```
/******************************************************************************/
/************************* Name:- Divesh Uttamchandani  ***************************/
/************************* Class :- XII  A               ***************************/
/************************* Date  :- 01 October 2014      ***************************/
/************************* Q-15):- Arrays                ***************************/
/******************************************************************************/

.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.- File: Arrays.Cpp -.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.
#include <iostream.h>
#include <conio.h>
#include <ctype.h>
#include <process.h>

#include "g:\bin\programs\filework\Q15\header.h"                    //header file containing all the
functions

//*******************************Void Main()*********************************//
void main()
{
 int A[20],B[20],C[20],n,m,l,ele,pos;

 int flag=0;
 char choice;
 int  choice1;

 do
 {
 clrscr();
 cout<<"\tMain Menu"
    <<"\na.\tInput"
    <<"\nb.\tDisplay"
    <<"\nc.\tSearch"
    <<"\nd.\tSort"
    <<"\ne.\tInsert"
    <<"\nf.\tDelete"
    <<"\ng.\tMerge Sort"
    <<"\nh.\tExit";

 cout<<"\n\nEnter Choice(a to h):\t";                    cin>>choice;

 switch(choice)
 {
 case 'a':
         input(A,n);      //n passed by reference
         flag=1;
         break;
 case 'b':
         if(flag)
         display(A,n);
         else
         cout<<"\nArray Not Entered";
         break;
case 'c':
        if(flag)
        {
         cout<<"\nSubmenu\n"
            <<"\n1)\tLinear Search"
            <<"\n2)\tBinary Search";

         cout<<"\nEnter Choice\t";    cin>>choice1;

         cout<<"\nEnter Element\t";
         switch(choice1)
         {
         case 1:
                  cin>>ele;
                  linear_search(A,n,ele);
                  break;
         case 2:
                  cin>>ele;
                  binary_search(A,n,ele);
                  break;
         default:
                  cout<<"\nInvalid Choice";
         }

        }
        else
         cout<<"\nArray Not Entered";
        break;
case 'd':
        if(flag)
        {
         cout<<"\nSubmenu\n"
            <<"\n1)\tInsertion Sort"
            <<"\n2)\tSelection Sort"
            <<"\n3)\tBubble Sort";

         cout<<"\nEnter Choice\t";    cin>>choice1;

         switch(choice1)
         {
         case 1:
                  insertion_sort(A,n);
                  break;
         case 2:
                  selection_sort(A,n);
                  break;
         case 3:
                  bubble_sort(A,n);
                  break;
         default:
                  cout<<"\nInvalid Choice";
         }
         cout<<"\nFinal Array\t     ";
         display_in_line(A,n);
        }
        else
         cout<<"\nArray Not Entered";
        break;
case 'e':
        if(flag)
        {
         cout<<"\nSubmenu\n"
            <<"\n1)\tInsert in Sorted"
            <<"\n2)\tInsert in Unsorted";

         cout<<"\nEnter Choice\t";    cin>>choice1;

         cout<<"\nEnter Element\t";

         switch(choice1)
         {
         case 1:
                  cin>>ele;
                  insert_sorted(A,n,ele);
                  break;
         case 2:
                  cin>>ele;

                  do
                  {
                  cout<<"\nEnter Position";      cin>>pos;

                  if(pos<1 || pos>n+1)
                    cout<<"\nInvalid Position! Please Enter Again.\n\n";
                  }while(pos<1 || pos>n+1);

                  insert_unsorted(A,n,ele,pos);
                  break;
         default:
                  cout<<"\nInvalid Choice";
         }
         cout<<"\nFinal   Array\t   : ";
         display_in_line(A,n);
        }
        else
         cout<<"\nArray Not Entered";
        break;
case 'f':
        if(flag)
        {
         cout<<"\nSubmenu\n"
            <<"\n1)\tDelete in Sorted"
            <<"\n2)\tDelete in Unsorted";
```

```
                 cout<<"\nEnter Choice\t";    cin>>choice1;

                 cout<<"\nEnter Element\t";

                 switch(choice1)
                 {
                 case 1:
                         cin>>ele;
                         cout<<"\nArray Before Delete\n";
                         display_in_line(A,n);
                         delete_sorted(A,n,ele);
                         cout<<"\nArray After Delete\n";
                         display_in_line(A,n);
                         break;
                 case 2:
                         cin>>ele;
                         cout<<"\nArray Before Delete\n";
                         display_in_line(A,n);
                         delete_sorted(A,n,ele);
                         cout<<"\nArray After Delete\n";
                         display_in_line(A,n);
                         break;
                 default:
                         cout<<"\nInvalid Choice";
                 }
                }
                else
                 cout<<"\nArray Not Entered";
                break;
        case 'g':
                if(flag)
                 {
                 cout<<"\nEnter Array B";
                 input(B,m);
                 cout<<"\nSubmenu\n"
                    <<"\n1)\tMerge Sort(A Asc+ B Desc= C Asc)"
                    <<"\n2)\tMerge A at Odd and B at Even in C";

                 cout<<"\nEnter Choice\t";    cin>>choice1;

                 switch(choice1)
                  {
                  case 1:
                         merge_sort(A,n,B,m,C,l);
                         cout<<"A\t";
                         display_in_line(A,n);
                         cout<<"\nB\t";
                         display_in_line(B,m);
                         cout<<"\nC\t";
                         display_in_line(C,l);
                         break;
```

```
          case 2:
                merge_sort_o_e(A,n,B,m,C,l);
                cout<<"A\t";
                display_in_line(A,n);
                cout<<"\nB\t";
                display_in_line(B,m);
                cout<<"\nC\t";
                display_in_line(C,l);
                break;
          default:
                cout<<"Invalid Choice";
          }
         }
         else
          cout<<"Array Not Entered!";

        break;
  case 'h':
        exit(0);
  default :
        cout<<"Invalid Choice\n";
  }

  cout<<"\nPress Y To Continue\t";

  }while(toupper(getche())=='Y');
  getch();
}
//*******************************End Of Main***********************************//
.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.
```

```
.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.- File: Header.h -.- -.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.
//Header file for Arrays.cpp
//////////////////////////////////////////////Declaration of Functions//////////////////////////////////////////////////////
void input(int A[],int &n);
void display(int A[],int n);

void display_in_line(int A[],int n);
void swap(int &a,int &b);

void linear_search(int A[],int n,int ele);
void binary_search(int A[],int n,int ele);

void insertion_sort(int A[],int n);
void selection_sort(int A[],int n);
void bubble_sort(int A[],int n);

void insert_sorted(int A[],int &n,int ele);
void insert_unsorted(int A[],int &n,int ele,int pos);
void delete_sorted(int A[],int &n,int ele);
void delete_unsorted(int A[],int &n,int ele);

void merge_sort(int A[],int n,int B[],int m,int C[],int &l);       //A-Asc+B-Dec=C-Asc
void merge_sort_o_e(int A[],int n,int B[],int m,int C[],int &l);    //odd_even
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////Defination of Functions//////////////////////////////////////////////////////////
void input(int A[],int &n)
{
int i;
cout<<"\nenter the size of the array(1-20)\t";
cin>>n;
cout<<"\n";
for(i=0;i<n;i++)
 {
 cout<<"Enter element\t"<<i+1<<"\t";
 cin>>A[i];
 }
}
//______________________________________________________________________________________________//
void display(int A[],int n)
{
int i;
cout<<"\n";
for(i=0;i<n;i++)
 {
 cout<<"\nelement\t"<<i+1<<"\t";
 cout<<A[i];
 }
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void display_in_line(int A[],int n)
{
int i;
for(i=0;i<n;i++)
 {
  cout<<A[i]<<" ";
 }
}
//______________________________________________________________________________________________//
void swap(int &a,int &b)
{
 int temp;
 temp=a;
 a=b;
 b=temp;
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

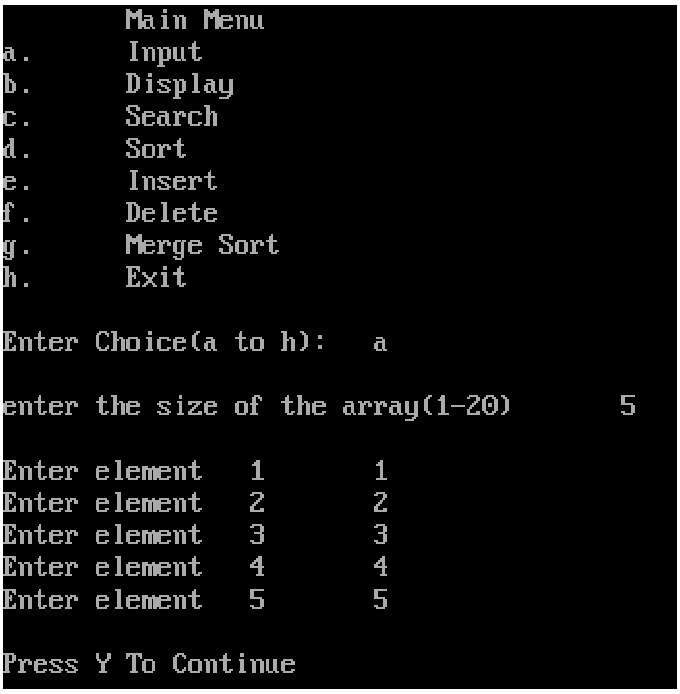
```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void linear_search(int A[],int n,int ele)
{
 int i,flag=0;

