Merge Sort

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

```
void mergesort(int b[],int l,int r) {
  if(l<r) {
    int m=(l+(r-1))/2;
    mergesort(b,l,m);
    mergesort(b,m+1,r);
    merge(b,l,m,r); }}
void printarray(int A[],int size) {
  int i;
  for(i=0;i<size;i++){
    printf("%d\t",A[i]); }
  printf("\n"); }
int main(){
  int i,n,a[50];
  printf("Enter the number of elements :\n");
  scanf("%d",&n);
  printf("Enter the elements : ");
  for(i=0;i<n;i++) {
    scanf("%d",&a[i]); }
  mergesort(a,0,n-1);
  printf("The Sorted Array \n");
  printarray(a,n);
  return 0;
}
```

OUTPUT

```
Enter the number of elements :
5
Enter the elements : 23 45 67 99 12
The Sorted Array
12	23	45	67	99
```

Stack Using Array

```
#include<stdio.h>
int a[50],n,top;
void push(int p);
int pop();
void display();
int main(){
  int i,pu;
  printf("Enter the number of elements : \n");
  scanf("%d",&n);
  printf("Enter the elements :\n");
  for(top=0;top<n;top++){
    scanf("%d",&a[top]);}
  printf("Enter the element to be pushed : \n");
  scanf("%d",&pu);
  push(pu);
  printf("The stack is \n");
  for(i=top;i>=0;i--){
    printf("%d\n",a[i]); }
  printf("The popped element is %d\n",pop());
  printf("The Current stack is \n");
  display();
  return 0;
}
void push(int p){
  a[top]=p;
}
int pop(){
  int po;
  po=a[top];
  top--;
  return po;
}
void display(){
  int i;
  for(i=top;i>=0;i--){
  printf("%d\n",a[i]);}
  return;
}
```

OUTPUT

```
Enter the number of elements :
4
Enter the elements :
1
2
3
4
Enter the element to be pushed :
9
The stack is
9
4
3
2
1
The popped element is 9
The Current stack is
4
3
2
1
```

Linear Search

```
#include<stdio.h>
int a[50];
void linear(int n,int ITEM)
{
  int i,LOC=-1;
  for(i=0;i<n;i++){
    if(a[i]==ITEM){
      printf("Item found at %d",i+1);
      LOC=0;
    }}
  if(LOC==-1)
  printf("The element not found");
}
int main()
{
  int i,n,ITEM;
  printf("Enter the number of elements in the array \n");
  scanf("%d",&n);
  printf("Enter the elements \n");
  for(i=0;i<n;i++){
    scanf("%d",&a[i]); }
  printf("Enter the element to be found \n");
  scanf("%d",&ITEM);
  linear(n,ITEM);
  return 0;
}
```

Binary Search

```
#include<stdio.h>
#include<stdlib.h>
int a[10];
void binary(int n,int ITEM) {
  int LOC=-1;
  int first=0;
  int last=n-1;
  while(first<last) {
    int middle=(first+last)/2;
    if(a[middle]==ITEM) {
      printf("Element is found at %d\n",middle+1);
      LOC=0;
      exit(0); }
    else if(ITEM>a[middle])
      first=middle+1;
    else if(ITEM<a[middle])
      last=middle-1; }
  if(LOC==-1)
    printf("Item not found\n"); }
int main() {
  int i,n,ITEM;
  printf("Enter the no. of elements in the array\n");
  scanf("%d",&n);
  printf("Enter the elements in the ascending order\n");
  for(i=0;i<n;i++)
    scanf("%d",&a[i]);
  printf("Enter the element to be found\n");
  scanf("%d",&ITEM);
  binary(n,ITEM);
  return 0;  }
```

Insertion Sort

```
#include<stdio.h>
void insertionSort(int array[],int n){
   int i,j,element;
   for(i=1;i<n;i++){
      element=array[i];
      for(j=i-1;j>=0 && array[j]>element;j--){
         array[j+1]=array[j];}
         array[j+1]=element;
   }}
void display(int array[],int n){
   int i;
   for(i=0;i<n;i++){
   printf("%d ",array[i]);}
}
int main(){
   int i,n,a[50];
   printf("Enter the number of elements : \n");
   scanf("%d",&n);
   printf("Enter the elements of the unsorted array : \n");
   for(i=0;i<n;i++){
       scanf("%d",&a[i]); }
   insertionSort(a,n);
   printf("The sorted array is \n");
   display(a,n);
   return 0;
}
```

Bubble Sort

```
#include<stdio.h>
void bubble(int *p,int n){
   int i,j,temp;
   for(i=0;i<n;i++){
      for(j=0;j<n-i-1;j++){
         if(p[j]>p[j+1]){
            temp=p[j];
            p[j]=p[j+1];
            p[j+1]=temp;
         }}}
   printf("Sorted Array is \n");
   for(i=0;i<n;i++){
      printf("%d\n",p[i]);
   }}
int main(){
   int i,n,a[50],*p;
   printf("Enter the number of elements :\n");
   scanf("%d",&n);
   printf("Enter the elements : ");
   for(i=0;i<n;i++){
      scanf("%d",&a[i]);
   }
   p=a;
   bubble(p,n);
   return 0;
}
```

Selection Sort

```
#include<stdio.h>
void select(int *p,int n){
  int i,j,temp,min;
  for(i=0;i<n;i++){
    min=i;
    for(j=i+1;j<n;j++){
      if(p[min]>p[j]){
        min=j;
      }}
    temp=p[i];
    p[i]=p[min];
    p[min]=temp; }
  printf("The Sorted Array is \n");
  for(i=0;i<n;i++){
    printf("%d\t",p[i]);
  }}
int main(){
  int a[50],i,n,*p;
  printf("Enter the number of elements : \n");
  scanf("%d",&n);
  printf("Enter the elements :\n");
  for(i=0;i<n;i++){
    scanf("%d",&a[i]);}
  p=a;
  select(p,n);
  return 0;
}
```

Stack Using Linked List

```
#include<stdio.h>
#include<stdlib.h>
struct node {
  int data;
  struct node *link; };
void display(struct node **top) {
  struct node *p;
  for(p=(*top);p!=NULL;) {
    printf("%d\n",p->data);
    p=p->link;
  }}
void push(struct node **top,int val) {
  struct node *newnode;
  newnode=(struct node *)malloc(sizeof(struct node));
  newnode->data=val;
  newnode->link=*top;
  *top=newnode;
}
void pop(struct node **tp) {
  if(*tp==NULL)
  printf("Underflow\n");
  else
  *tp=(*tp)->link;
}
int main() {
  struct node *top=NULL;
  int i,n,value,choice,ch1=1;
  char cho;
  printf("Enter the number of elements: \n");
  scanf("%d",&n);
```

```
   printf("Enter the elements : \n");
   for(i=1;i<=n;i++) {
     scanf("%d",&value);
     push(&top,value); }
   do {
     printf("Choose an option :\n1.PUSH\n2.POP\n3.DISPLAY\n");
     scanf("%d",&choice);
     switch(choice) {
       case 1:
       printf("Enter the value to be pushed : \n");
       scanf("%d",&value);
       push(&top,value);
       break;
       case 2:
       pop(&top);
       break;
       case 3:
       display(&top);
       break;
     }
     printf("Continue ? y/n:\n");
     scanf(" %c",&cho);
   }while(cho=='y'|| cho=='Y');
  return 0;
}
```

Queue using linked list

```
#include<stdio.h>
#include<stdlib.h>
struct node{
   int data;
   struct node *link; };
void enqueue(int value,struct node **fr,struct node **re){
   struct node *newnode;
   newnode=(struct node *)malloc(sizeof(struct node));
   newnode->data=value;
   newnode->link=NULL;
   if(*fr==NULL){
      *fr=*re=newnode;
      return; }
   (*re)->link=newnode;
   *re=newnode; }
void dequeue(struct node **fr){
   if(*fr==NULL)
   printf("Underflow\n");
   else{
      printf("Dequeued element is %d\n",(*fr)->data);
      *fr=(*fr)->link;
   }}
void display(struct node **fr){
   struct node *p;
   printf("The queue is \n");
   for(p=*fr;p!=NULL;){
      printf("%d\n",p->data);
      p=p->link;
   }}
int main(){
 struct node *front=NULL,*rear=NULL;
 int i,n,value,choice;
 char cho='y';
 printf("Enter the number of elements : \n");
 scanf("%d",&n);
 printf("Enter the elements:\n");
 for(i=1;i<=n;i++) {
   scanf("%d",&value);
   enqueue(value,&front,&rear); }
 do{
   printf("Choose an option\n1.ENQUEUE\n2.DEQUEUE\n3.DISPLAY\n");
   scanf("%d",&choice);
   switch(choice){
     case 1:
     printf("Enter the value to be enqueued : \n");
     scanf("%d",&value);
     enqueue(value,&front,&rear);
     break;
     case 2:
     dequeue(&front);
     break;
     case 3:
     display(&front);
     break; }
   printf("Continue ? y/n\n");
   scanf(" %c",&cho);
 }while(cho=='y'|| cho=='Y');
 return 0;
}
```

