;
      par[i]=-1;
      flag[i]=0;
      i++;
   }
   value[0]=0;
   while(count<n-1)
   {
      index=find_min(value,flag,n);
      flag[index]=1;
      i=0;
      while(i<n)
      {
         if(matrix[index][i]>0&&flag[i]==0&&matrix[index][i]<value[i])
         {
            par[i]=index;
            value[i]=matrix[index][i];
         }
         i++;
      }
      count++;
```

```
    }
    cout<<"included edges are\n";
    i=1;
    while(i<n)
    {
        cout<<i<<"---"<<par[i]<<"\n";
        i++;
    }
}
int main()
{
    int n,edges,i=0,j;
    cout<<"enter no. of nodes and edges\n";
    cin>>n>>edges;
    cout<<"enter edge and weight of edge\n";
    int **matrix=(int**)malloc(sizeof(int*)*n);
    for(i=0;i<n;i++)
        matrix[i]=(int*)malloc(sizeof(int)*n);
     for(i=0;i<n;i++)
        for(j=0;j<n;j++)
            matrix[i][j]=0;
    int temp1,temp2,temp3;
    i=0;
    while(i<edges)
    {
        cin>>temp1>>temp2>>temp3;
        matrix[temp1][temp2]=temp3;
        matrix[temp2][temp1]=temp3;
        i++;
    }
    prim(matrix,n);
}
```

"F:\document\Ankit\17103011_17(prim algo).exe"

```
enter no. of nodes and edges
9 14
enter edge and weight of edge
0 1 4
0 7 8
1 7 11
1 2 8
7 8 7
7 6 1
2 8 2
8 6 6
2 3 7
2 5 4
5 6 2
3 5 14
3 4 9
5 4 10
included edges are
1---0
2---1
3---2
4---3
5---2
6---5
7---6
8---2

Process returned 0 (0x0)   execution time : 2.113 s
Press any key to continue.
```

PROGRAM NO.18

```
/*
PROGRAM NAME = WRITE A PROGRAM TO FIND MINIMUM SPANNING
TREE USING KRUSKAL'S ALGORITHM
ROLL NO. = 17103011
NAME = ANKIT GOYAL
DATE = 24/04/2019  */

#include<bits/stdc++.h>
using namespace std;
struct edge
{
   int s,d,value;
};
bool compare(edge e1,edge e2)
{
   if(e1.value>e2.value)
      return 0;
   return 1;
}
int find_parent(int par[],int temp,int n)
{
   while(par[temp]!=-1)
      temp=par[temp];
   return temp;
}
void kruskal_algo(edge array[],int n,int edges)
{
   int par[n],count=0,i=0,index,set1,set2,temp3,edge_count=0;
   sort(array,array+edges,compare);
   for(i=0;i<n;i++)
      par[i]=-1;
   queue <int> q;
   while(count<n-1)
   {
      set1=find_parent(par,array[edge_count].s,n);
      set2=find_parent(par,array[edge_count].d,n);
      if(set1!=set2)
      {
         count++;
         q.push(edge_count);
         par[set1]=set2;
      }
      edge_count++;
   }
   cout<<"included edges are\nn1---n2  weight\n ";
   while(q.empty()!=1)
   {
      i=q.front();
```

```
        cout<<array[i].s<<"---"<<array[i].d<<"     "<<array[i].value<<"\n ";
        q.pop();
    }
}
int main()
{
    int n,edges,i=0,j;
    cout<<"enter no. of nodes and edges\n";
    cin>>n>>edges;
    cout<<"enter edge and weight of edge\n";
    edge array[edges];
    i=0;
    while(i<edges)
    {
        cin>> array[i].s>>array[i].d>>array[i].value;
        i++;
    }
    kruskal_algo(array,n,edges);
}
```