 for(i=0;i<n;i++)
 {
  if(A[i]==ele)
   {
    flag=1;
    break;
   }
 }

 if(flag)
  cout<<"\nThe Element Found At\t"<<i+1;
 else
  cout<<"\nElement not found";
}
//_____________________________________________________________________________________________//
void binary_search(int A[],int n,int ele)        //for ascending
{
 int flag=0;
 int u=n-1,l=0,mid;

 do
 {
 mid=(u+l)/2;

 if(A[mid]==ele)
  {
   flag=1;
   break;
  }

 else {
       if(A[mid]>ele)
           u=mid-1;
       else
           l=mid+1;
      }
 }while(u>=l);

 if(flag)
  cout<<"\nThe Element Found At\t"<<mid+1;
 else
  cout<<"\nElement not found";
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void insertion_sort(int A[],int n)
{
 int i,j,ele;
 cout<<"\nInitial Array\t   : ";
 display_in_line(A,n);

 i=1;
 while(i<n)
 {
   ele=A[i];
   j=i-1;
   while(A[j]>ele && j>=0)
    {
    A[j+1]=A[j];
    j--;
    }
   A[j+1]=ele;
   cout<<"\nArray After Pass "<<i<<" : ";
   display_in_line(A,n);
   i++;
 }
}
//________________________________________________________________________________//
void selection_sort(int A[],int n)
{
 int i,j,*min;

 cout<<"\nInitial Array\t   : ";
 display_in_line(A,n);

 for(i=0;i<n-1;i++)
 {
   min=&A[i];
   for(j=i;j<n;j++)
     {
     if(A[j]<*min)
     min=&A[j];
     }
   swap(A[i],*min);
   cout<<"\nArray After Pass "<<i+1<<" : ";
   display_in_line(A,n);
 }
}
//________________________________________________________________________________//
```

