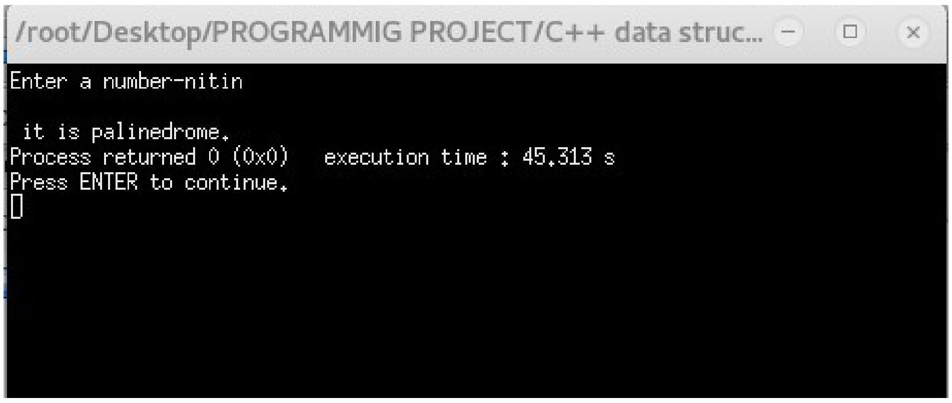
Note – please ignore the last two line in output, they are the default messages that occur in console window of “Code Blocks ” IDE, and there is no way to remove those.....

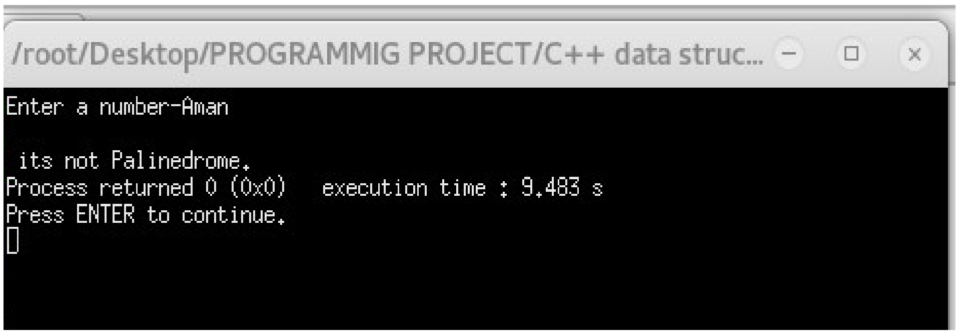
1.a.
Program-

```
#include<iostream>
#include<string.h>
#include<stdio.h>

using namespace std;

int main()
{
    char a[10];
    int flag = 0, pos=-1, temp;

    cout<<"Enter a number-";
    cin.getline(a, 10);

    for(int i=0; a[i]!= '\0';i++)
    {   pos++;  }

    for(int i=0; i<pos/2 ;i++)
    {   if(a[i] != a[pos-i])
            flag=1;
    }

    if(flag==1)
        cout<<"\n its not Palinedrome.";
    else
        cout<<"\n it is palinedrome.";

    return 0;
}
```

Output-



```
Enter a number-nitin

 it is palinedrome.
Process returned 0 (0x0)   execution time : 45.313 s
Press ENTER to continue.
```



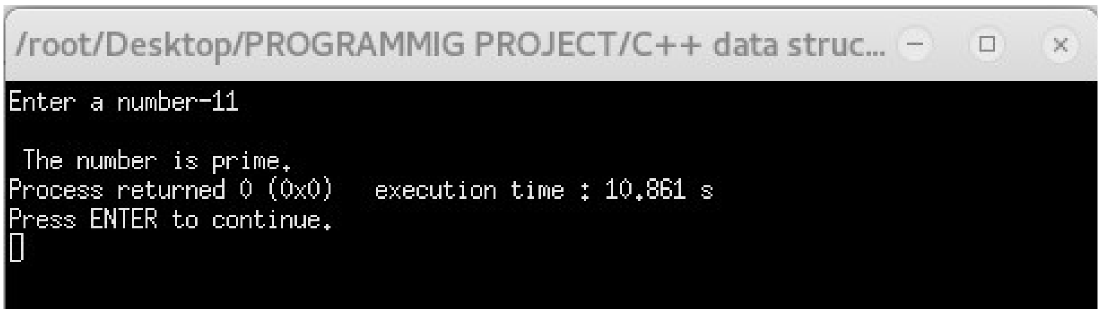
```
Enter a number-Aman

 its not Palinedrome.
Process returned 0 (0x0)   execution time : 9.483 s
Press ENTER to continue.
```

1.b.

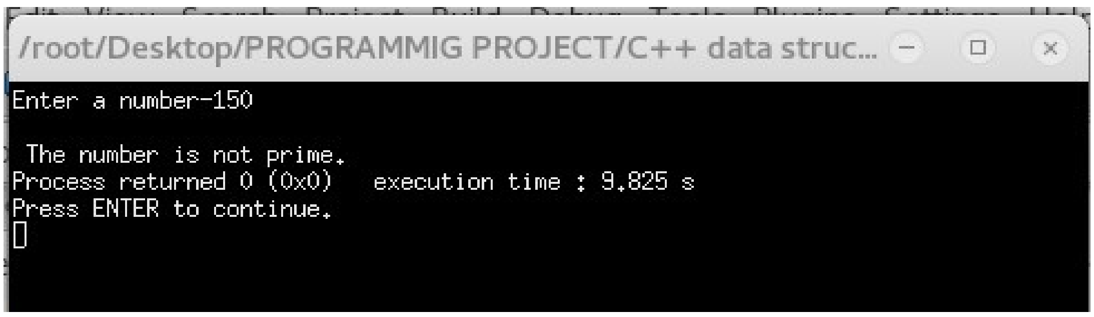
Program-

```
#include<iostream>

using namespace std;

int main()
{
    int num, flag=0;
    cout<<"Enter a number-";
    cin>>num;

    if(num % 2==0)
            flag=1;
    else
    {   for(int i=3; i<=num/2; i++)
        {
            if(num % i==0)
                flag=1;
        }
    }

    if(flag==0)
            cout<<"\n The number is prime.";
    else
            cout<<"\n The number is not prime.";


    return 0;
}
```

Output-



```
Enter a number-11

 The number is prime.
Process returned 0 (0x0)   execution time : 10.861 s
Press ENTER to continue.
```



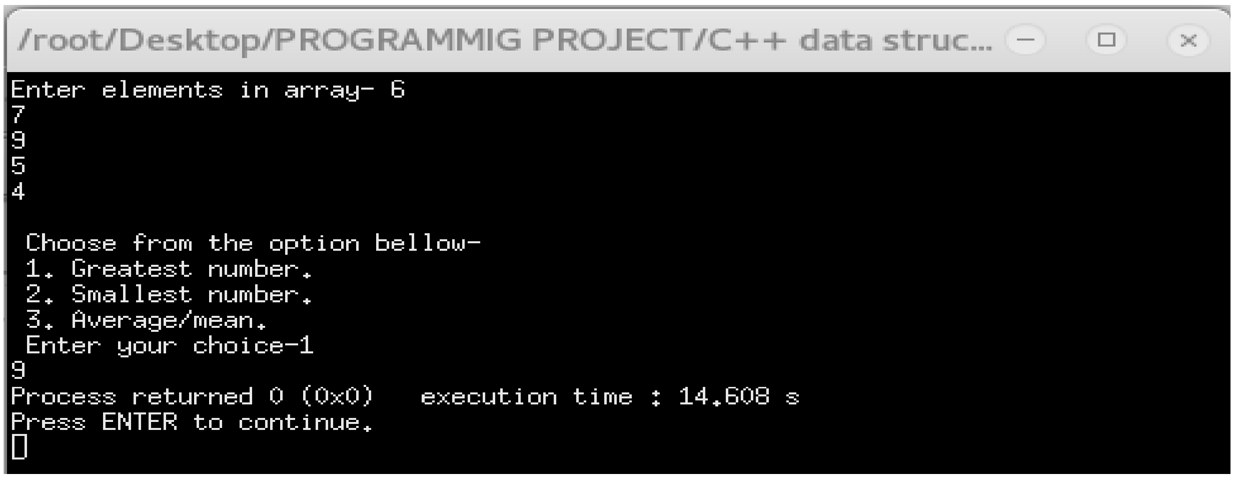
```
Enter a number-150

 The number is not prime.
Process returned 0 (0x0)   execution time : 9.825 s
Press ENTER to continue.
```