List

```
#include<stdio.h>
#include<stdlib.h>
struct node{
  int data;
  struct node *link; };
struct node *front=NULL,*rear=NULL,*p;
void entry(int value){
  struct node *newnode;
  newnode=(struct node *)malloc(sizeof(struct node));
  newnode->data=value;
  newnode->link=NULL;
  if(front==NULL){
    front=rear=newnode;
    return; }
  rear->link=newnode;
  rear=newnode;
}
void insert(int ins,int pis){
  int j;
  struct node *newnode;
  newnode=(struct node *)malloc(sizeof(struct node));
  newnode->data=ins;
  for(j=1,p=front;j<pis;j++){
    p=p->link; }
  newnode->link=p;
  for(j=1,p=front;j<pis-1;j++){
    p=p->link; }
  p->link=newnode; }
void del(int a){
  struct node *prev,*curr,*temp;
  if(front->data==a){
    front=front->link;
  }
  for(prev=curr=front;curr!=NULL;){
    if(curr->data==a){
      temp=curr;
      prev->link=curr->link;
      free(temp);
      break; }
    prev=curr;
    curr=curr->link;
  }}
void display(){
  printf("The list is \n");
  for(p=front;p!=NULL;){
    printf("%d\n",p->data);
    p=p->link; }}
int main(){
  int i,n,value,choice,pos;
  char cho;
  printf("Enter the number of elements :\n");
  scanf("%d",&n);
  printf("Enter the elements :\n");
  for(i=1;i<=n;i++){
    scanf("%d",&value);
    entry(value); }
  do{
    printf("Choose an option\n1.Insert\n2.Delete\n");
    scanf("%d",&choice);
    switch(choice){
      case 1:
      printf("Enter the value to be inserted\n");
      scanf("%d",&value);
      printf("Enter the position to be inserted\n");
      scanf("%d",&pos);
      insert(value,pos);
      display();
      break;
      case 2:
      printf("Enter the value to be deleted\n");
      scanf("%d",&value);
      del(value);
      display();
      break;
    }
    printf("Continue ? y/n \n");
    scanf(" %c",&cho);
  }while( cho=='y' || cho=='Y');
  return 0;
}
```

OUTPUT

```
Enter the number of elements :
3
Enter the elements :
12
13
15
Choose an option
1.Insert
2.Delete
1
```

Queue using Array

```
#include<stdio.h>
# define MAX 50
int front=0,rear=-1,b[MAX];
void enqueue(int a);
void dequeue();
void display();
int main(){
  int choice,e;
  char c;
  do{
    printf("Choose an option\n1.Add an element\n2.Remove an element\n");
    scanf("%d",&choice);
    switch(choice){
      case 1:
      printf("Enter the element : \n");
      scanf("%d",&e);
      enqueue(e);
      break;
      case 2:
      dequeue();
      break; }
    printf("Continue ? Y/N\n");
    scanf(" %c",&c);
  }while(c=='y' || c=='Y');
  return 0; }
void enqueue(int a){
  if(rear<MAX-1){
    rear++;
    b[rear]=a;
    display(); }
  else
  printf("OVERFLOW\n");
  return; }
void dequeue(){
  if(front<=rear){
    front++;
    display();}
  else
  printf("UNDERFLOW\n");
  return;
}
void display(){
  int i;
  printf("The Queue is \n");
  for(i=front;i<=rear;i++){
  printf("%d\n",b[i]); }
  return;
}
```

OUTPUT

```
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
12
The Queue is
12
Continue ? Y/N
Y
Choose an option
```

Binary Search Tree

```
#include<stdio.h>
#include<stdlib.h>
struct node{
   int data;
   struct node *lchild,*rchild; };
struct node *root=NULL,*found,*parent;
struct node* indisp(struct node *root){
    if(root!=NULL)  {
      indisp(root->lchild);
      printf("%d\t",root->data);
      indisp(root->rchild); }}
void insert(int key) {
   struct node *new_node,*temp,*parent;
   int flag=0;
   new_node=(struct node*)malloc(sizeof(struct node));
   new_node->data=key;
   new_node->lchild=NULL;
   new_node->rchild=NULL;
   if(root==NULL) {
      root=new_node;
      return; }
   for(temp=parent=root;temp!=NULL;){
      if(key<temp->data) {
         parent=temp;
         temp=temp->lchild;
         flag=1; }
      else if(key>temp->data) {
         parent=temp;
         temp=temp->rchild;
         flag=2; } }
```

```
  if(flag==1) {
    parent->lchild=new_node; }
  else if(flag==2) {
    parent->rchild=new_node;}}
void search(int val) {
  struct node *temp;
  int flag=0;
  for(temp=root;temp!=NULL;){
    if(temp->data==val) {
      found=temp;
      return ;}
    if(val<temp->data) {
      parent=temp;
      temp=temp->lchild; }
    else if(val>temp->data) {
      parent=temp;
      temp=temp->rchild;  }  }  }
void delete(int val) {
  struct node *curr=NULL,*next=NULL,*prev;
  int flag;
  if(root==NULL){
    printf("Tree is empty");
    return; }
  search(val);
  if(found==root && found->lchild==NULL && found->rchild==NULL) {
    root=NULL;
    printf("Tree is now empty");
    return; }
  if(found->lchild==NULL && found->rchild==NULL) {
    if(parent->lchild==found)
      parent->lchild=NULL;
    else
      parent->rchild=NULL;
    found=NULL;
    return; }
  curr=found;
  if(found->rchild!=NULL) {
    next=found->rchild;
    flag=1;}
  else{
    next=found->lchild;
    flag=2; }
  if(flag==1){
  while(next!=NULL) {
    prev=curr;
    curr=next;
    next=next->lchild;  }}
  if(flag==2){
    while(next!=NULL){
    prev=curr;
    curr=next;
    next=next->rchild; }}
  found->data=(curr)->data;
  if(curr->lchild==NULL&&curr->rchild==NULL) {
  if(curr==prev->lchild)
    prev->lchild=NULL;
  else
    prev->rchild=NULL;
  curr=NULL; }
  else {
  if(curr==prev->lchild){
    prev->lchild=curr->rchild;
```

```
        curr->rchild=NULL; }
      else{
        prev->rchild=curr->lchild;
        curr->lchild=NULL;  }  } }
int main() {
  int n,data,del,choice;
  printf("Enter the no. of elements\n");
  scanf("%d",&n);
  while(n>0) {
      printf("Enter the data :");
      scanf("%d",&data);
      insert(data);
      n--; }
  indisp(root);
  do{
    printf("\nChoose an option\n1.Insert\n2.Delete\n3.exit\n");
    scanf("%d",&choice);
    switch(choice) {
        case 1:
          printf("Enter the data :");
          scanf("%d",&data);
          insert(data);
          indisp(root);
          break;
        case 2:
          printf("Enter the value to be deleted : ");
          scanf("%d",&del);
          delete(del);
          indisp(root);
          break; }
  }while(choice!=3);
```

```
   return 0;
}
```

OUTPUT

```
Enter the no. of elements
5
Enter the data :12
Enter the data :14
Enter the data :17
Enter the data :19
Enter the data :10
10     12     14     17     19
Choose an option
1.Insert
2.Delete
3.exit
1
Enter the data :15
10     12     14     15     17     19
Choose an option
1.Insert
2.Delete
3.exit
2
Enter the value to be deleted : 19
10     12     14     15     17
Choose an option
1.Insert
2.Delete
3.exit
3
```

Doubly Linked List

```
#include<stdio.h>
#include<stdlib.h>
struct node {
   int data;
   struct node *prev;
   struct node *next;
}*head,*tail=NULL;
void forward_display() {
   struct node *temp;
   for(temp=head;temp!=NULL;) {
      printf("%d\t",temp->data);
      temp=temp->next; }}
void backward_display() {
   struct node *temp;
   for(temp=tail;temp!=NULL;) {
      printf("%d\t",temp->data);
      temp=temp->prev; }}
struct node* create_newnode(int value) {
   struct node *newnode;
   newnode=(struct node*)malloc(sizeof(struct node));
   newnode->data=value;
   newnode->prev=newnode->next=NULL;
   return newnode; }
void insert_begin(int value) {
   struct node* newnode = create_newnode(value);
   if(head==NULL) {
      head=tail=newnode;  }
   else {
   head->prev=newnode;
   newnode->next=head;
```

```
  head=newnode; }}
void insert_end(int value) {
  struct node* newnode = create_newnode(value);
  if(head==NULL) {
    head=tail=newnode; }
  else {
  tail->next=newnode;
  newnode->prev=tail;
  tail=newnode;  } }
void insert_after(int value,int key) {
  int flag=0;
  struct node *newnode,*temp;
  newnode = create_newnode(value);
  if(head==NULL)  {
    head=tail=newnode; }
  else {
  for(temp=head;temp!=NULL;) {
    if(temp->data==key) {
      flag=1;
      newnode->next=temp->next;
      newnode->prev=temp;
      temp->next=newnode;
      if(newnode->next!=NULL)
        newnode->next->prev=newnode; }
    temp=temp->next; }
  if(flag==0)
    printf("\nElement not present in the list\n"); }  }
void delete_begin() {
  struct node* temp;
  if(head==NULL) {
    printf("\nlist is empty\n");
```

```
      return; }
    if(head==tail) {
      head=tail=NULL;
      return;   }
    temp=head;
    head=temp->next;
    head->prev=NULL;
    temp->next=NULL;
    free(temp); }
void delete_end() {
    struct node* temp;
    if(head==NULL) {
      printf("\nlist is empty\n");
      return; }
    if(head==tail)  {
      head=tail=NULL;
      return;  }
    temp=tail;
    tail=temp->prev;
    tail->next=NULL;
    temp->prev=NULL;
    free(temp);  }
void delete_element(){
    int key;
    struct node *temp;
    if(head==NULL) {
      printf("\nlist is empty\n");
      return; }
    printf("\nEnter the element to be deleted : ");
    scanf("%d",&key);
    if(head==tail) {
      head=tail=NULL;
      return; }
    for(temp=head;temp!=NULL;) {
      if(temp->data==key) {
        if(temp==head) {
          delete_begin();
          return; }
        if(temp==tail) {
          delete_end();
          return;  }
        temp->next->prev=temp->prev;
        temp->prev->next=temp->next;
        temp->prev=temp->next=NULL;
        free(temp); }
      temp=temp->next; } }
int main() {
    int choice,num,key;
    do {
    printf("\nChoose an option");
    printf("\n1.Insert at the beginning");
    printf("\n2.Insert at the end");
    printf("\n3.Insert after an element");
    printf("\n4.Delete at the beginning");
    printf("\n5.Delete at the end");
    printf("\n6.Delete an element");
    printf("\n7.Forward display");
    printf("\n8.Backward display");
    printf("\n9.Exit\n\n");
    scanf("%d",&choice);
    switch(choice) {
      case 1:
```

```
printf("\nEnter the value to be inserted : ");
scanf("%d",&num);
insert_begin(num);
break;
case 2:
printf("\nEnter the value to be inserted : ");
scanf("%d",&num);
insert_end(num);
break;
case 3:
printf("\nEnter the value to be inserted : ");
scanf("%d",&num);
printf("\nEnter the element after which the value to be inserted : ");
scanf("%d",&key);
insert_after(num,key);
break;
case 4:
delete_begin();
break;
case 5:
delete_end();
break;
case 6:
delete_element();
break;
case 7:
forward_display();
break;
case 8:
backward_display();
break; }
```

```
    }while(choice!=9);
    return 0;
}
```