```
void bubble_sort(int A[],int n)
{
 int i,j;

 cout<<"\nInitial Array\t   : ";
 display_in_line(A,n);

 for(i=0;i<n-1;i++)
    {
        for(j=0;j<n-i-1;j++)
          if(A[j]>A[j+1])
           swap(A[j],A[j+1]);

    cout<<"\nArray After Pass "<<i+1<<" : ";
    display_in_line(A,n);
    }
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void insert_sorted(int A[],int &n,int ele)
{
 int i=0,pos;
 cout<<"\nInitial Array\t   : ";

 display_in_line(A,n);

 while(A[i]<ele && i<n)
 {
  i++;
 }

 pos=i;

 for(i=n-1;i>=pos;i--)
 {
  A[i+1]=A[i];
 }

 A[pos]=ele;
 n++;
}
//__________________________________________________________________________________________________//


//__________________________________________________________________________________________________//
void insert_unsorted(int A[],int &n,int ele,int pos)
{
 int i;

 cout<<"\nInitial Array\t   : ";
 display_in_line(A,n);
```

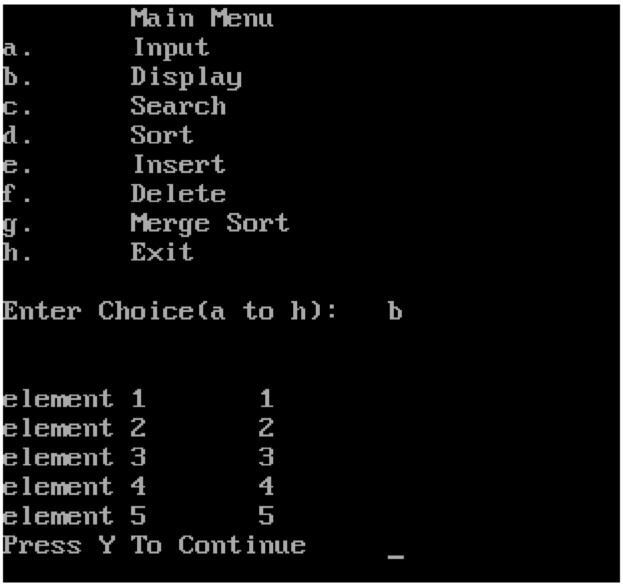
```
for(i=n-1;i>=pos-1;i--)
 {
 A[i+1]=A[i];
 }

 A[pos-1]=ele;
 n++;
}
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void delete_sorted(int A[],int &n,int ele)
{
  int i=0,j=0,flag=0;

  for(i=0;i<n;i++)
   {
   if(A[i]==ele)
    {
    flag=1;
    break;
    }

   else if(A[i]>ele)
    break;
   }

   if(flag==1)
   {
    for(j=i;j<n-1;j++)
           A[j]=A[j+1];

    n--;
   }

   else
   cout<<"\nele not found";
}
//________________________________________________________________________________//
void delete_unsorted(int A[],int &n,int ele)
{
  int i=0,j=0,flag=0;

  for(i=0;i<n;i++)
   {
   if(A[i]==ele)
     {
     flag=1;
     break;
     }
   }
  if(flag)
  {
  for(j=i;j<n-1;j++)
         A[j]=A[j+1];
   n--;
  }
  else
  cout<<"\nele not found";
}
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void merge_sort(int A[],int n,int B[],int m,int C[],int &l)       //A-Asc+B-Dec=C-Asc
{
 int i=0,j=0,k=0;
 l=m+n;

 for(i=0,j=0,k=m-1;i<l &&j<n && k>=0;i++)
 {
  if(A[j]<B[k])
   {
   C[i]=A[j];
   j++;
   }
  else
  {
   C[i]=B[k];
   k--;
  }
 }

 if(k<0)
  {
  for(;i<l;i++)
    C[i]=A[j++];
  }

 if(j>=n)
  {
  for(;i<l;i++)
  C[i]=B[k--];
  }
}
//_______________________________________________________________________________________________//

void merge_sort_o_e(int A[],int n,int B[],int m,int C[],int &l)      //odd_even
{