2.
Program-

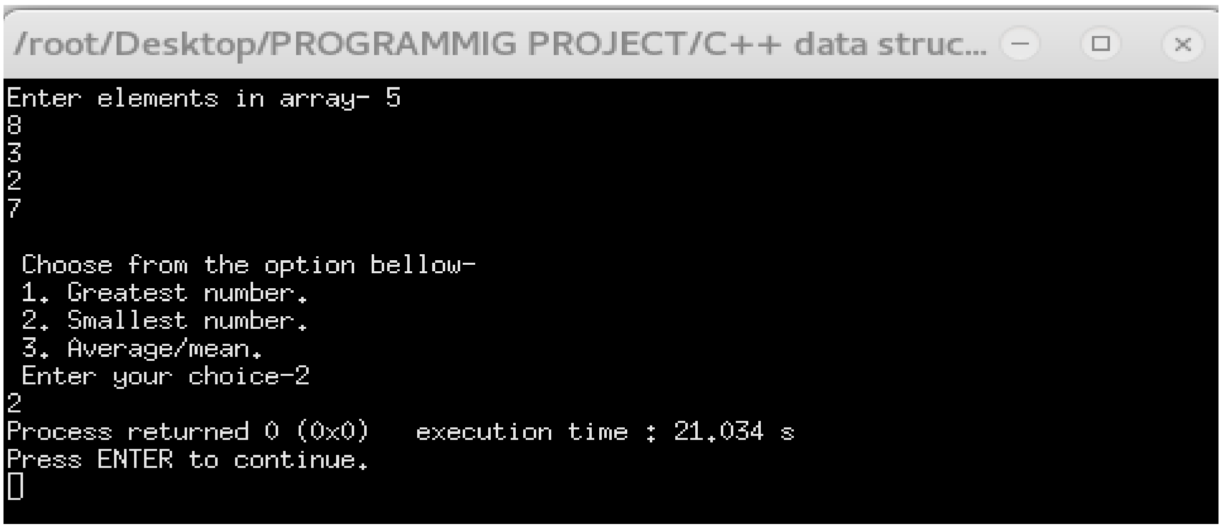
```
#include<iostream>
#include<string.h>
#include<stdio.h>

# define N 5

using namespace std;

int main()
{
      int a[N], choice ;
      cout<<"Enter elements in array- ";
      for( int i=0; i< N ; i++ )
            cin>> a[ i ] ;

      cout<<"\n Choose from the option bellow-";
      cout<<"\n 1. Greatest number.";
      cout<<"\n 2. Smallest number.";
      cout<<"\n 3. Average/mean.";
      cout<<"\n Enter your choice-";
       cin>> choice;

        switch(choice)
          {

           case 1 :      int max ;
                          for( int i=0; i< N ; i++ )
                          {
                                if ( max < a[i] )
                                {  max=a[ i ] ; }
                         }
                         cout<< max;
                         break;

           case 2 :      int min ;
                          for( int i=0; i< N ; i++ )
                           {
                             if ( min > a[i] )
                             {  min=a[ i ] ; }
                          }
                          cout<< min ;
                          break;

          case 3 :       float sum=0;
                          for( int i=0; i< N ; i++ )
                           {  sum=sum+a[i];   }

                           cout<<"Average/mean is -"<<sum/N;
                           break;


          default : cout<<"Wrong input";

      }

    return 0;
```

}

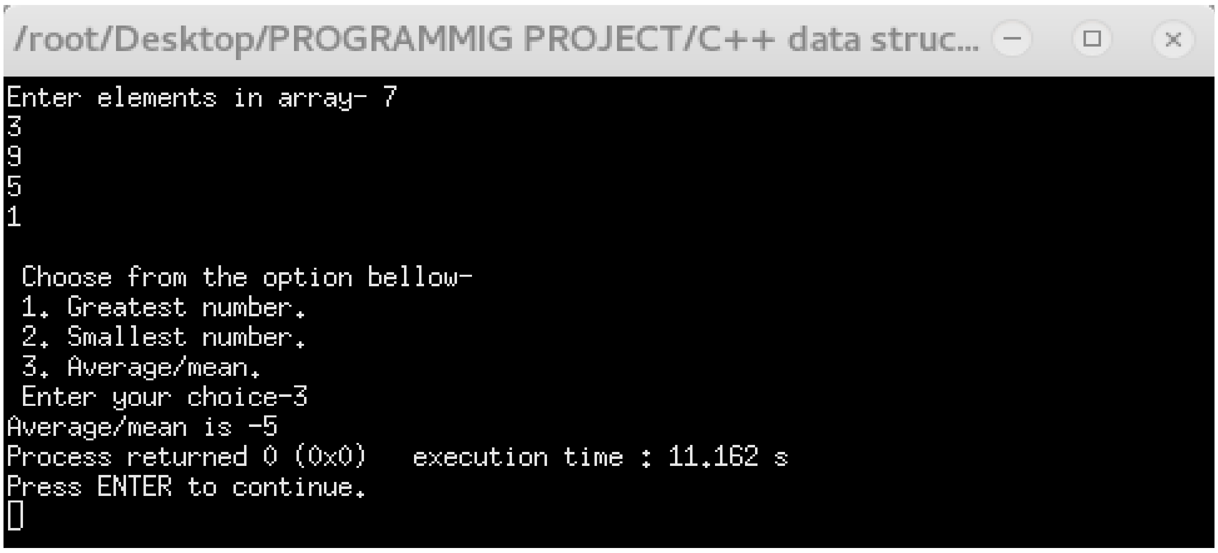
Output-



```
Enter elements in array- 6
7
9
5
4

 Choose from the option bellow-
 1. Greatest number.
 2. Smallest number.
 3. Average/mean.
 Enter your choice-1
9
Process returned 0 (0x0)   execution time : 14.608 s
Press ENTER to continue.
```



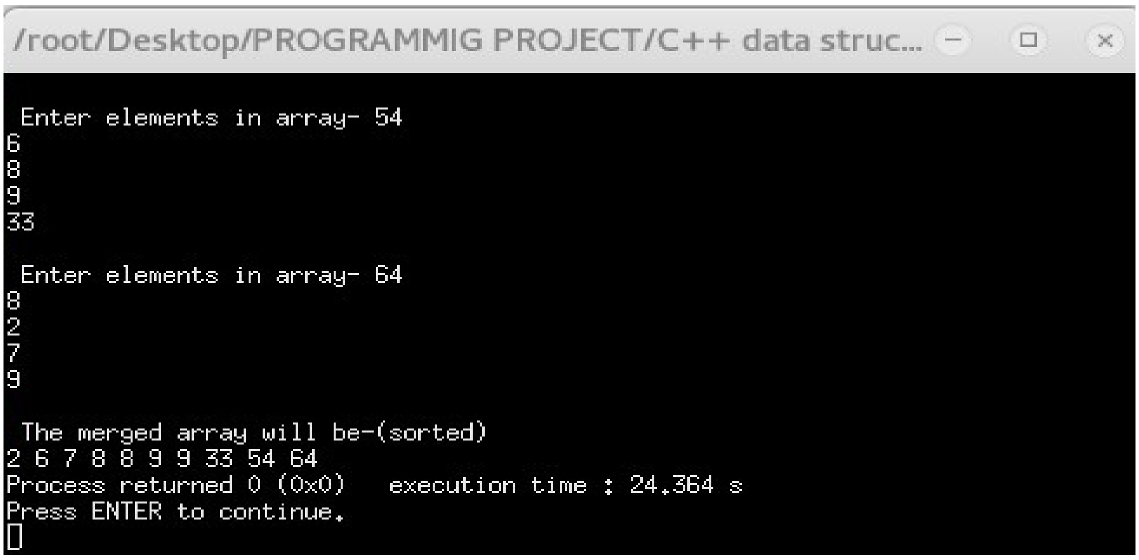
```
Enter elements in array- 5
8
3
2
7

 Choose from the option bellow-
 1. Greatest number.
 2. Smallest number.
 3. Average/mean.
 Enter your choice-2
2
Process returned 0 (0x0)   execution time : 21.034 s
Press ENTER to continue.
```



```
Enter elements in array- 7
3
9
5
1

 Choose from the option bellow-
 1. Greatest number.
 2. Smallest number.
 3. Average/mean.
 Enter your choice-3
Average/mean is -5
Process returned 0 (0x0)   execution time : 11.162 s
Press ENTER to continue.
```

3.
Program-
```
#include<iostream>
#include<stdio.h>

# define N 5

using namespace std;


void sort_array(int arr[])
{   int temp;
    for(int i=0; i<N-1; i++)
    {   for(int j=i+1; j<N; j++)
        {   if(arr[i]>arr[j])
            {   temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }
}

int main()
{
    int a[N], b[N],c[10], pos1=0,pos2=0,pos3=0 ;

    cout<<"\n Enter elements in array- ";
    for( int i=0; i< N ; i++ )
        cin>> a[ i ] ;

    sort_array(a);

    cout<<"\n Enter elements in array- ";
    for( int i=0; i< N ; i++ )
        cin>> b[ i ] ;

    sort_array(b);

      while(pos3!=N*2)
      {   if(a[pos1] < b[pos2])
        {   c[pos3] = a[pos1];
            pos1++;
            pos3++;
        }
        else
        {   c[pos3] = b[pos2];
            pos2++;
            pos3++;
        }

      }

      cout<<"\n The merged array will be-(sorted) \n";
     for( int i=0; i< 10 ; i++ )
        cout<< c[ i ] <<" ";

    return 0;
}
```