OUTPUT

```
Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

1

Enter the value to be inserted : 23

Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit
```

String Palindrome – Linked List

```
#include<stdio.h>
#include<stdlib.h>
struct node {
   char ch;
   struct node* prev;
   struct node* next; };
struct node *head=NULL,*tail=NULL;
void create_node(char ch) {
   struct node *newnode;
   newnode=(struct node*)malloc(sizeof(struct node));
   newnode->ch=ch;
   newnode->prev=NULL;
   newnode->next=NULL;
   if(head==NULL)
      head=tail=newnode;
   else {
      tail->next=newnode;
      newnode->prev=tail;
      tail=newnode; } }
int is_palindrome() {
   struct node *forward,*backward;
   for(forward=head,backward=tail;(backward->next!=forward)&&(forward!=backward);) {
      if(forward->ch!=backward->ch) {
         return 0; }
      forward=forward->next;
      backward=backward->prev;   }
   return 1; }
int main() {
   char string[50];
   int i,result;
```

```
  printf("Enter the string\n");
  scanf("%[^\n]",&string);
  for(i=0;string[i]!='\0';i++)
  {
    create_node(string[i]);
  }
  result=is_palindrome();
  if(result==1)
    printf("String is a Palindrome\n");
  else
    printf("String is NOT a Palindrome\n");
  return 0;
}
```

OUTPUT 1

```
Enter the string
madam
String is a Palindrome
```

OUTPUT 2

```
Enter the string
PALINDROME
String is NOT a Palindrome
```

Sparse Matrix

```
#include<stdio.h>
int array1[50][50],array2[50][50],m,n,p;
void display(int b[][50],int row,int col) {
  int i,j;
  for(i=0;i<row;i++) {
    for(j=0;j<col;j++)  {
      printf("%d\t",b[i][j]); }
    printf("\n"); } }
void matrix_enter() {
  int i,j;
  printf("\nNo. of rows : ");
  scanf("%d",&m);
  printf("No. of columns : ");
  scanf("%d",&n);
  printf("\nEnter the elements\n");
  for(i=0;i<m;i++)   {
    for(j=0;j<n;j++)  {
      scanf("%d",&array1[i][j]); }  }
  printf("\nThe matrix is\n");
  display(array1,m,n); }
void sparse() {
  int a=1,i,j,b;
  array2[0][0]=m;
  array2[0][1]=n;
  for(i=0;i<m;i++)    {
    for(j=0;j<n;j++) {
      if(array1[i][j]!=0)    {
        array2[a][0]=i;
        array2[a][1]=j;
        array2[a][2]=array1[i][j];
```

```
            a++;  } } }
   array2[0][2]=a-1;
   printf("\nThe sparse matrix is\n");
   display(array2,a,3);  }
int main() {
   matrix_enter();
   sparse();
   return 0;
}
```

OUTPUT

```
No. of rows : 2
No. of columns : 3

Enter the elements
16
0
78
0
89
0

The matrix is
16    0     78
0     89    0

The sparse matrix is
2     3     3
0     0     16
0     2     78
1     1     89
```

Breadth First Search

```
#include<stdio.h>
int queue[10],is_visited[10],is_added[10],front=0,rear=-1,vertices=5,adj_matrix[10][10];
void enqueue(int vertice){
    queue[++rear]=vertice;  }
int dequeue(){
  int element=queue[front++];
  return element; }
void matrix(){
  for(int i=0;i<vertices;i++) {
    is_visited[i]=0;
    is_added[i]=0;
    for(int j=0;j<vertices;j++)  {
      adj_matrix[i][j]=0;  } }
  adj_matrix[0][1]=adj_matrix[0][2]=adj_matrix[0][3]=1;
  adj_matrix[1][0]=1;
  adj_matrix[2][0]=adj_matrix[2][4]=1;
  adj_matrix[3][0]=1;
  adj_matrix[4][2]=1;  }
int is_not_empty(){
  if(front<=rear)
    return 1;
  else
    return 0; }
int main() {
  int i,j;
  matrix();
  int start=0;
  enqueue(start);
  is_added[start]=1;
  while(is_not_empty()){
    i=dequeue();
    for(j=0;j<vertices;j++){
      if(adj_matrix[i][j]==1) {
          if(is_added[j]!=1) {
              enqueue(j);
              is_added[j]=1;
          }}}
    printf("%d ",i);
    is_visited[i]=1;
    //i=dequeue();
  }
}
```

OUTPUT

```
0 1 2 3 4
```

Depth First Search

```
#include<stdio.h>
int stack[10],is_visited[10],is_added[10],top=-1,vertices=5,adj_matrix[10][10];
void push(int vertex) {
    stack[++top]=vertex; }
int pop(){
  int element=stack[top--];
  return element; }
void matrix(){
  for(int i=0;i<vertices;i++) {
    is_visited[i]=0;
    is_added[i]=0;
    for(int j=0;j<vertices;j++) {
      adj_matrix[i][j]=0; } }
  adj_matrix[0][1]=adj_matrix[0][2]=adj_matrix[0][3]=1;
  adj_matrix[1][0]=1;
  adj_matrix[2][0]=adj_matrix[2][4]=1;
  adj_matrix[3][0]=1;
  adj_matrix[4][2]=1; }
int is_not_empty(){
  if(top!=-1)
    return 1;
  else
    return 0; }
int main() {
  int i,j;
  matrix();
  int start=0;
  push(start);
  is_added[start]=1;
  while(is_not_empty()){
    i=pop();
    for(j=0;j<vertices;j++){
      if(adj_matrix[i][j]==1) {
        if(is_added[j]!=1) {
          push(j);
          is_added[j]=1;
        }}}
    printf("%d ",i);
    is_visited[i]=1;
    //i=pop(); }
}
```

OUTPUT

```
0 3 2 4 1
```

Call and Return

```
#include<stdio.h>
int top=-1;
char a[50];
void push(char ch);
void pop();
void display();
int main() {
  int i,choice;
  char ch,p;
  do{
    printf("Choose an option\n1.Call\n2.Return\n3.Display\n");
    scanf("%d",&choice);
    switch(choice){
      case 1:
      printf("Enter the character\n");
      scanf(" %c",&ch);
      push(ch);
      break;
      case 2:
      pop();
      break;
      case 3:
      printf("The Stack is\n");
      display();
      break;  }
    printf("Continue ? Y/N\n");
    scanf(" %c",&p);
  }while(p=='y' || p=='Y');
  return 0; }
void push(char ch){
  top++;
  a[top]=ch;
  return; }
void pop(){
  top--;
  return; }
void display(){
  int i;
  for(i=top;i>=0;i--){
    printf(" %c\n",a[i]);  }
  if(top==-1){
  printf("Finished\n");}
  return;
}
```

OUTPUT

```
Choose an option
1.Call
2.Return
3.Display
1
Enter the character
h
Continue ? Y/N
y
Choose an option
1.Call
2.Return
3.Display
1
Enter the character
```

Circular Queue

```
#include<stdio.h>
# define SIZE 10
int front=0, rear=-1,a[SIZE];
void enqueue(int val){
  if((rear+2)%SIZE==front)
    printf("OVERFLOW\n");
  else {
  rear=(rear+1)%SIZE;
  a[rear]=val;}
  display(); }
void dequeue(){
  if((rear+1)%SIZE==front)
  printf("UNDERFLOW\n");
  else{
    front=(front+1)%SIZE;}
    display(); }
void display(){
  int p;
  printf("The Queue is\n");
  for(p=front;(rear+1)%SIZE!=p;){
    printf("%d\n",a[p]);
    p=(p+1)%SIZE; } }
int main(){
  int choice,e;
  char c;
  do{
    printf("Choose an option\n1.Add an element\n2.Remove an element\n");
    scanf("%d",&choice);
    switch(choice){
      case 1:
      printf("Enter the element : \n");
      scanf("%d",&e);
      enqueue(e);
      break;
      case 2:
      dequeue();
      break; }
    printf("Continue ? y/n\n");
    scanf(" %c",&c);
  }while(c=='Y' || c=='y');
  return 0;
}
```

OUTPUT

```
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
34
The Queue is
34
Continue ? y/n
y
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
67
The Queue is
```