 if(m==n)
 {
  l=m+n;
  int j=0,k=0,i=0;
 for(i=0;i<l;i++)
  {
  if(i%2==1)   //if odd
   {
   C[i]=A[j++];
   }
  else
   {
   C[i]=B[k++];
   }
  }
 }
 else cout<<"The arrays cannot be merged by odd even as their\n"
            <<"sizes are not equal";
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.
```

OUTPUT

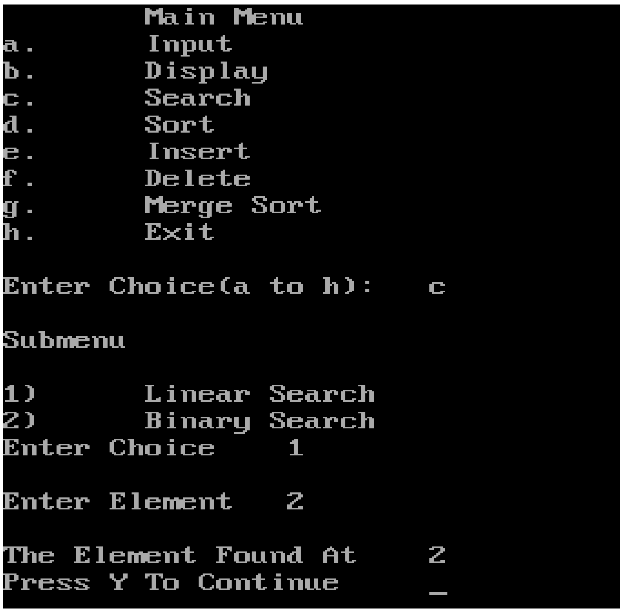


```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   a

enter the size of the array(1-20)       5

Enter element   1       1
Enter element   2       2
Enter element   3       3
Enter element   4       4
Enter element   5       5

Press Y To Continue
```

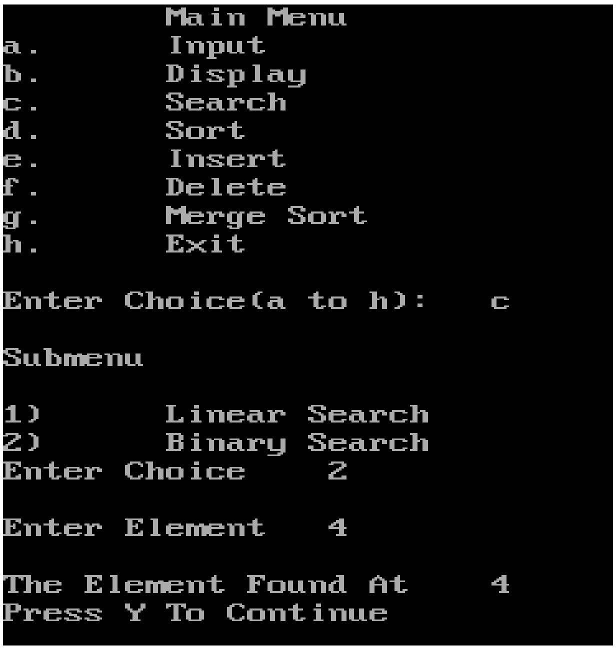
Please Turn Over..



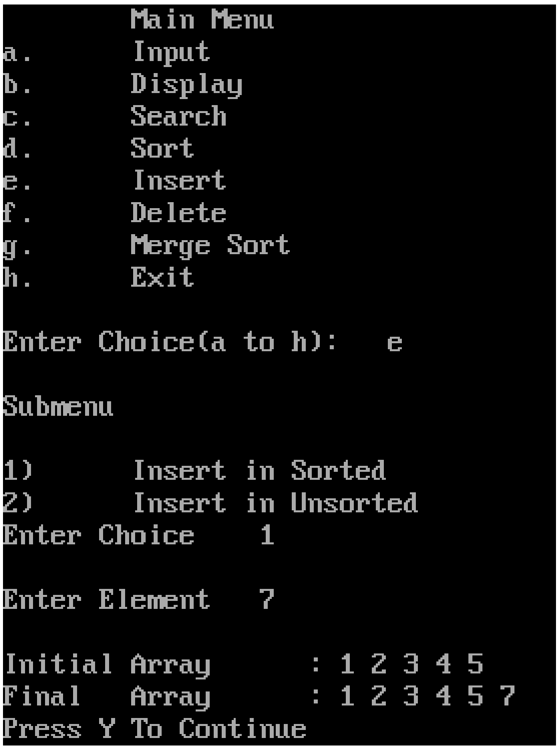
```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   b


element 1       1
element 2       2
element 3       3
element 4       4
element 5       5
Press Y To Continue     _
```



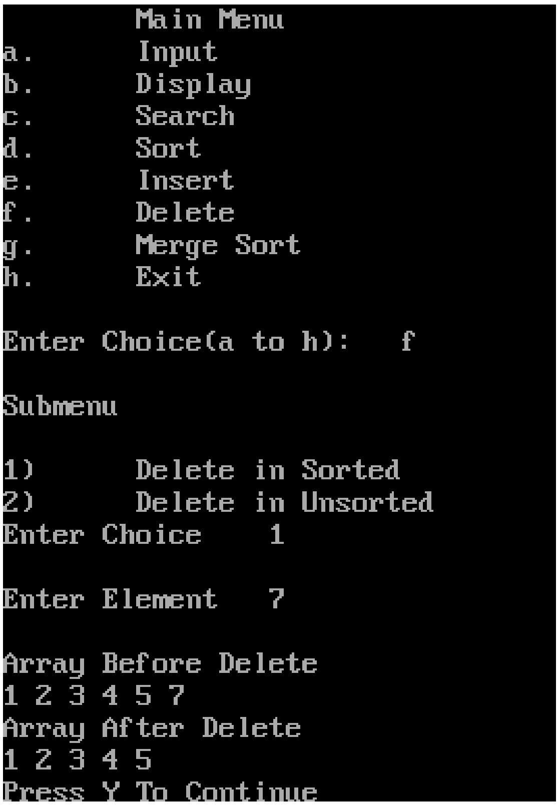
```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   c