Output-



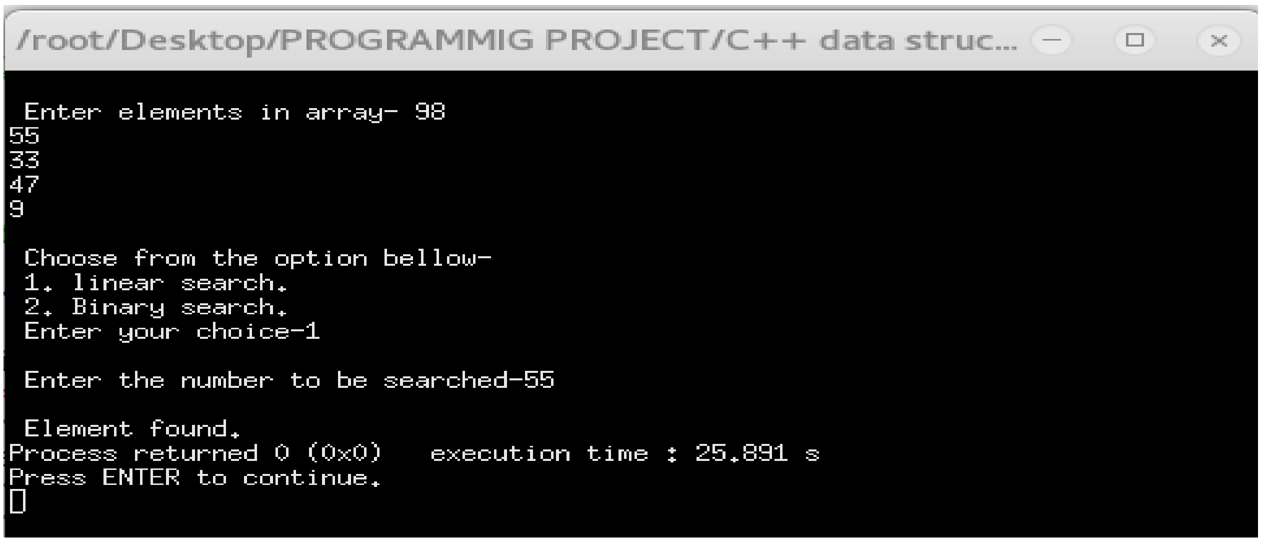
/root/Desktop/PROGRAMMIG PROJECT/C++ data struc...

```
 Enter elements in array- 54
6
8
9
33

 Enter elements in array- 64
8
2
7
9

 The merged array will be-(sorted)
2 6 7 8 8 9 9 33 54 64
Process returned 0 (0x0)   execution time : 24.364 s
Press ENTER to continue.
```

4.

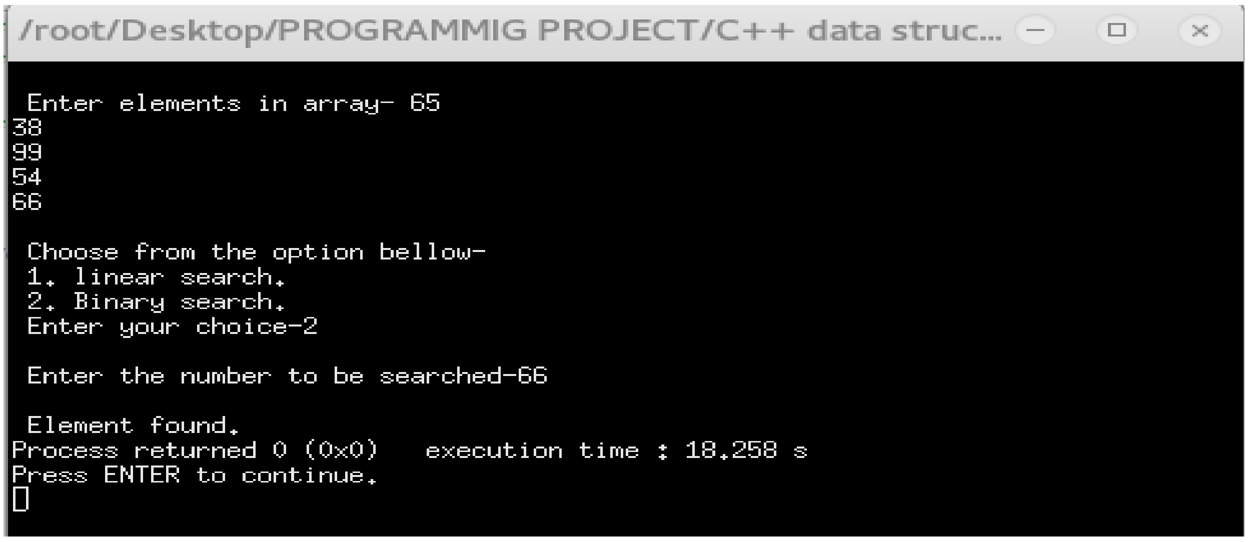
Program-
```
#include<iostream>
#include<stdio.h>

# define N 5

using namespace std;

void sort_array(int arr[])
{   int temp;
    for(int i=0; i<N-1; i++)
    {   for(int j=i+1; j<N; j++)
        {   if(arr[i]>arr[j])
            {   temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }
}


void linear(int arr[])
{   int elem, flag=0;
    cout<<"\n Enter the number to be searched-";
    cin>>elem;

    for(int i=0; i<N; i++)
    {   if(arr[i]== elem)
        {   flag= 1;
            break;
        }
    }

    if(flag== 1)
        cout<<"\n Element found. "; //at position "<<i+1;
    else
        cout<<"\n Element not found.";

}

void binary(int arr[])
{   int elem, flag=0, beg=0, last=N-1, mid=(beg+last)/2;
    cout<<"\n Enter the number to be searched-";
    cin>>elem;

    sort_array(arr);

    while(beg<=last)
    {   mid=(beg+last)/2;
        if(arr[mid]== elem)
        {   flag= 1;
            break;
        }
        else if(elem < arr[mid])
        {   last=mid-1;   }
        else
        {   beg=mid+1; }
    }

    if(flag== 1)
        cout<<"\n Element found. "; //at position "<<i+1;
    else
        cout<<"\n Element not found.";

}


int main()
{
    int a[N],choice;

      cout<<"\n Enter elements in array- ";
     for( int i=0; i< N ; i++ )
        cin>> a[ i ] ;


      cout<<"\n Choose from the option bellow-";
      cout<<"\n 1. linear search.";
      cout<<"\n 2. Binary search.";
      cout<<"\n Enter your choice-";
      cin>> choice;

    switch(choice)
    {   case 1:     linear(a);
                    break;

        case 2:     binary(a);
                    break;

        default:    cout<<"Wrong input.";

    }

    return 0;
}
```

Output-

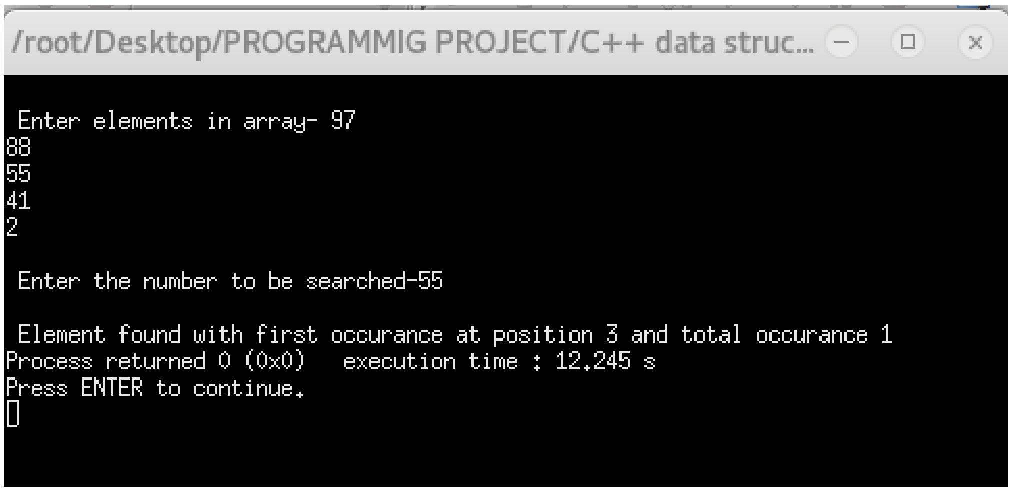


/root/Desktop/PROGRAMMIG PROJECT/C++ data struc...

```
 Enter elements in array- 98
55
33
47
9

 Choose from the option bellow-
 1. linear search.
 2. Binary search.
 Enter your choice-1

 Enter the number to be searched-55

 Element found.
Process returned 0 (0x0)   execution time : 25.891 s
Press ENTER to continue.
```