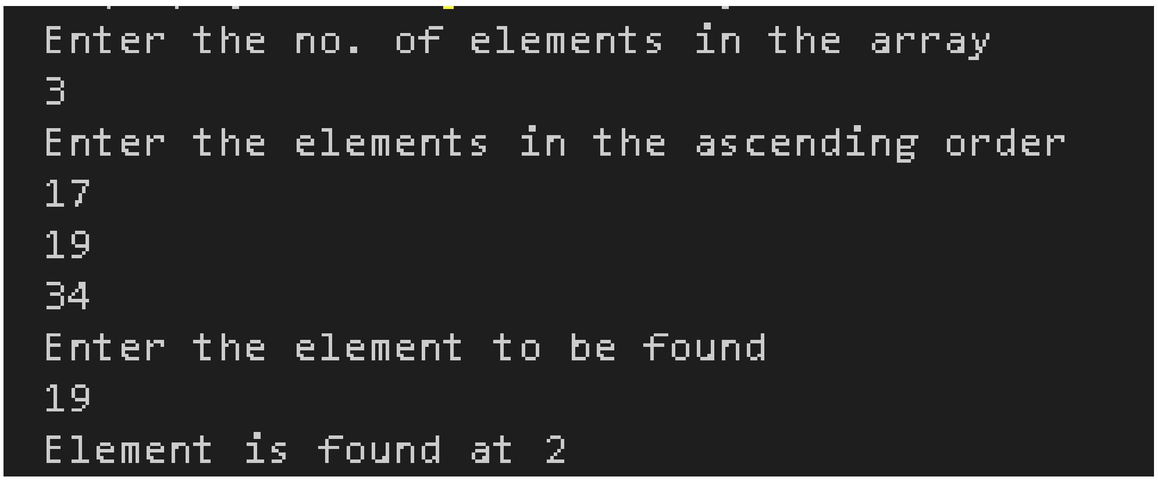
Polynomial Array

```
#include<stdio.h>
struct node {
  int coeff;
  int exp;
}poly1[10],poly2[10],sum[10];
void get_sum() {
  int i=0,j=0,k=0;
  while((poly1[i].coeff)!=0||(poly1[i].exp!=0)||(poly2[j].coeff)!=0||(poly2[j].exp!=0)) {
    if(poly1[i].exp==poly2[j].exp)  {
      sum[k].coeff=poly1[i].coeff+poly2[j].coeff;
      sum[k].exp=poly1[i].exp;
      i++;
      j++;
      k++; }
    else if(poly1[i].exp>poly2[j].exp) {
      sum[k].coeff=poly1[i].coeff;
      sum[k].exp=poly1[i].exp;
      i++;
      k++; }
    else {
      sum[k].coeff=poly2[j].coeff;
      sum[k].exp=poly2[j].exp;
      j++;
      k++; } } }
void display_polynomial(struct node poly[]) {
  printf("\n");
  for(int i=0;(poly[i].coeff)!=0||(poly[i].exp!=0);i++)  {
    if(poly[i].coeff!=0) {
      if(poly[i].exp!=0)
        printf("%dX^%d",poly[i].coeff,poly[i].exp);
        else
          printf("%d",poly[i].coeff);
        printf("\t"); } } }
int main() {
  char choice='y';
  int i;
  for(i=0;choice=='y';i++) {
    printf("Enter the value of coefficient of FIRST polynomial\n");
    scanf("%d",&poly1[i].coeff);
    printf("Enter the value of exponent of FIRST polynomial\n");
    scanf("%d",&poly1[i].exp);
    printf("continue? y/n\n");
    scanf(" %c",&choice);  }
  choice='y';
  for(i=0;choice=='y';i++) {
    printf("Enter the value of coefficient of SECOND polynomial\n");
    scanf("%d",&poly2[i].coeff);
    printf("Enter the value of exponent of SECOND polynomial\n");
    scanf("%d",&poly2[i].exp);
    printf("continue? y/n\n");
    scanf(" %c",&choice); }
  get_sum();
  display_polynomial(poly1);
  display_polynomial(poly2);
  display_polynomial(sum);
  return 0;
}
```

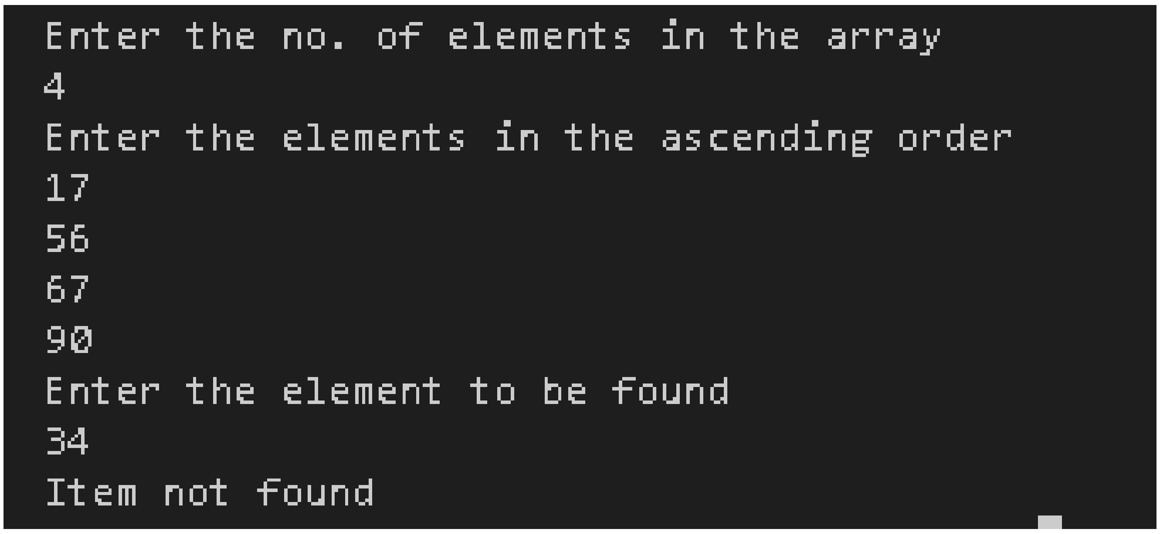
Hash Table

```
#include<stdio.h>
void display(int table[],int size){
  printf("Hash Table\n");
  for(int i=0;i<size;i++){
    if(table[i]==-1)
      printf("%d\tNULL\n",i);
    else
      printf("%d\t%d\n",i,table[i]);  } }
void insert(int table[],int size,int key){
  int hash_value,old_hash,i,flag=0;
  hash_value=key%size;
  if(table[hash_value]==-1)
    table[hash_value]=key;
  else {
    old_hash=hash_value;
    for(i=0;i<size;i++)  {
    hash_value=(old_hash+i)%size;
    if(table[hash_value]==-1){
      table[hash_value]=key;
      flag=1;
      break;  } }
    if(flag==0)
      printf("Hash Table is full");  } }
void search(int table[],int size,int key){
  int hash_value,old_hash,i,flag=0;
  hash_value=key%size;
  if(table[hash_value]==key)
    printf("Element found at postion %d\n",hash_value);
  else  {
    old_hash=hash_value;
```

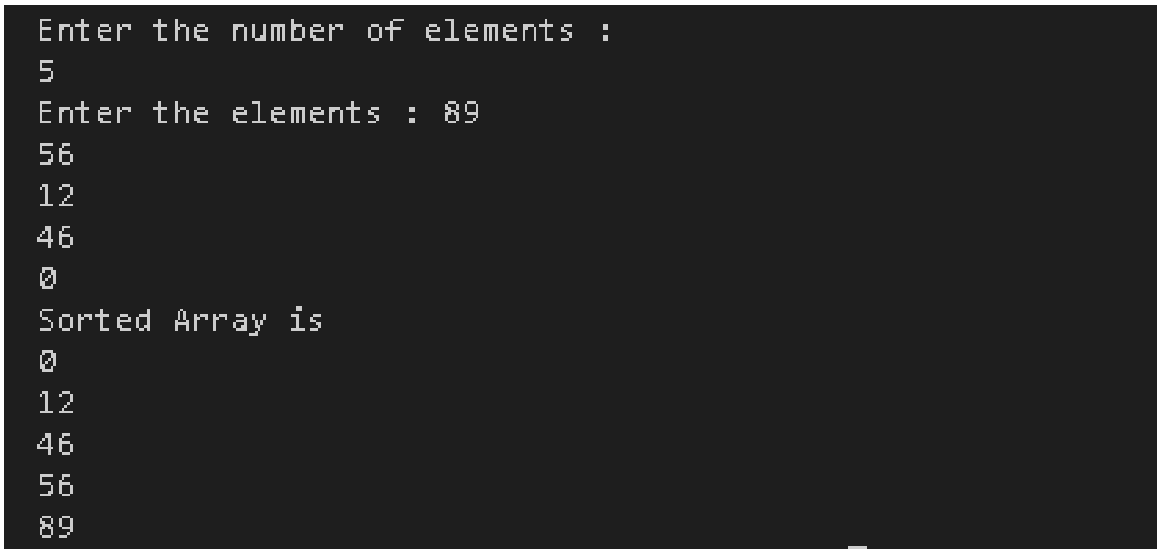
```
        for(i=0;i<size;i++) {
        hash_value=(old_hash+i)%size;
        if(table[hash_value]==key){
          printf("Element found at postion %d\n",hash_value);
          flag=1;
          break;  } }
        if(flag==0)
          printf("Element not found");    } }
int main() {
    int size,key;
    char choice;
    printf("Enter the size of the hash table : ");
    scanf("%d",&size);
    int hash_table[size];
    for(int i=0;i<size;i++)
        hash_table[i]=-1;
    do{
    printf("Enter the key to be inserted : ");
    scanf("%d",&key);
    insert(hash_table,size,key);
    printf("Continue? y/n\t");
    scanf(" %c",&choice);
    }while(choice=='y');
    printf("Enter the key to be searched : ");
    scanf("%d",&key);
    search(hash_table,size,key);
    display(hash_table,size);
}
```