Submenu

1)      Linear Search
2)      Binary Search
Enter Choice    1

Enter Element   2

The Element Found At    2
Press Y To Continue     _
```



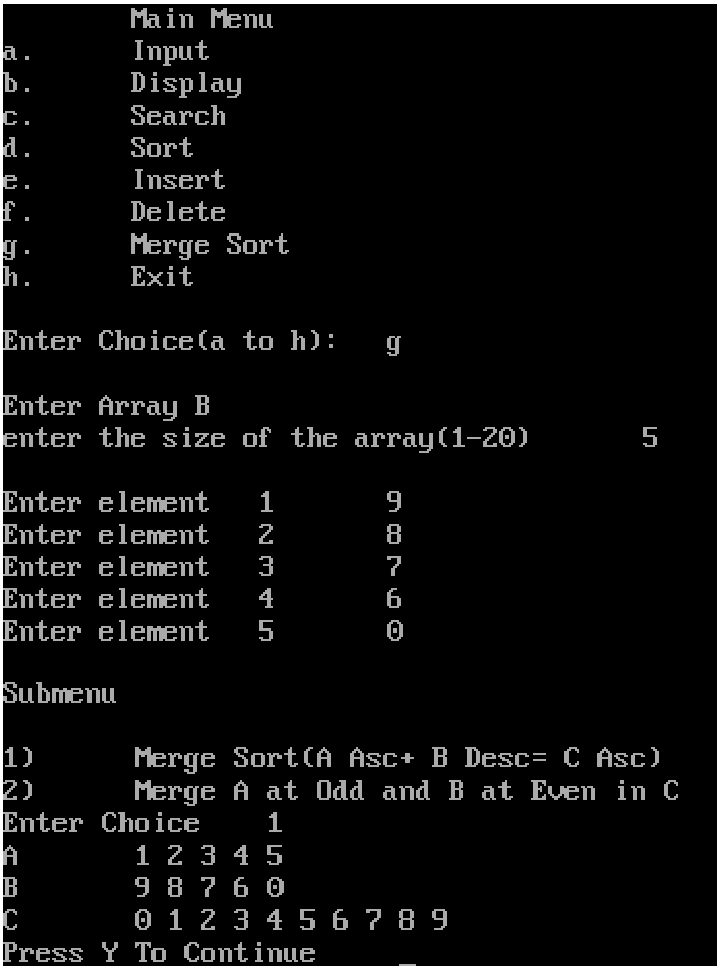
```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   c

Submenu

1)      Linear Search
2)      Binary Search
Enter Choice    2

Enter Element   4

The Element Found At    4
Press Y To Continue
```



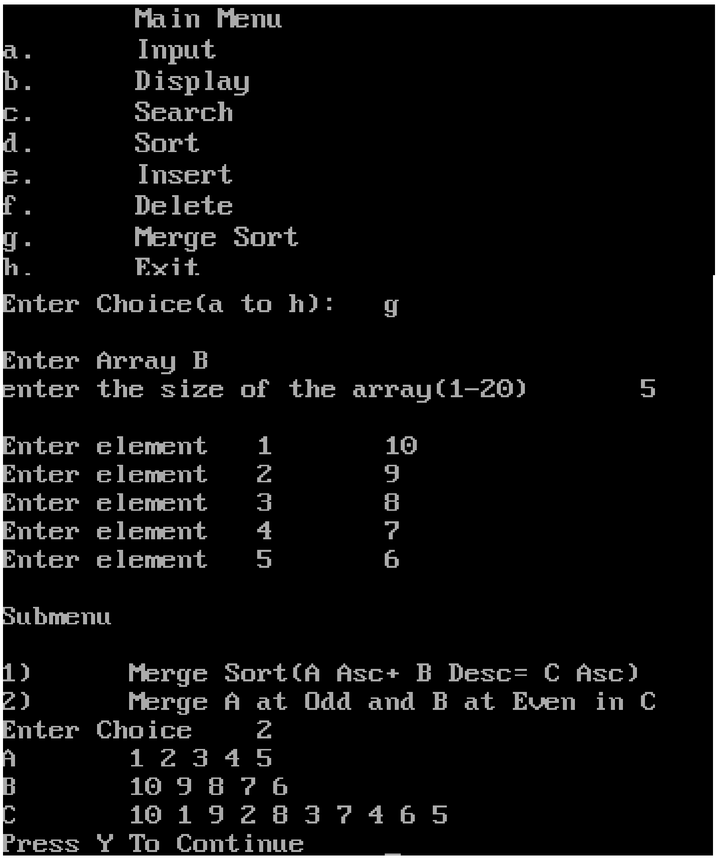
```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   e