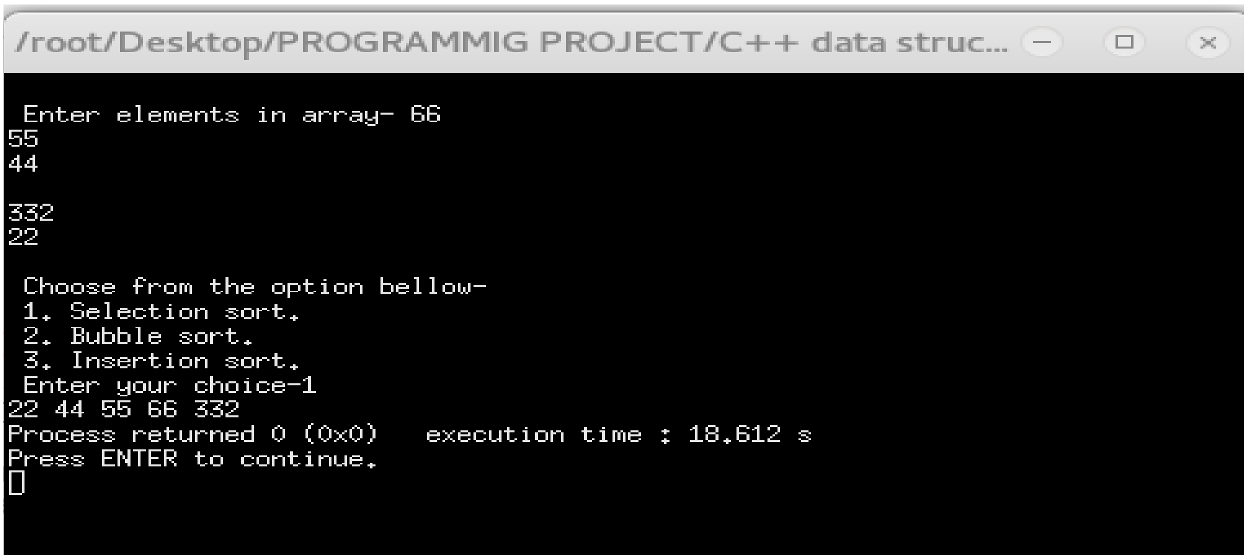


```
 Enter elements in array- 65
38
99
54
66

 Choose from the option bellow-
 1. linear search.
 2. Binary search.
 Enter your choice-2

 Enter the number to be searched-66

 Element found.
Process returned 0 (0x0)   execution time : 18.258 s
Press ENTER to continue.
```

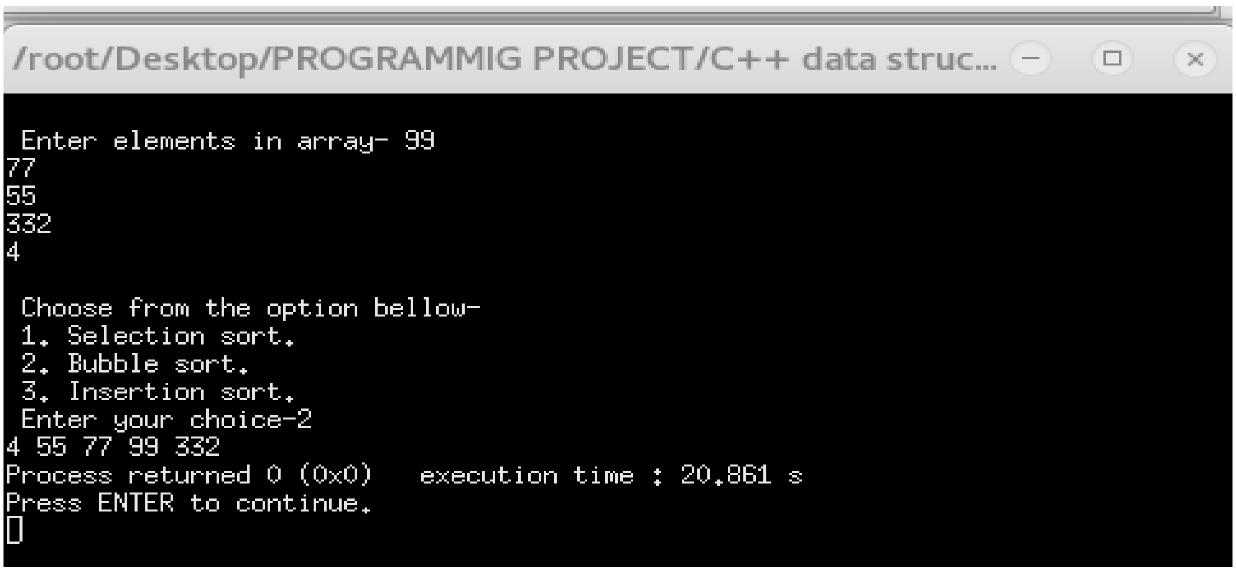
5.
Program-
```
#include<iostream>
#include<stdio.h>

# define N 5

using namespace std;


void linear(int arr[])
{   int elem, i, pos=0,count_elem=0 ;
    cout<<"\n Enter the number to be searched-";
    cin>>elem;

    for( i=0; i<N; i++)
    {   if(arr[i]== elem)
        {   if(count_elem == 0)
                {   pos = i+1; }
            count_elem++ ;
        }
    }

    if(count_elem !=  0)
        cout<<"\n Element found with first occurance at position "<<pos<<"
                 and total occurance "<<count_elem;
    else
        cout<<"\n Element not found.";
}
int main()
{
    int a[N],choice;

      cout<<"\n Enter elements in array- ";
    for( int i=0; i< N ; i++ )
        cin>> a[ i ] ;

    linear(a);
    return 0;
}
```
Output-


```
 Enter elements in array- 97
88
55
41
2

 Enter the number to be searched-55

 Element found with first occurance at position 3 and total occurance 1
Process returned 0 (0x0)   execution time : 12.245 s
Press ENTER to continue.
```
6.
Program-

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

# define N 5

using namespace std;

void selection_sort(int arr[])
{   int temp;
    for(int i=0; i<N-1; i++)
       { for(int j=i+1; j<N; j++)
           { if(arr[i]>arr[j])
                {temp = arr[i];
                 arr[i] = arr[j];
                 arr[j] = temp;
                }
           }
       }
}

void insertion_sort(int arr[])
{   int temp;

    for(int i=1; i<N; i++)
    {
        for(int j=0; j<i; j++)
        {   temp= arr[i];
            if(arr[i]<arr[j])
            {
                int k;
                for(k=i; k>j; k--)
                {    arr[k]=arr[k-1];
                }
                arr[k]=temp;
            }
        }
    }
}

void bubble_sort(int arr[])
{   int temp, flag=0;
    for(int i=0; i<N-1; i++)
    { flag= 0;
        for(int j=0; j<N-1; j++)
        {   if(arr[j]>arr[j+1])
            {    temp = arr[j];
                 arr[j] = arr[j+1];
                 arr[j+1] = temp;

                    flag=1;
            }
          if(flag == 1)
          {    break;   }
        }
    }
}

int main()
{
    int a[N],choice;

      cout<<"\n Enter elements in array- ";
       for( int i=0; i< N ; i++ )
         cin>> a[ i ] ;


      cout<<"\n Choose from the option bellow-";
      cout<<"\n 1. Selection sort.";
      cout<<"\n 2. Bubble sort.";
      cout<<"\n 3. Insertion sort.";
      cout<<"\n Enter your choice-";
       cin>> choice;

    switch(choice)
    {   case 1:     selection_sort(a);
                    break;

        case 2:     insertion_sort(a);
                    break;

          case 3:     bubble_sort(a);
                    break;

        default:    cout<<"Wrong input.";
                    break;