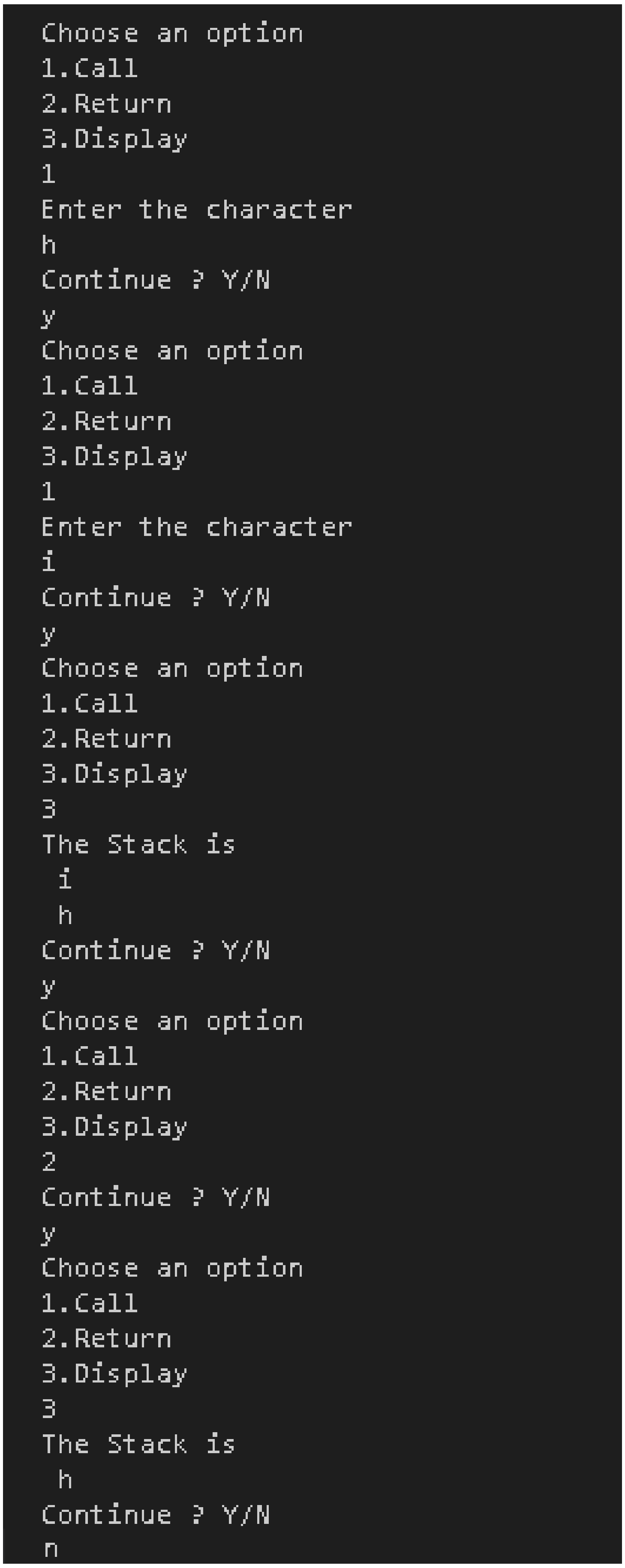
```
Enter the no. of elements in the array
3
Enter the elements in the ascending order
17
19
34
Enter the element to be found
19
Element is found at 2
```



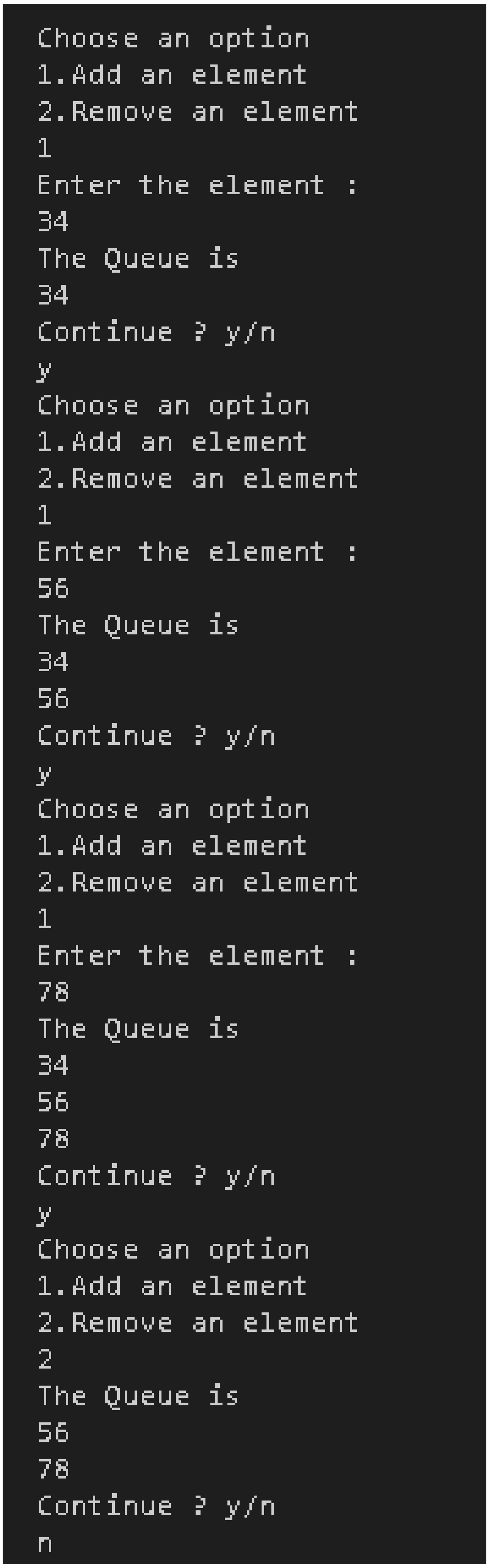
```
Enter the no. of elements in the array
4
Enter the elements in the ascending order
17
56
67
90
Enter the element to be found
34
Item not found
```



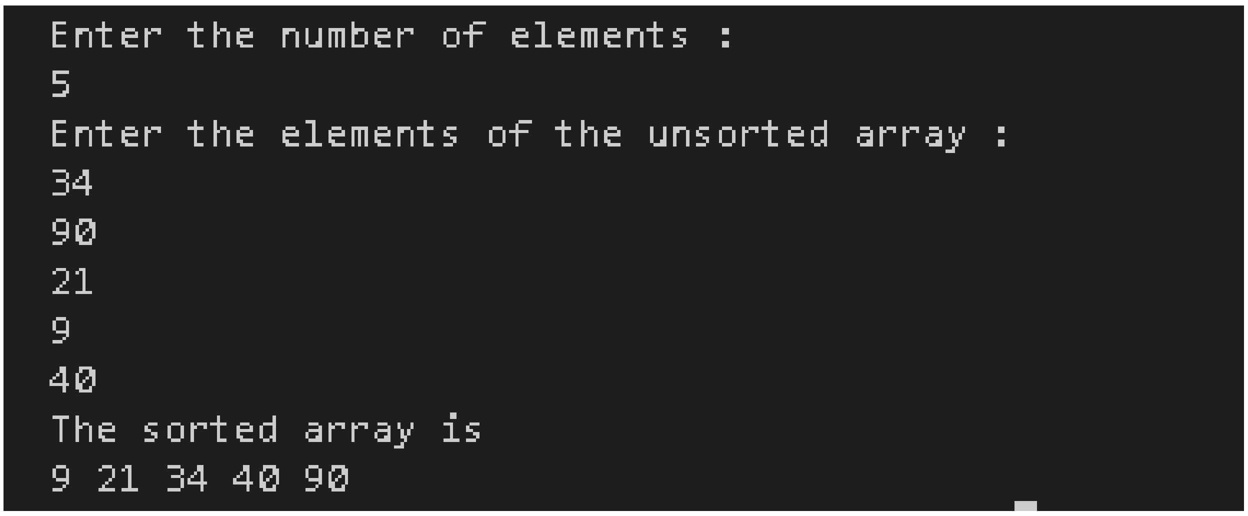
```
Enter the number of elements :
5
Enter the elements : 89
56
12
46
0
Sorted Array is
0
12
46
56
89
```



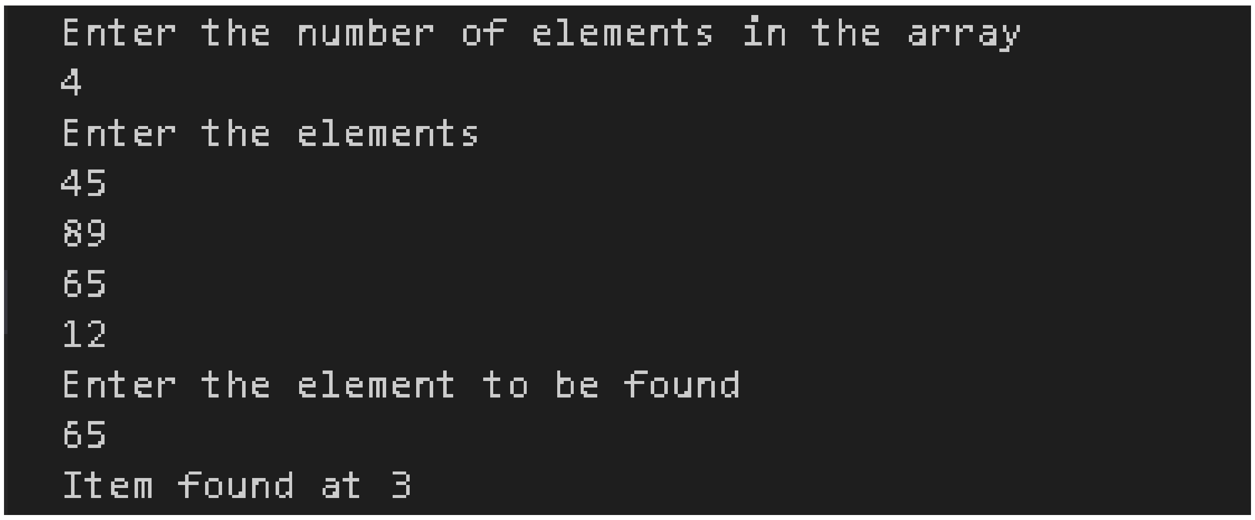
```
Choose an option
1.Call
2.Return
3.Display
1
Enter the character
h
Continue ? Y/N
y
Choose an option
1.Call
2.Return
3.Display
1
Enter the character
i
Continue ? Y/N
y
Choose an option
1.Call
2.Return
3.Display
3
The Stack is
 i
 h
Continue ? Y/N
y
Choose an option
1.Call
2.Return
3.Display
2
Continue ? Y/N
y
Choose an option
1.Call
2.Return
3.Display
3
The Stack is
 h
Continue ? Y/N
n
```