Submenu

1)      Insert in Sorted
2)      Insert in Unsorted
Enter Choice    1

Enter Element   7

Initial Array      : 1 2 3 4 5
Final   Array      : 1 2 3 4 5 7
Press Y To Continue
```



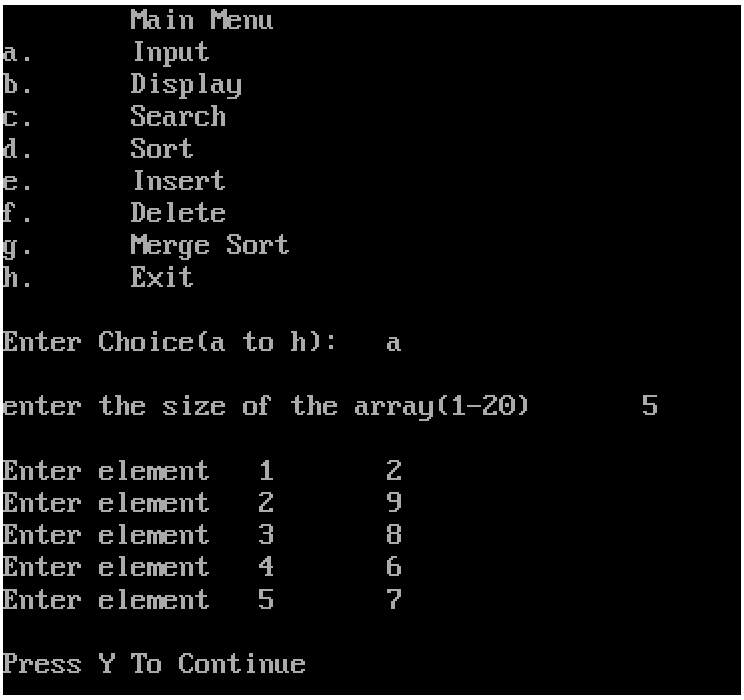
```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   f

Submenu

1)      Delete in Sorted
2)      Delete in Unsorted
Enter Choice    1

Enter Element   7

Array Before Delete
1 2 3 4 5 7
Array After Delete
1 2 3 4 5
Press Y To Continue
```



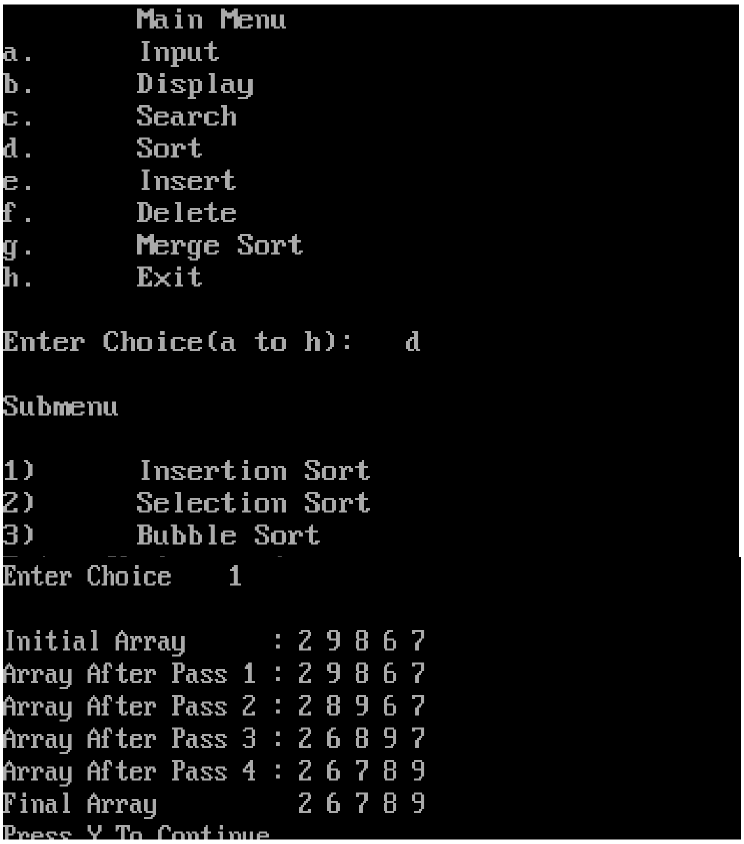
```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   g

Enter Array B
enter the size of the array(1-20)       5

Enter element   1       9
Enter element   2       8
Enter element   3       7
Enter element   4       6
Enter element   5       0

Submenu

1)      Merge Sort(A Asc+ B Desc= C Asc)
2)      Merge A at Odd and B at Even in C
Enter Choice    1
A       1 2 3 4 5
B       9 8 7 6 0
C       0 1 2 3 4 5 6 7 8 9
Press Y To Continue       _
```



```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit
Enter Choice(a to h):   g

Enter Array B
enter the size of the array(1-20)       5

Enter element   1       10
Enter element   2       9
Enter element   3       8
Enter element   4       7
Enter element   5       6