    }

    for( int i=0; i< N ; i++ )
        cout<< a[i]<<" " ;

    return 0;
}
```
Output-



```
 Enter elements in array- 66
55
44

332
22

 Choose from the option bellow-
 1. Selection sort.
 2. Bubble sort.
 3. Insertion sort.
 Enter your choice-1
22 44 55 66 332
Process returned 0 (0x0)   execution time : 18.612 s
Press ENTER to continue.
```

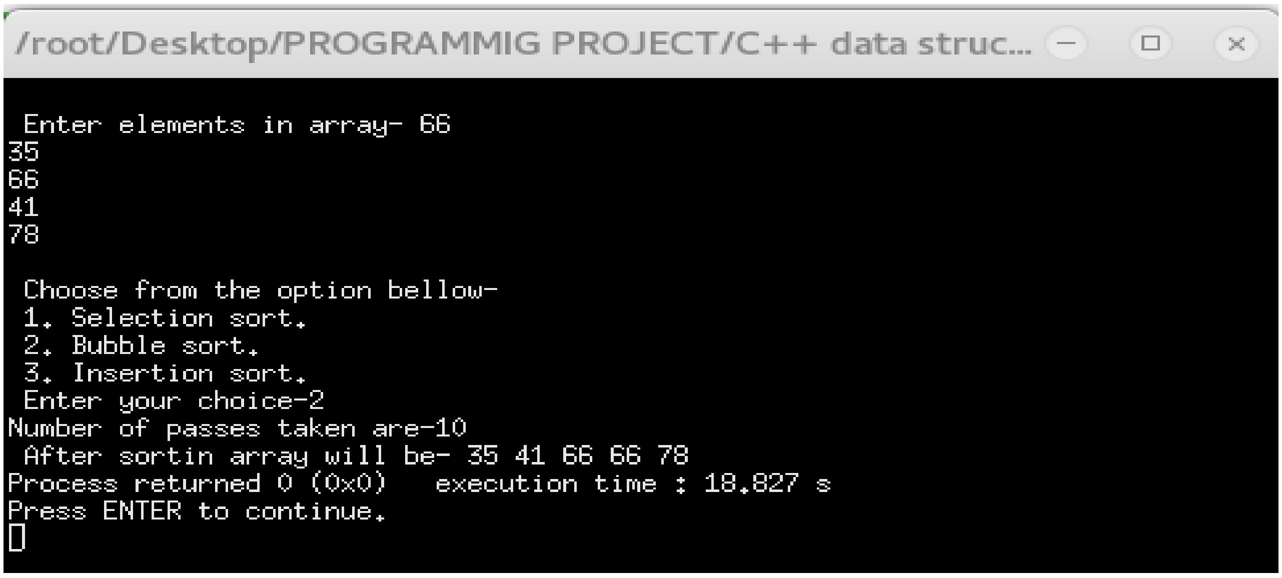

/root/Desktop/PROGRAMMIG PROJECT/C++ data struc...

```
 Enter elements in array- 99
77
55
332
4

 Choose from the option bellow-
 1. Selection sort.
 2. Bubble sort.
 3. Insertion sort.
 Enter your choice-2
4 55 77 99 332
Process returned 0 (0x0)   execution time : 20.861 s
Press ENTER to continue.
```

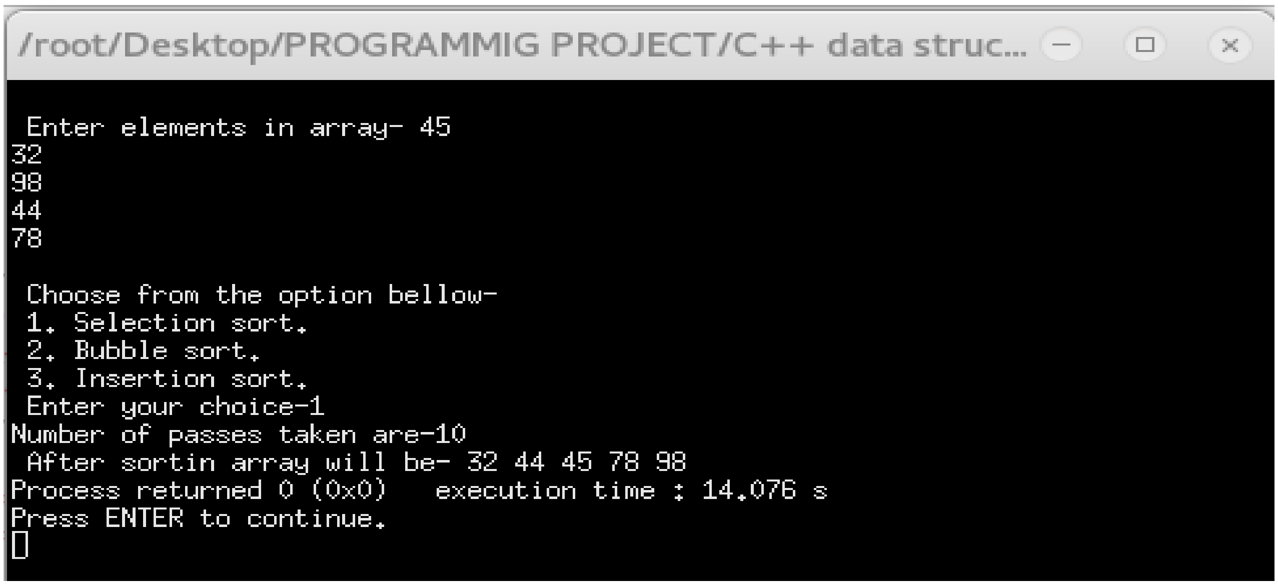
7.

Program-

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

# define N 5

using namespace std;

int selection_sort(int arr[])
{   int temp; int count=0;
    for(int i=0; i<N-1; i++)
       { for(int j=i+1; j<N; j++)
           { if(arr[i]>arr[j])
                {temp = arr[i];
                 arr[i] = arr[j];
                 arr[j] = temp;
                }
                count++;
           }
       }
       return count;
}

int insertion_sort(int arr[])
{   int temp; int count=0;

    for(int i=1; i<N; i++)
    {
        for(int j=0; j<i; j++)
        {   temp= arr[i];
            if(arr[i]<arr[j])
            {
                int k;
                for( k=i; k>j; k--)
                {    arr[k]=arr[k-1];
                }
                arr[k]=temp;
            }
            count++;
        }
    }
    return count;
}

int bubble_sort(int arr[])
{   int temp, flag=0; int count=0;
    for(int i=0; i<N-1; i++)
    { flag= 0;
        for(int j=0; j<N-1; j++)
        {   if(arr[j]>arr[j+1])
            {    temp = arr[j];
                 arr[j] = arr[j+1];
                 arr[j+1] = temp;

                    flag=1;
            }

            if(flag == 1)
            {    break;   }

            count++;
        }
    }
    return count;
}

int main()
{
    int a[N],choice;

      cout<<"\n Enter elements in array- ";
     for( int i=0; i< N ; i++ )
        cin>> a[ i ] ;


      cout<<"\n Choose from the option bellow-";
      cout<<"\n 1. Selection sort.";
      cout<<"\n 2. Bubble sort.";
      cout<<"\n 3. Insertion sort.";
      cout<<"\n Enter your choice-";
     cin>> choice;

    switch(choice)
    {   case 1:     cout<<"Number of passes taken are-"<<selection_sort(a);
                    break;

        case 2:     cout<<"Number of passes taken are-"<<insertion_sort(a);
                    break;

        case 3:     cout<<"Number of passes taken are-"<<bubble_sort(a);
                    break;

        default:    cout<<"Wrong input.";
                    break;

    }

    cout<<"\n After sortin array will be- ";
    for( int i=0; i< N ; i++ )
        cout<< a[i]<<" " ;

    return 0;
}
```
Output-



```
/root/Desktop/PROGRAMMIG PROJECT/C++ data struc...

 Enter elements in array- 66
35
66
41
78

 Choose from the option bellow-
 1. Selection sort.
 2. Bubble sort.
 3. Insertion sort.
 Enter your choice-2
Number of passes taken are-10
 After sortin array will be- 35 41 66 66 78
Process returned 0 (0x0)   execution time : 18.827 s
Press ENTER to continue.
```



```
/root/Desktop/PROGRAMMIG PROJECT/C++ data struc...

 Enter elements in array- 45
32
98
44
78

 Choose from the option bellow-
 1. Selection sort.
 2. Bubble sort.
 3. Insertion sort.
 Enter your choice-1
Number of passes taken are-10
 After sortin array will be- 32 44 45 78 98
Process returned 0 (0x0)   execution time : 14.076 s
Press ENTER to continue.
```