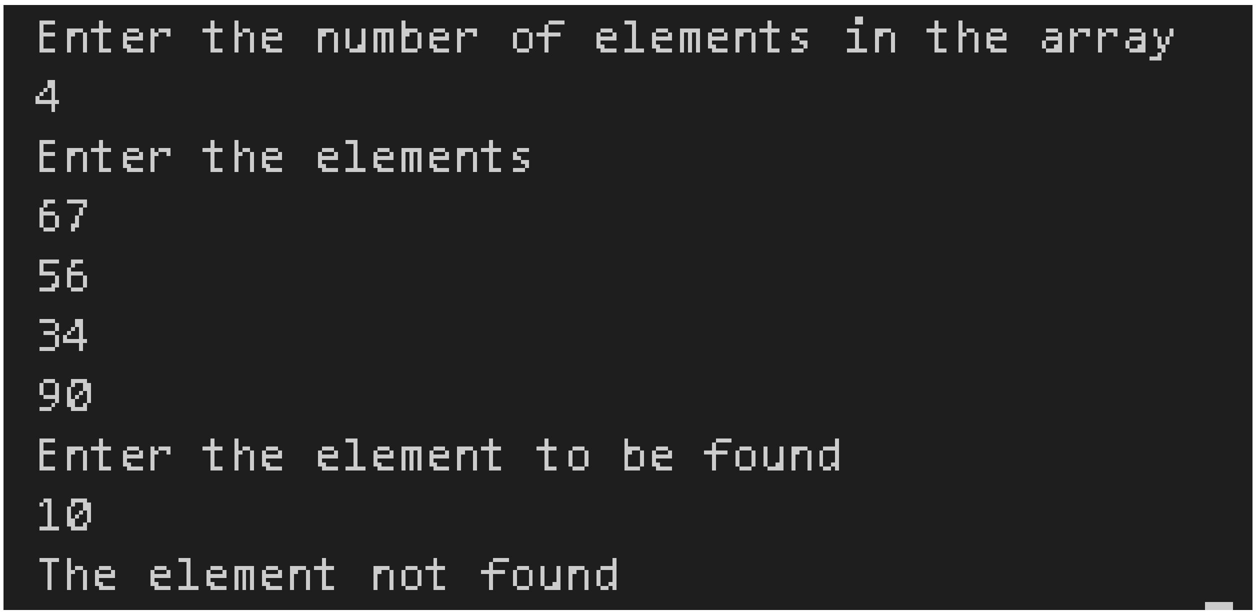
```
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
34
The Queue is
34
Continue ? y/n
y
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
56
The Queue is
34
56
Continue ? y/n
y
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
78
The Queue is
34
56
78
Continue ? y/n
y
Choose an option
1.Add an element
2.Remove an element
2
The Queue is
56
78
Continue ? y/n
n
```



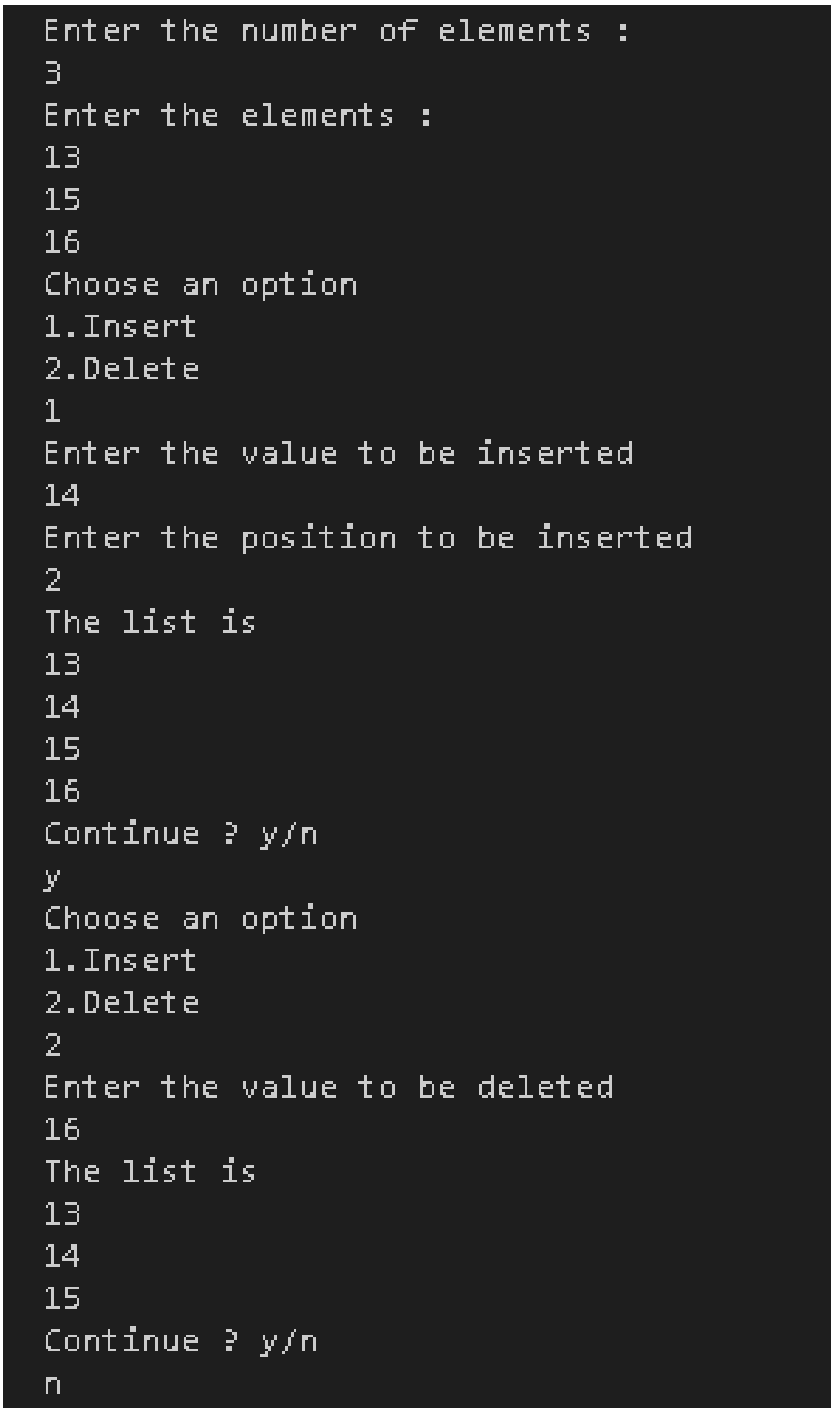
```
Enter the number of elements :
5
Enter the elements of the unsorted array :
34
90
21
9
40
The sorted array is
9 21 34 40 90
```

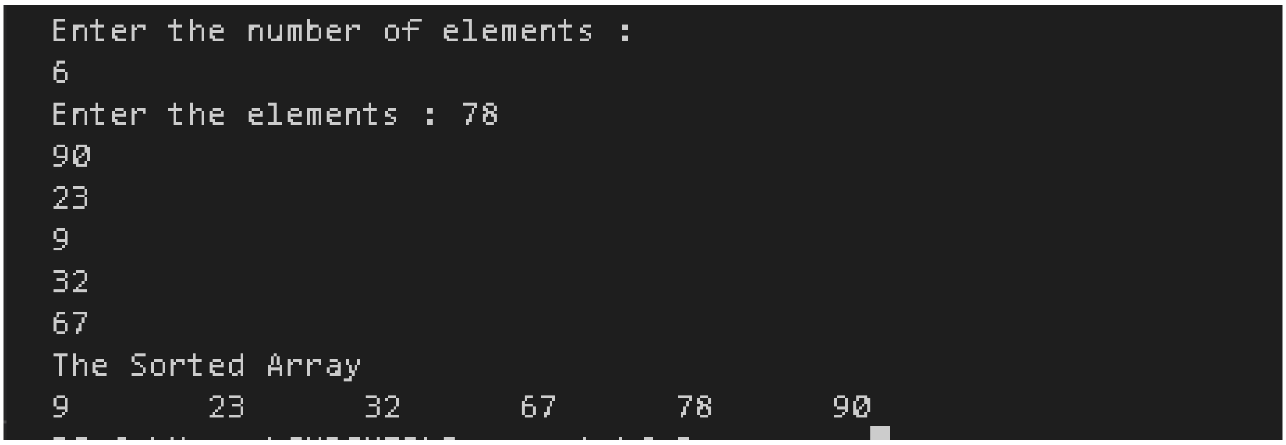


```
Enter the number of elements in the array
4
Enter the elements
45
89
65
12
Enter the element to be found
65
Item found at 3
```