Submenu

1)      Merge Sort(A Asc+ B Desc= C Asc)
2)      Merge A at Odd and B at Even in C
Enter Choice    2
A       1 2 3 4 5
B       10 9 8 7 6
C       10 1 9 2 8 3 7 4 6 5
Press Y To Continue       _
```

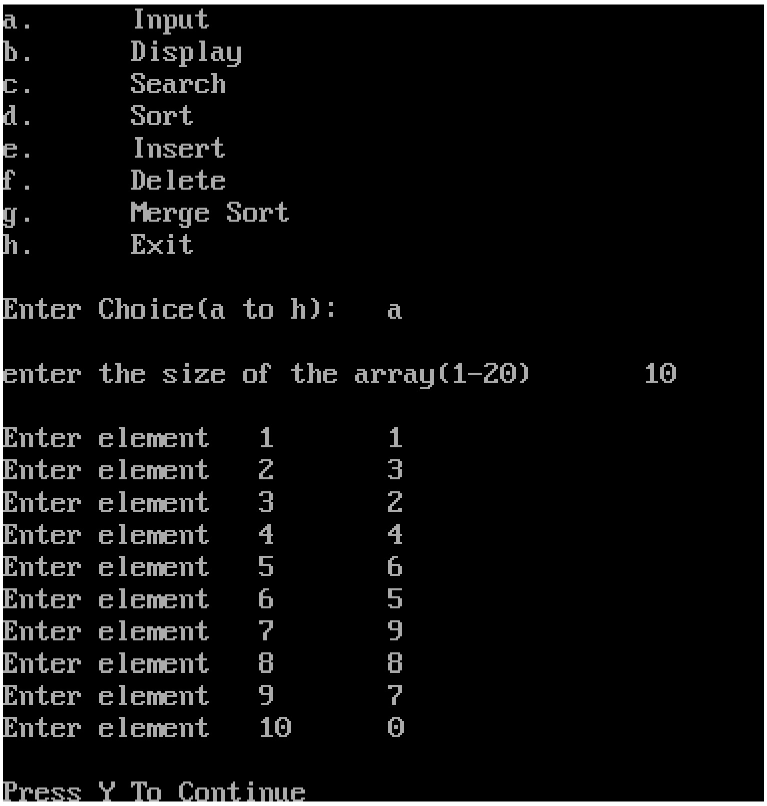


```
         Main Menu
a.       Input
b.       Display
c.       Search
d.       Sort
e.       Insert
f.       Delete
g.       Merge Sort
h.       Exit

Enter Choice(a to h):   a

enter the size of the array(1-20)       5

Enter element   1       2
Enter element   2       9
Enter element   3       8
Enter element   4       6
Enter element   5       7

Press Y To Continue
```

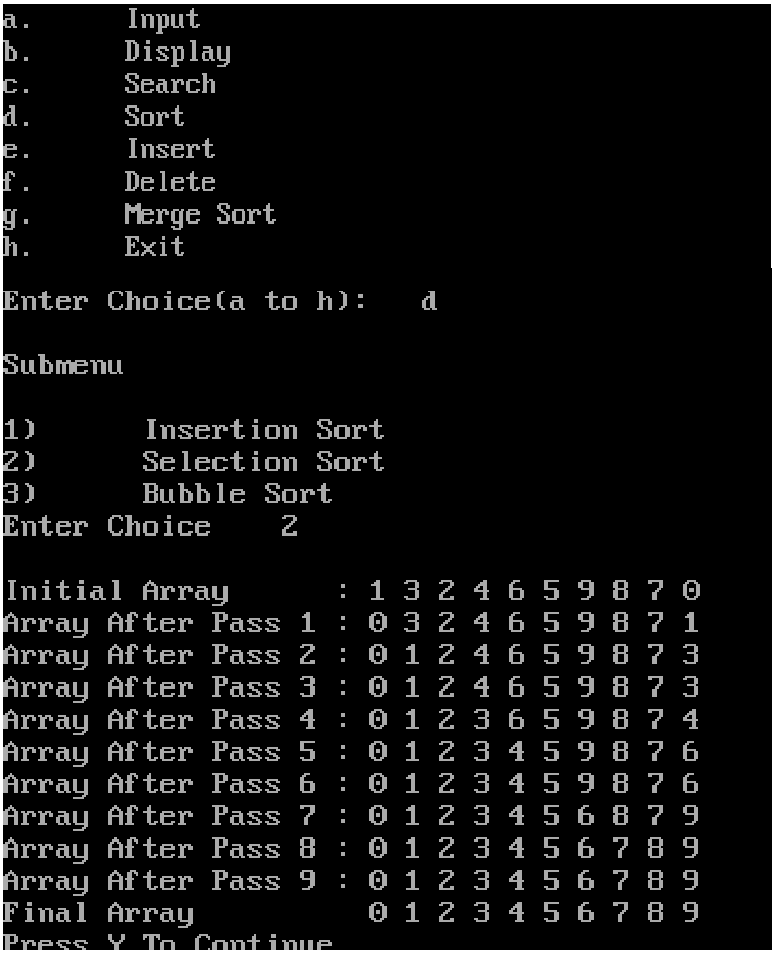


```
         Main Menu
a.       Input
b.       Display
c.       Search
d.       Sort
e.       Insert
f.       Delete
g.       Merge Sort
h.       Exit

Enter Choice(a to h):   d

Submenu

1)       Insertion Sort
2)       Selection Sort
3)       Bubble Sort
Enter Choice    1