8.
Program-

```
#include<iostream>
#define N 5
using namespace std;

class LinkedListed_Queue{

    private:    struct node
                {
                    int data ;
                    node *link ;
                };

                node * start ;


    public:     LinkedListed_Queue()
                {   start = NULL ; }

                int is_emptyQueue()
                {
                    if( start == NULL )
                        {   return 1 ;   }
                    else
                        {   return 0 ;   }
                }

                void insert_last(int num)
                {   node *temp = new node ;

                    temp-> data = num ;
                    temp->link = NULL ;

                    if(start == NULL)
                    {   start = temp ;}
                    else
                    {   node *t= start ;

                        while( t->link != NULL)
                        {   t = t->link ;    }

                        t->link = temp ;
                    }
                }

                int extract_begning()
                {   node * temp = start ;
                    int val;

                    val= start->data ;
                    start= start->link ;
                    delete temp;

                    return val;
                }
};

LinkedListed_Queue LLQ[10];

void redix_sort(int arr[])
{   int max_value= arr[0], max_digit= 0, greater_num= 1, divident= 1 ;

    for(int i=1; i<N; i++)
    {
         if(max_value < arr[i] )
         {   max_value = arr[i] ;    }
    }

    while( max_value > greater_num)
    {   max_digit++ ;
        greater_num = greater_num*10 ;
    }

    for(int k=1; k<= max_digit; k++)
    {
         for(int i=0; i<N; i++)
        {
           LLQ[(arr[i]/divident)%10].insert_last(arr[i]);
        }

        divident = divident*10 ;

        for(int i=0,j=0; i<N; )
        {
            if(LLQ[j].is_emptyQueue())
            {   j++; }
            else
            {   arr[i]=LLQ[j].extract_begning();
                i++;
            }

        }

    }

}


int main()
{   int a[N];
    cout<<"Enter the elements-";
    for(int i=0; i<N; i++)
        cin>>a[i];

    redix_sort(a);

    cout<<"After sorting the elements are-";

    for(int i=0; i<N; i++)
        cout<<a[i]<<" ";

    return(0);
}
```

Output-

9.
Program-

Output-

10.
Program-

Output-

11.
Program-

Output-

12.
Program-

```
#include <iostream>

# define N 5
using namespace std;


 class STACK
   {  private : int ar[N], top;

      public: STACK()
               { top = -1;}

              void PUSH(int n)
               { if( top== N-1)
                  {cout<<"The stack is full\n";}
                 else
                  { top++;
                    ar[top]= n;
                  }
               }

               int POP()
               { if( top == -1)
                  {cout<<"The stack is empty\n";}
                 else
                  { cout<<"The element deleted is-"<<ar[top];
                    top--;
                  }
               }

               void Display()
               {  cout<<"The stack from top bottom is-  ";
                  for(int i= top; i>-1 ; i--)
                   {cout<<ar[i]<<" ";}
               }

   };
int main()
{ int n;
  STACK S;
  while(1)
   {
      cout<<"Choose from the option bellow-\n";
      cout<<"1.PUSH\n";
      cout<<"2.POP\n";
      cout<<"3.Display element-\n";
      cout<<"4.exit\n";
      cout<<"Enter the option number bellow-";
      cin>>n;

      switch(n)
      {  case 1 : int num;
                  cout<<"Enter the element to be inserted(pushed)-";
                  cin>> num;
                  S.PUSH( num );
                  break;

         case 2 : S.POP();
```

```
                    break;

         case 3 : S.Display();
                  break;

         case 4 : exit(0);
                  break;

         default : cout<<"Enter a correct option";
                   break;

     }


   }
 return 0;
}
```
Output-

13.

Program-

```
#include<iostream>
#define N 5

using namespace std;

class Queue{

      private : int FRONT, REAR ;
                int ar[N] ;

      public : Queue()
               {
                    FRONT=0;
                    REAR=-1;
               }

               void insert_element( int num)
               {
                   if(REAR == N-1)
                   {
                        cout<<"\n The queue is Full";
                   }
                   else
                   {
                        REAR++ ;
                        ar[REAR]= num;
                   }
               }

               void delete_element()
               {
                   if(REAR<FRONT)//REAR == -1 || FRONT== REAR+1)
                   {
                        cout<<"\n The queue is Empty";
                   }
                   else
                   {
                        cout<<"\n Element deleted is-"<<ar[FRONT];
                        ar[FRONT] = NULL;
                        FRONT++;
                   }

               }


                void display()
                {
                    if(REAR == -1 || FRONT== REAR+1)
                    {
                        cout<<"\n The queue is Empty";
                    }

                    else
                    {
                        for(int i= FRONT; i<=REAR; i++)
                        {
                            cout<<ar[i]<<" ";
                        }
                    }

                }

};

Queue q;

int main()
{
    int choice;

    cout<<"\n Select a operation to be performed on Queue-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;

    switch(choice)
    {
        case 1:     int n;
                    cout<<"\n Enter a number-";
                    cin>>n;
                    q.insert_element(n);
                    break;

        case 2:     q.delete_element();
                    break;

        case 3:     cout<<"\n The element in the queue are as-";
                    q.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                    wrong input.";
                    break;

    }

    main();

    return 0;
}
```
Output-

14.

Program-

```
#include<iostream>

using namespace std;

class CircularLinkedList{

    private:    struct node
                {
                    int data ;
                    node *link ;
                };

                node * start ;
                int count_elem ;


    public:     CircularLinkedList()
                {   start = NULL ;
                    count_elem = 0 ;
                }

                void add_node()
                {   node *temp = new node ;

                    cout<<"Enter the element-" ;
                    cin>>temp->data ;

                    if(start == NULL) //  first positon when list is empty.
                    {
                        start = temp ;
                        temp->link = start ;
                    }

                    else
                    {   int position;
                        cout<<"Enter the position where you want to insert
                         data-";
                        cin>>position;

                        if( position < 1 || position > count_elem+1)
                        {   cout<<"No such position exixt in list till
                            now.";
                        }
                        else if( position == 1)
                        {   temp->link = start ;
                            start = temp ;
                        }

                        else if( position == count_elem+1 )
                        {   node *t = start ;
                            temp->link = start ;
                            for( int i=1; i != count_elem; i++ )
                            {   t = t->link ;    }

                            t->link = temp ;
                        }
```

```
        else //  middle positon.
        {   node *t = start ;

            for( int i=1; i != position-1 ; i++ )
            {   t = t->link ;   }

            temp->link = t->link ;
            t->link=temp ;
        }
    }

    count_elem++ ;
}


void delete_node()
{   int num , pos=0, found=0 ;

    cout<<"Enter the data you want to delete-";
    cin>> num ;

    node *t = start ;
    if( start == NULL)
    {   cout<<"List is empty" ;  }

    else
    {
        for( int i=1; i != count_elem; i++ )
        {   pos++;
            if( t->data == num )
            {   found = 1 ;
                break ;
            }
            else
            {   t = t->link ;  }
        }

        if(found == 0)
        {   cout<<"data dosen't exist in record. ";    }

        else
        {
            if( pos == 1 )    // delete first
            {   node * t ;
                t = start ;
                start = start->link ;
            }
            else if( pos == count_elem )    // delete last
            {
                node *t = start ;
                node * previous ;

                for( int i=1; i != count_elem; i++ )
                {   previous = t;
                    t = t->link ;
                }

                previous->link = start ;
            }
```

```
                            else    // delete mid
                            {
                                node *t = start ;
                                node * previous ;

                                for( int i=1; i != count_elem; i++ )
                                {   if( t->data == num)
                                    {   break ; }
                                    else
                                    {   previous = t ;
                                        t = t->link ;
                                    }
                                }

                                previous->link = t->link ;
                            }

                        }
                    }

                    delete t ;
                    count_elem-- ;
                }


                void display()
                {   node *t = start ;

                    for( int i=1; i <= count_elem ; i++ )
                    {   cout<<t->data<<" ";
                        t = t->link ;
                    }
                }

                ~CircularLinkedList()
                {   node *t = start ;
                    node *temp;

                    for( int i=1; i != count_elem ; i++ )
                    {   temp=t;
                        t = t->link ;
                        delete temp;
                    }
                }
};


class CircularLinkedList CL;

int main()
{   int choice;

    cout<<"\n Select a operation to be performed on List-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;
    switch(choice)
    {
        case 1:     CL.add_node();
                    break;

        case 2:     CL.delete_node();
                    break;

        case 3:     cout<<"\n The element in the List are as-";
                    CL.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                    wrong input.";
                    break;

    }

    main();


    return(0);
}
```

Output-

15.
Program-

```
#include<iostream>

#define N 5

using namespace std;

class PQueue
{
    private:
            struct elements
            {
                int data;
                int priority;
            };

            struct elements e[N], temp;
            int FRONT, REAR;

            void sort_element()
            {
                for(int i= FRONT ; i<REAR ; i++)
                {   for(int j=i+1 ; j<=REAR; j++)
                    {
                        if(e[i].priority > e[j].priority)
                        {
                            temp = e[i];
                            e[i] = e[j];
                            e[j] = temp;
                        }

                    }
                }

            }

    public:
            PQueue()
            { FRONT = 0 ; REAR = -1 ;}

            void insert_element()
            {
                REAR++ ;

                if(REAR == N-1)
                {
                    cout<<"\n The queue is Full";
                }
                else
                {
                    cout<<"Enter element data-";
                    cin>> e[REAR].data;
                    cout<<"Ente its priority-";
                    cin>> e[REAR].priority;
                }

                sort_element();
            }

            void delete_element()
            {
                if(REAR == -1 || FRONT== REAR+1)
                {
                    cout<<"\n The queue is Empty";
                }
                else
                {
                    e[FRONT].data = NULL;
                    e[FRONT].priority = NULL;
                }

                FRONT++;
            }

            void display()
            {
                if(REAR == -1 || FRONT== REAR+1)
                {
                    cout<<"\n The queue is Empty";
                }
                else if(REAR == N-1)
                {
                    cout<<"\n The queue is Full";
                }
                else
                {
                    for(int i= FRONT; i<=REAR; i++)
                    {
                        cout<<e[i].data<<" ";
                    }
                }
            }

};



PQueue q;

int main()
{
    int choice;

    cout<<"\n Select a operation to be performed on Queue-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;

    switch(choice)
    {
        case 1:     q.insert_element();
                    break;

        case 2:     q.delete_element();
                    break;
        case 3:     cout<<"\n The element in the queue are as-";
                    q.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                    wrong input.";
                    break;

    }

    main();

    return 0;
}
```