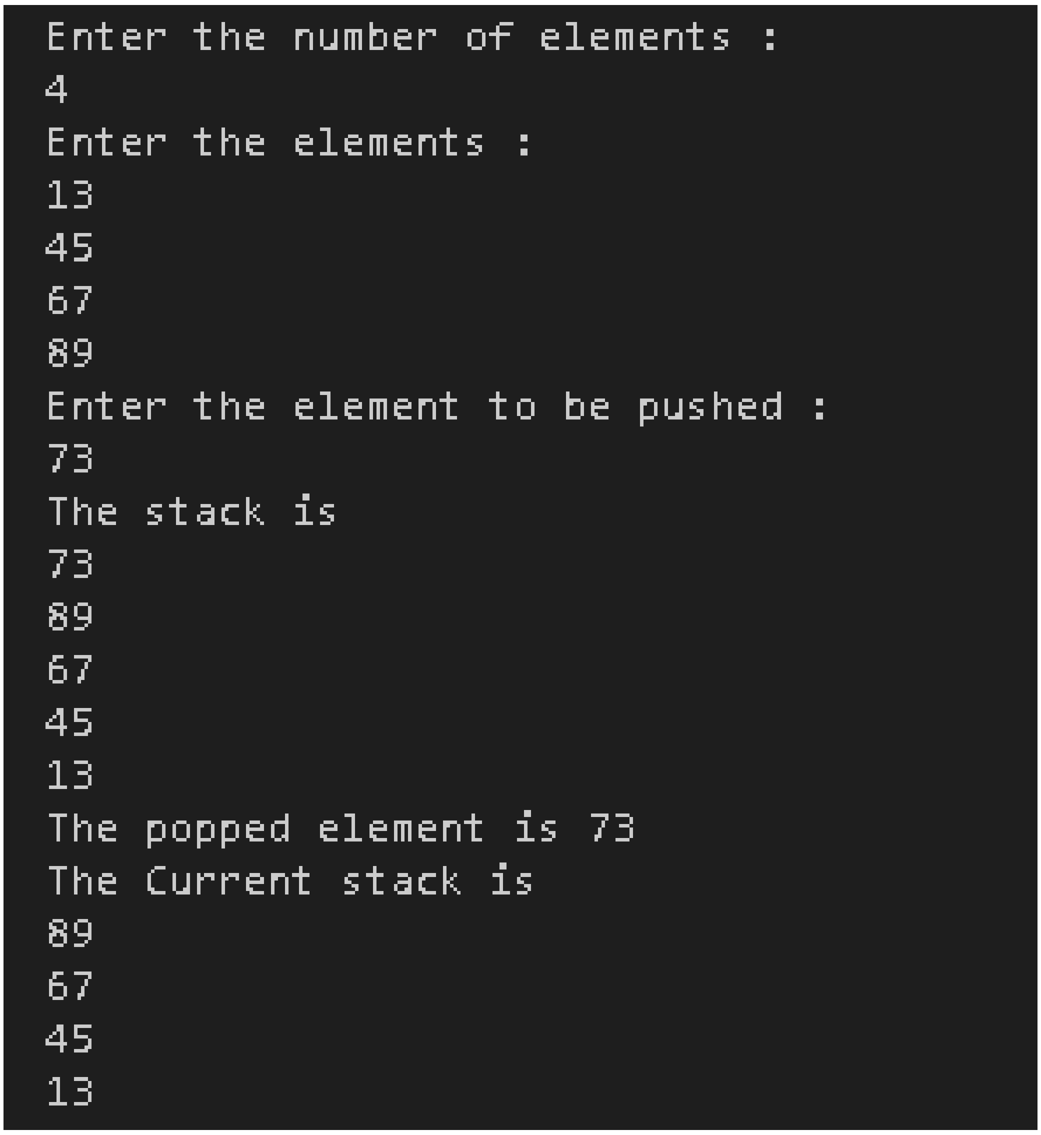


```
Enter the number of elements in the array
4
Enter the elements
67
56
34
90
Enter the element to be found
10
The element not found
```

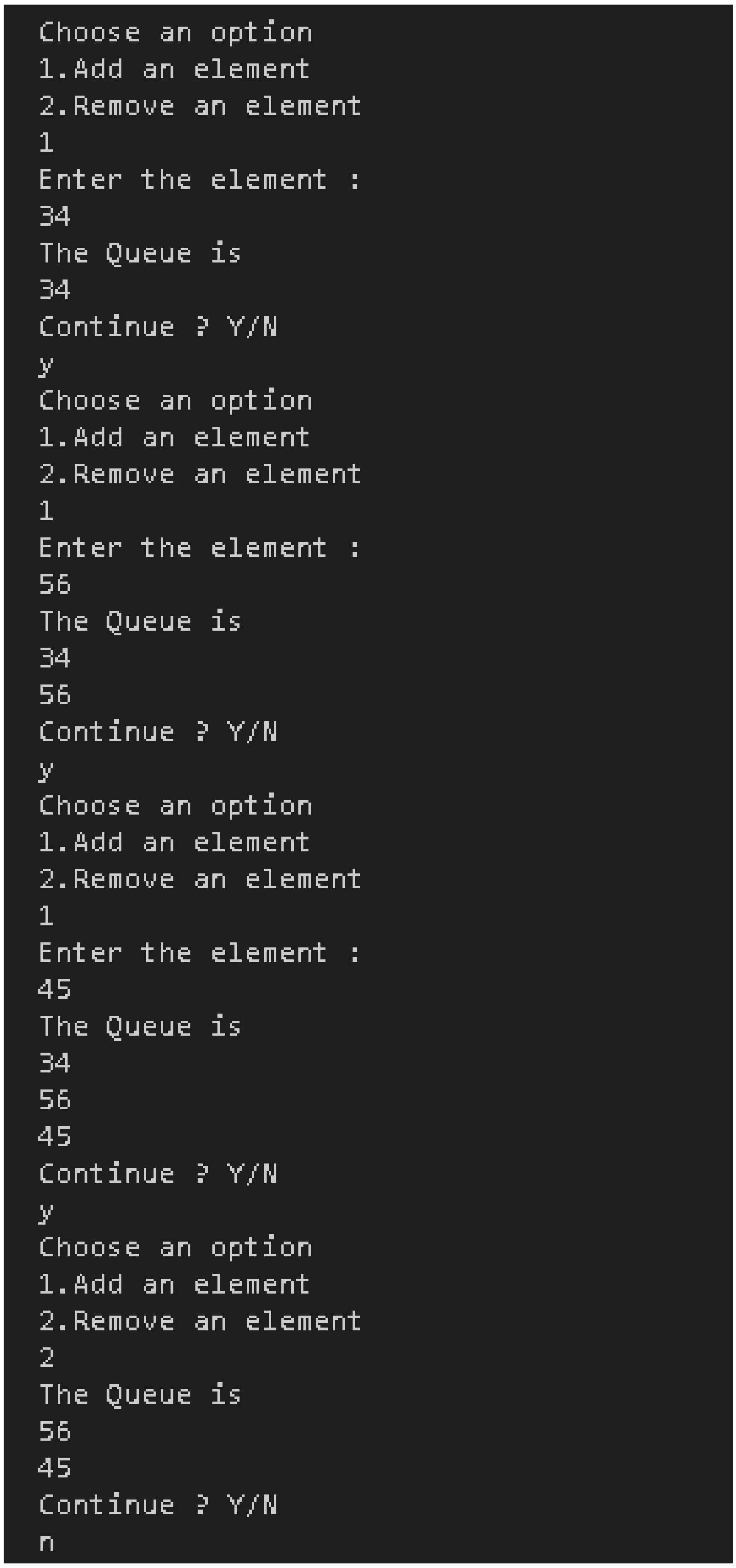

```
Enter the number of elements :
3
Enter the elements :
13
15
16
Choose an option
1.Insert
2.Delete
1
Enter the value to be inserted
14
Enter the position to be inserted
2
The list is
13
14
15
16
Continue ? y/n
y
Choose an option
1.Insert
2.Delete
2
Enter the value to be deleted
16
The list is
13
14
15
Continue ? y/n
n
```



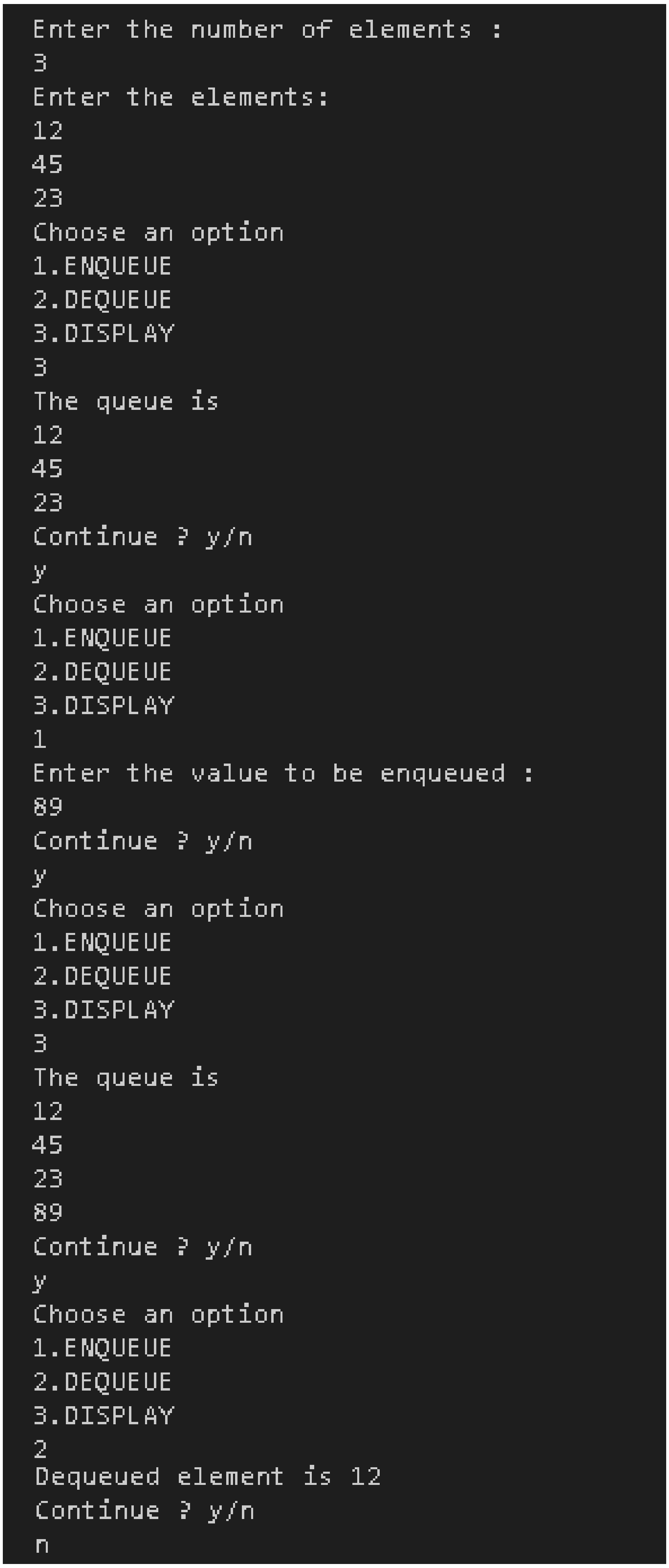
```
Enter the number of elements :
6
Enter the elements : 78
90
23
9
32
67
The Sorted Array
9       23      32      67      78      90
```