Initial Array      : 2 9 8 6 7
Array After Pass 1 : 2 9 8 6 7
Array After Pass 2 : 2 8 9 6 7
Array After Pass 3 : 2 6 8 9 7
Array After Pass 4 : 2 6 7 8 9
Final Array          2 6 7 8 9
Press Y To Continue
```

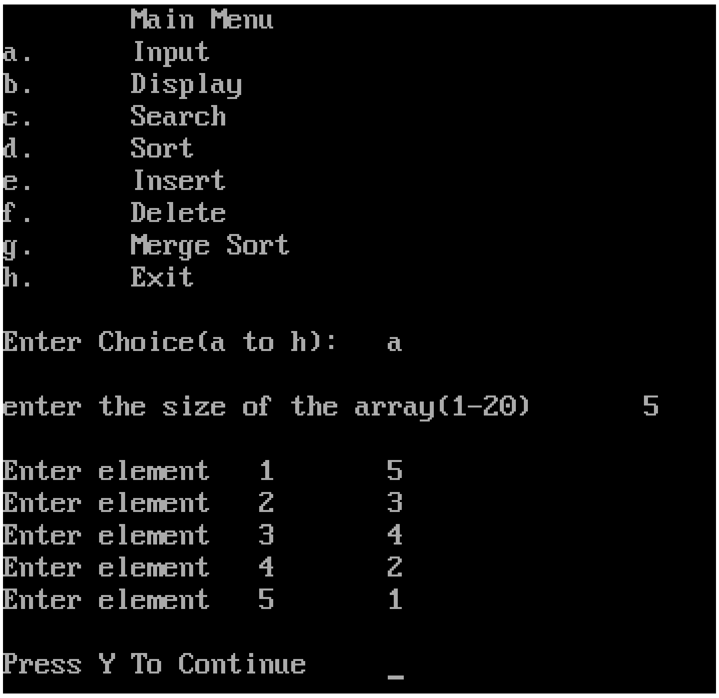
Please Turn Over..



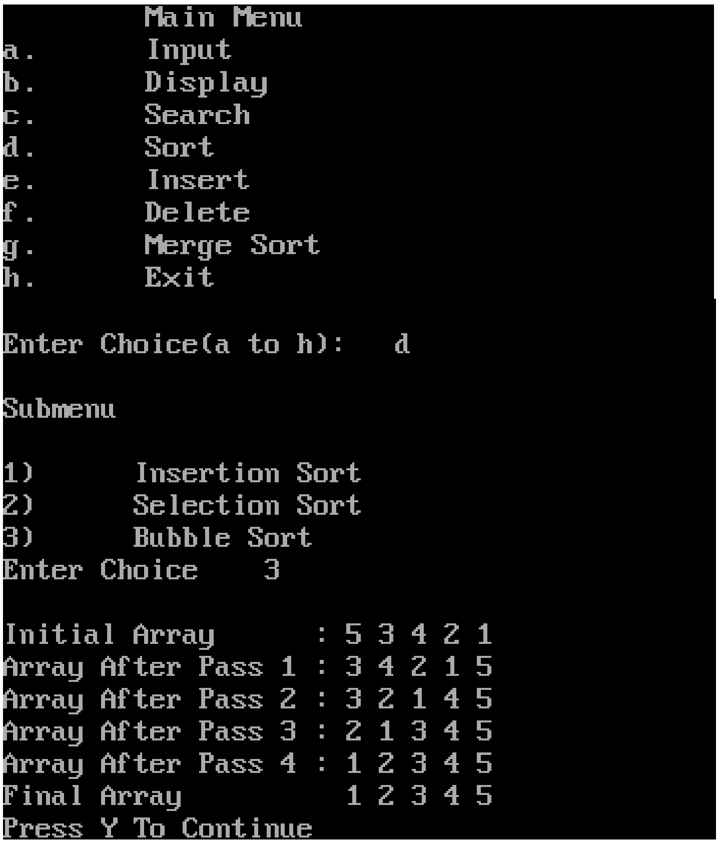
```
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   a

enter the size of the array(1-20)       10

Enter element   1       1
Enter element   2       3
Enter element   3       2
Enter element   4       4
Enter element   5       6
Enter element   6       5
Enter element   7       9
Enter element   8       8
Enter element   9       7
Enter element   10      0

Press Y To Continue
```



```
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   d

Submenu

1)      Insertion Sort
2)      Selection Sort
3)      Bubble Sort
Enter Choice    2

Initial Array      : 1 3 2 4 6 5 9 8 7 0
Array After Pass 1 : 0 3 2 4 6 5 9 8 7 1
Array After Pass 2 : 0 1 2 4 6 5 9 8 7 3
Array After Pass 3 : 0 1 2 4 6 5 9 8 7 3
Array After Pass 4 : 0 1 2 3 6 5 9 8 7 4
Array After Pass 5 : 0 1 2 3 4 5 9 8 7 6
Array After Pass 6 : 0 1 2 3 4 5 9 8 7 6
Array After Pass 7 : 0 1 2 3 4 5 6 8 7 9
Array After Pass 8 : 0 1 2 3 4 5 6 7 8 9
Array After Pass 9 : 0 1 2 3 4 5 6 7 8 9
Final Array          0 1 2 3 4 5 6 7 8 9
Press Y To Continue
```



```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   a

enter the size of the array(1-20)       5

Enter element   1       5
Enter element   2       3
Enter element   3       4
Enter element   4       2
Enter element   5       1

Press Y To Continue     _
```



```
        Main Menu
a.      Input
b.      Display
c.      Search
d.      Sort
e.      Insert
f.      Delete
g.      Merge Sort
h.      Exit

Enter Choice(a to h):   d

Submenu

1)      Insertion Sort
2)      Selection Sort
3)      Bubble Sort
Enter Choice    3

Initial Array      : 5 3 4 2 1
Array After Pass 1 : 3 4 2 1 5
Array After Pass 2 : 3 2 1 4 5
Array After Pass 3 : 2 1 3 4 5
Array After Pass 4 : 1 2 3 4 5
Final Array          1 2 3 4 5
Press Y To Continue
```