Output-

Program-

```
#include<iostream>
#define N 5

using namespace std;

class IRDQueue{

    private : int FRONT, REAR ;
              int ar[N] ;

    public :   IRDQueue()
               {
                    FRONT=0;
                    REAR=-1;
               }

               void insert_element( int num)
               {
                   if(REAR == N-1)
                   {
                        cout<<"\n The queue is Full.";
                   }
                   else
                   {
                        REAR++ ;
                        ar[REAR]= num;
                   }
               }

               void delete_element_REAR( )
               {
                   if(REAR == -1)
                   {
                        cout<<"\n The queue is Empty at Rear end.";
                   }
                   else
                   {
                        cout<<"\n Element deleted is-"<<ar[REAR];
                        ar[REAR] = NULL;
                        REAR-- ;
                   }
               }

               void delete_element_FRONT()
               {
                   if( FRONT == N-1 || FRONT > REAR )
                   {
                        cout<<"\n The queue is Empty at Front end.";
                   }
                   else
                   {
                        cout<<"\n Element deleted is-"<<ar[FRONT];
                        ar[FRONT] = NULL;
                        FRONT++;
                   }

               }
```

```
                void display()
                {
                    if(REAR == -1 || FRONT== REAR+1)
                    {
                        cout<<"\n The queue is Empty ";
                    }

                    else
                    {
                        for(int i= FRONT; i<=REAR; i++)
                        {
                            cout<<ar[i]<<" ";
                        }
                    }

                }

};

IRDQueue q;

int main()
{   int choice;

    cout<<"\n Select a operation to be performed on Queue-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;

    switch(choice)
    {
        case 1:     int n;
                    cout<<"\n Enter a number-";
                    cin>>n;
                    q.insert_element(n);
                    break;

        case 2:     int opt;
                    cout<<"\n Choose at which end you want to element-";
                    cout<<"\n 1. Front";
                    cout<<"\n 2. Rear";
                    cout<<"\n Enter your choice-";
                    cin>>opt;

                    if( opt== 1)
                    {q.delete_element_FRONT();}
                    else if( opt== 2)
                    {q.delete_element_REAR();}
                    else
                    {cout<<"\n Wrong input";}
                    break;



        case 3:     cout<<"\n The element in the queue are as-";
```

```
                    q.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                    wrong input.";
                    break;

    }

    main();


    return(0);
}
```
Output-

Program-
Output-

18.

Program-
Output-

19.

Program-

```
#include<iostream>

using namespace std;

class LinkedList{

    private:    struct node
                {
                    int data ;
                    node *link ;
                };

                node * start ;



    public:     LinkedList()
                {   start = NULL ; }

                void add_begning()
                {   node *temp = new node ;

                    cout<<"Enter the element-" ;
                    cin>>temp-> data ;

                    if(start== NULL)
                    {   temp->link=NULL;
                        start = temp ;
                    }

                    else
                    {   temp->link = start ;
                        start = temp ;
                    }
                }

                void add_last()
                {   node *temp = new node ;

                    cout<<"Enter the element-" ;
                    cin>>temp-> data ;
                    temp->link =NULL ;

                    node *t= start ;

                    while( t->link != NULL)
                    {   t = t->link ;    }
                    t->link = temp ;
                }

                void add_mid(int num)
                {   node * temp = new node;

                    cout<<"Enter the element-" ;
                    cin>>temp-> data ;
                    node * t = start ;
                    while(t!=NULL)
                    {   if(t->data == num)
                        {   break ;  }

                        t = t->link ;
                    }

                    temp->link = t->link ;
                    t->link=temp ;

                }

                void delete_begning()
                {   node * temp ;
                    temp=start ;
                    start= start->link ;
                    delete temp;
                }

                void delete_last()
                {   node *t= start ;
                    node * previous ;

                    while( t->link != NULL)
                    {   previous = t ;
                        t = t->link ;
                    }

                    previous->link = NULL ;
                    delete t ;
                }

                void delete_mid(int num)
                {   node *t = start ;
                    node * previous ;

                    while( t != NULL)
                    {   if( t->data == num)
                        { break ; }
                        else
                        {previous = t ;
                         t = t->link ;
                        }
                    }

                    previous->link = t->link ;

                    delete t ;
                }

                void display()
                {   node *t= start ;
                    while( t != NULL)
                    {   cout<<t->data<<" " ;
                        t = t->link ;
                    }

                }
```

```
                ~LinkedList()
                {   node *t= start ;
                    node *temp;

                    while( t != NULL)
                    {   temp=t;
                        delete temp;
                    }
                }
};


class LinkedList L;

int main()
{   int choice;

    cout<<"\n Select a operation to be performed on List-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;

    switch(choice)
    {
        case 1:     int n, opt;

                    cout<<"\n Enter where you want to add element- ";
                    cout<<"\n 1. Begning";
                    cout<<"\n 2. Middle.";
                    cout<<"\n 3.End.";
                    cout<<"\n enter your choice-";
                    cin>>opt;
                    switch(opt)
                    {   case 1:     L.add_begning();
                                    break;

                        case 2:     cout<<"\n Enter, after which element you
                                    want to insert new element-";
                                    cin>>opt;

                                    L.add_mid(opt);
                                    break;

                        case 3:     L.add_last();
                                    break;

                        default:    cout<<"\n Wrong choice";
                    }

                    break;

        case 2:     int  opt2;

                    cout<<"\nEnter from where you want to delete element- ";
                    cout<<"\n 1. Begning";
                    cout<<"\n 2. Middle.";
                    cout<<"\n 3.End.";
                    cout<<"\n enter your choice-";
                    cin>>opt;
                    switch(opt)
                    {   case 1:     L.delete_begning();
                                    break;

                        case 2:     cout<<"\n Enter, element you want to
                                   delete-";
                                    cin>>opt;

                                    L.delete_mid(opt);
                                    break;

                        case 3:     L.delete_last();
                                    break;

                        default:    cout<<"\n Wrong choice";
                    }

                    break;

        case 3:     cout<<"\n The element in the List are as-";
                    L.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                   wrong input.";
                    break;

    }

    main();


    return(0);
}
```

Output-

Program-
Output-

Program-

```
#include<iostream>

using namespace std;

class LinkList
{
    public:    struct node
                {
                int data;
                node *link;
                };

                node *start;


                LinkList()
                {
                  start=NULL;
                }

                void add_beg()
                {   node *temp = new node;

                    cout<<"Enter element -";
                    cin>> temp->data;

                    if( start== NULL)
                    {   start = temp;  }
                    else
                    {   temp->link=start;
                        start=temp;
                    }

                }

                void delete_beg()
                {
                    node *temp = new node;

                    if(start == NULL)
                    {   cout<<"Stack is empty";}
                    else
                    {
                    temp = start;
                    start= start->link ;

                    cout<<"Element deleted is-"<<temp->data;
                    delete temp;
                    }
                }

                void travers()
                {   node *temp = new node;
                    temp = start;
                    while(temp != NULL)
                        {
                            cout<<temp->data<<" ";
                            temp=temp->link;
                        }
                }



};

class STACK : private LinkList
{
    public:   STACK()
              { }

              void PUSH()
              { add_beg();}

              void POP()
              { delete_beg();}

              void display()
              { travers();}
};


STACK S;

int main()
{   int choice;

    cout<<"\n Select a operation to be performed on List-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;

    switch(choice)
    {
        case 1:     S.PUSH();
                    break;

        case 2:     S.POP();
                    break;

        case 3:     cout<<"\n The element in the List are as-";
                    S.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                    wrong input.";
                    break;

    }

    main();
    return(0);
}
```

Output-

Program-

```
#include<iostream>

using namespace std;

class QUEUE_LL          // lisked list for Queue.
{   public :
                struct node
                {   int data;
                    node * link;
                };

                node * FRONT;
                node * REAR;

                node*start = NULL;

                QUEUE_LL()
                {   FRONT = NULL;
                    REAR = NULL;
                }

                void add_last()
                {   node *temp = new node;

                    cout<<"Enter a number-";
                    cin>>temp->data;

                    if(start == NULL)
                    {   temp->link = NULL;
                        start = temp;

                        REAR = temp;
                        FRONT = temp;
                    }

                    else
                    {   temp->link =NULL;
                        REAR->link = temp;

                        REAR = temp;
                    }
                }

                void delete_beg()
                {   if(FRONT == NULL || REAR == NULL )
                    {cout<<"\n The queue is empty.";}

                    else
                    {   node *t = FRONT;
                        cout<<"\n The element deleted was- "<<FRONT->data;
                        FRONT = FRONT->link;

                        delete t;
                    }
                }

                void display()
                {   node *t = new node;
```

```
                    t = FRONT;

                    while( t != NULL)
                    {   cout<< t->data;
                        t = t->link;
                    }

                }

                ~QUEUE_LL()
                {   node *t = new node;

                    while( FRONT != NULL)
                    {
                        t = FRONT;
                        FRONT = FRONT->link;
                        delete t;
                    }

                }

};

QUEUE_LL q;

int main()
{   int choice;

    cout<<"\n Select a operation to be performed on Queue-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;

    switch(choice)
    {
        case 1:     q.add_last();
                    break;

        case 2:     q.delete_beg();
                    break;

        case 3:     cout<<"\n The element in the List are as-";
                    q.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly entered
                    wrong input.";
                    break;

    }

    main();
    return(0);
}
```
Output-

23.

Program-
```
#include<iostream>

using namespace std;

class DoublyLinkedList{

    private:    struct node
                {
                    int data;
                    node *next;
                    node *previous;
                };

                node * start;



    public:     DoublyLinkedList()
                {   start = NULL; }

                void add_begning()
                {   node *temp = new node;

                    cout<<"Enter the element-";
                    cin>>temp-> data;

                    temp->previous=NULL;

                    if(start == NULL)
                    {   temp->next = NULL;
                        start = temp ;
                    }

                    else
                    {   temp->next = start;
                        start = temp;
                    }
                }

                void add_last()
                {   node *temp = new node;

                    cout<<"Enter the element-";
                    cin>>temp -> data;

                    temp->next = NULL;

                    node *t = start;

                    while( t->next != NULL)
                    {   t = t->next;    }

                    temp->previous = t;
                    t->next = temp;

                }

                void add_mid(int num)
```

```
{
    node *temp = new node;

    cout<<"Enter the element-";
    cin>>temp->data;

    node *t = start;
    while(t != NULL)
    {   if(t->data == num)
        {   break;  }
        else
        {   t = t->next;    }
    }

    temp->next = t->next;
    temp->previous = t->next->previous;
    t->next->previous = temp;
    t->next=temp;
}

void delete_begning()
{   node *temp;
    temp = start;
    start = start->next;
    delete temp;
}

void delete_last()
{   node *t = start;

    while( t->next != NULL)
    {
        t = t->next;
    }

    t->previous->next = NULL;
    delete t;
}

void delete_mid(int num)
{   node *t = start;
    node * temp;

    while( t!= NULL)
    {   if( t->data == num)
        {   break;  }
        else
        {
            t = t->next;
        }
    }

    temp = t;
    t->next->previous = t->previous;
    t->previous->next = t->next;

    delete temp;
}
```

```
                void display()
                {   node *t= start;
                    while( t != NULL)
                    {   cout<<t->data<<" ";
                        t = t->next;
                    }
                }

                ~DoublyLinkedList()
                {   node *t= start;
                    node *temp;

                    while( t != NULL)
                    {   temp = t;
                        t = t->next;

                        delete temp;
                    }

                }


};


class DoublyLinkedList DL;

int main()
{   int choice;

    cout<<"\n Select a operation to be performed on List-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;

    switch(choice)
    {
        case 1:     int n, opt;

                    cout<<"\n Enter where you want to add element- ";
                    cout<<"\n 1. Begning";
                    cout<<"\n 2. Middle.";
                    cout<<"\n 3.End.";
                    cout<<"\n enter your choice-";
                    cin>>opt;
                    switch(opt)
                    {   case 1:     DL.add_begning();
                                    break;

                        case 2:     cout<<"\n Enter after which element you
                                   want to insert new element-";
                                    cin>>opt;

                                    DL.add_mid(opt);
                                    break;
                        case 3:     DL.add_last();
                                    break;

                        default:    cout<<"\n Wrong choice";
                    }

                    break;

        case 2:     int  opt2;

                    cout<<"\nEnter from where you want to delete element- ";
                    cout<<"\n 1. Begning";
                    cout<<"\n 2. Middle.";
                    cout<<"\n 3.End.";
                    cout<<"\n enter your choice-";
                    cin>>opt;
                    switch(opt)
                    {   case 1:     DL.delete_begning();
                                    break;

                        case 2:     cout<<"\n Enter element you want to
                                   insert new element-";
                                    cin>>opt;

                                    DL.delete_mid(opt);
                                    break;

                        case 3:     DL.delete_last();
                                    break;

                        default:    cout<<"\n Wrong choice";
                    }

                    break;

        case 3:     cout<<"\n The element in the List are as-";
                    DL.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                   wrong input.";
                    break;

    }

    main();


    return(0);
}
```

Output-

24.

Program-
```
#include<iostream>

using namespace std;

class CircularLinkedList{

    private:    struct node
                {
                    int data ;
                    node *link ;
                };

                node * start ;
                int count_elem ;


    public:     CircularLinkedList()
                {   start = NULL ;
                    count_elem = 0 ;
                }

                void add_node()
                {   node *temp = new node ;

                    cout<<"Enter the element-" ;
                    cin>>temp->data ;

                    if(start == NULL) //  first positon when list is empty.
                    {
                        start = temp ;
                        temp->link = start ;
                    }

                    else
                    {   int position;
                        cout<<"Enter the position where you want to insert
                         data-";
                        cin>>position;

                        if( position < 1 || position > count_elem+1)
                        {   cout<<"No such position exixt in list till
                            now.";
                        }
                        else if( position == 1)
                        {   temp->link = start ;
                            start = temp ;
                        }

                        else if( position == count_elem+1 )
                        {   node *t = start ;
                            temp->link = start ;
                            for( int i=1; i != count_elem; i++ )
                            {   t = t->link ;    }

                            t->link = temp ;
                        }

                        else //  middle positon.
```

```
            {   node *t = start ;

                for( int i=1; i != position-1 ; i++ )
                {   t = t->link ;   }

                temp->link = t->link ;
                t->link=temp ;
            }
        }

        count_elem++ ;
    }


    void delete_node()
    {   int num , pos=0, found=0 ;

        cout<<"Enter the data you want to delete-";
        cin>> num ;

        node *t = start ;
        if( start == NULL)
        {   cout<<"List is empty" ;  }

        else
        {
            for( int i=1; i != count_elem; i++ )
            {   pos++;
                if( t->data == num )
                {   found = 1 ;
                    break ;
                }
                else
                {   t = t->link ;  }
            }

            if(found == 0)
            {   cout<<"data dosen't exist in record. ";    }

            else
            {
                if( pos == 1 )    // delete first
                {   node * t ;
                    t = start ;
                    start = start->link ;
                }
                else if( pos == count_elem )    // delete last
                {
                    node *t = start ;
                    node * previous ;

                    for( int i=1; i != count_elem; i++ )
                    {   previous = t;
                        t = t->link ;
                    }

                    previous->link = start ;
                }
                else    // delete mid
```

```
                            {
                                node *t = start ;
                                node * previous ;

                                for( int i=1; i != count_elem; i++ )
                                {   if( t->data == num)
                                    {   break ; }
                                    else
                                    {   previous = t ;
                                        t = t->link ;
                                    }
                                }

                                previous->link = t->link ;
                            }

                        }
                    }

                    delete t ;
                    count_elem-- ;
                }


                void display()
                {   node *t = start ;

                    for( int i=1; i <= count_elem ; i++ )
                    {   cout<<t->data<<" ";
                        t = t->link ;
                    }
                }

                ~CircularLinkedList()
                {   node *t = start ;
                    node *temp;

                    for( int i=1; i != count_elem ; i++ )
                    {   temp=t;
                        t = t->link ;
                        delete temp;
                    }
                }
};


class CircularLinkedList CL;

int main()
{   int choice;

    cout<<"\n Select a operation to be performed on List-";
    cout<<"\n 1.Insert element.";
    cout<<"\n 2.delete element.";
    cout<<"\n 3.display all element.";
    cout<<"\n 4.exit";
    cout<<"\n enter your choice here-";
    cin>>choice;
    switch(choice)
    {
        case 1:     CL.add_node();
                    break;

        case 2:     CL.delete_node();
                    break;

        case 3:     cout<<"\n The element in the List are as-";
                    CL.display();
                    break;

        case 4:     exit(0);
                    break;

        default:    cout<<"\n Ooops, it seems that you have possibly intered
                    wrong input.";
                    break;

    }

    main();


    return(0);
}
```

Output-

Program-
Output-