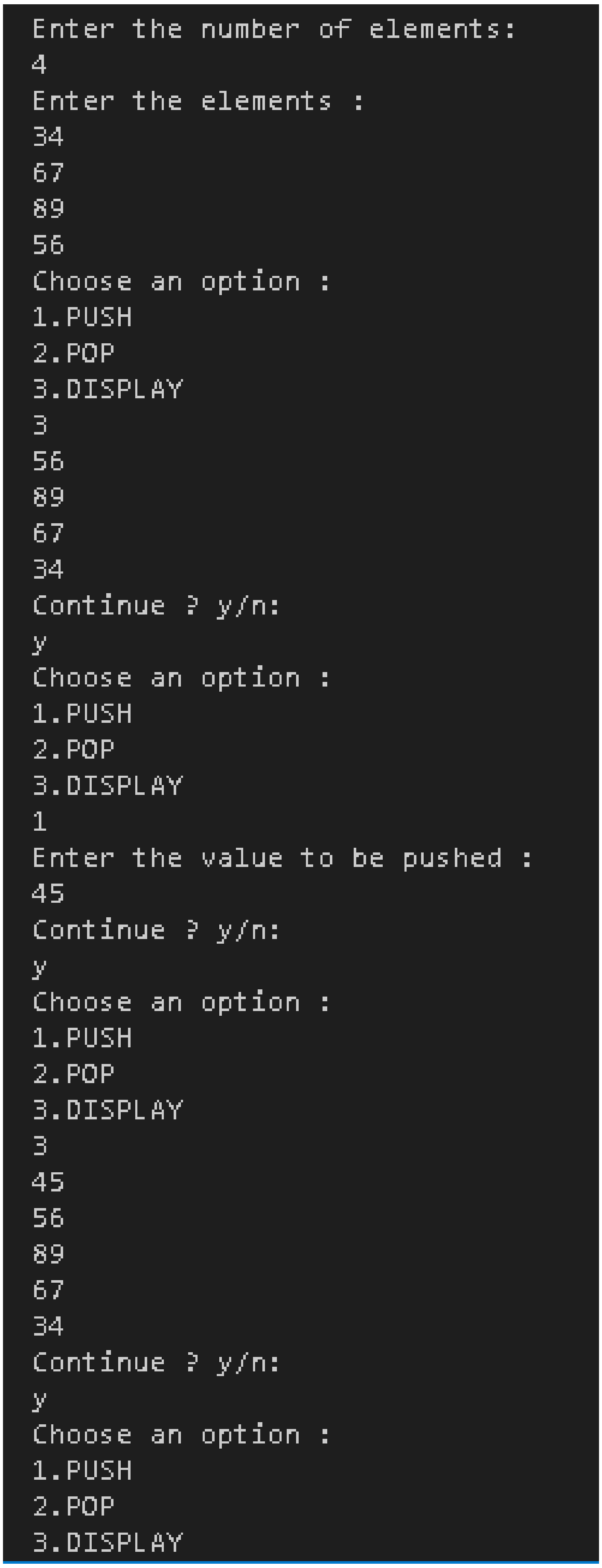


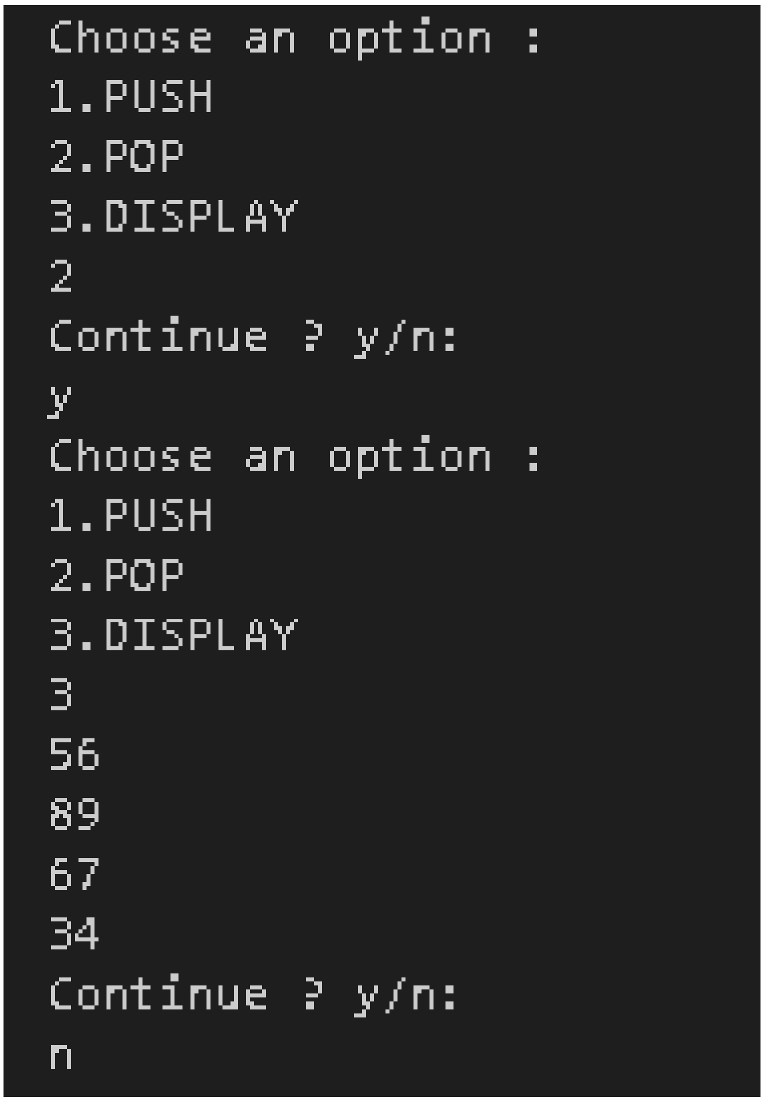
```
Enter the number of elements :
4
Enter the elements :
13
45
67
89
Enter the element to be pushed :
73
The stack is
73
89
67
45
13
The popped element is 73
The Current stack is
89
67
45
13
```



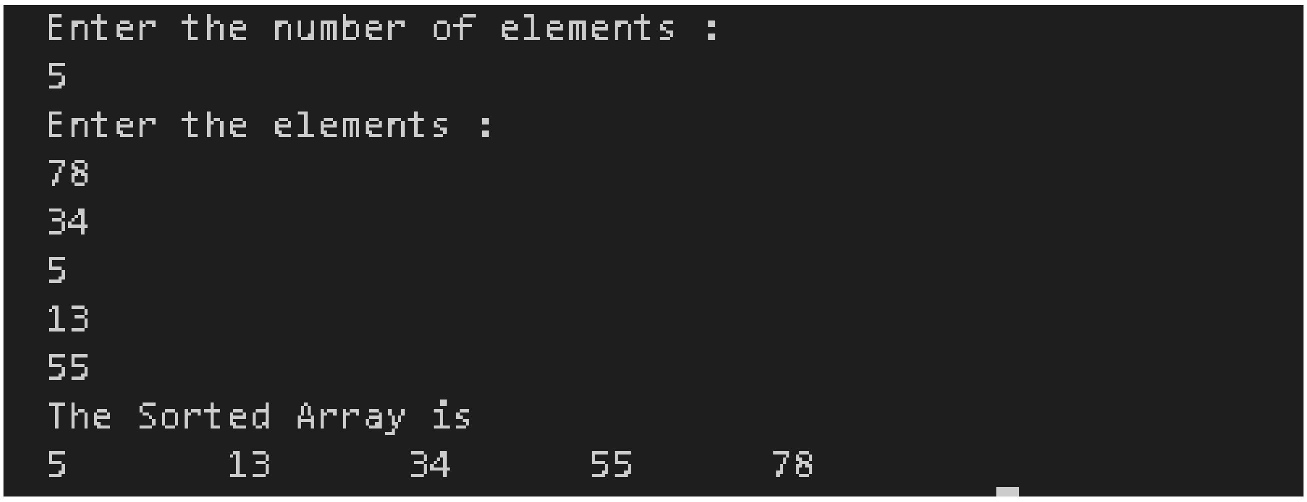
```
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
34
The Queue is
34
Continue ? Y/N
y
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
56
The Queue is
34
56
Continue ? Y/N
y
Choose an option
1.Add an element
2.Remove an element
1
Enter the element :
45
The Queue is
34
56
45
Continue ? Y/N
y
Choose an option
1.Add an element
2.Remove an element
2
The Queue is
56
45
Continue ? Y/N
n
```


```
Enter the number of elements :
3
Enter the elements:
12
45
23
Choose an option
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
3
The queue is
12
45
23
Continue ? y/n
y
Choose an option
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
1
Enter the value to be enqueued :
89
Continue ? y/n
y
Choose an option
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
3
The queue is
12
45
23
89
Continue ? y/n
y
Choose an option
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
2
Dequeued element is 12
Continue ? y/n
n
```

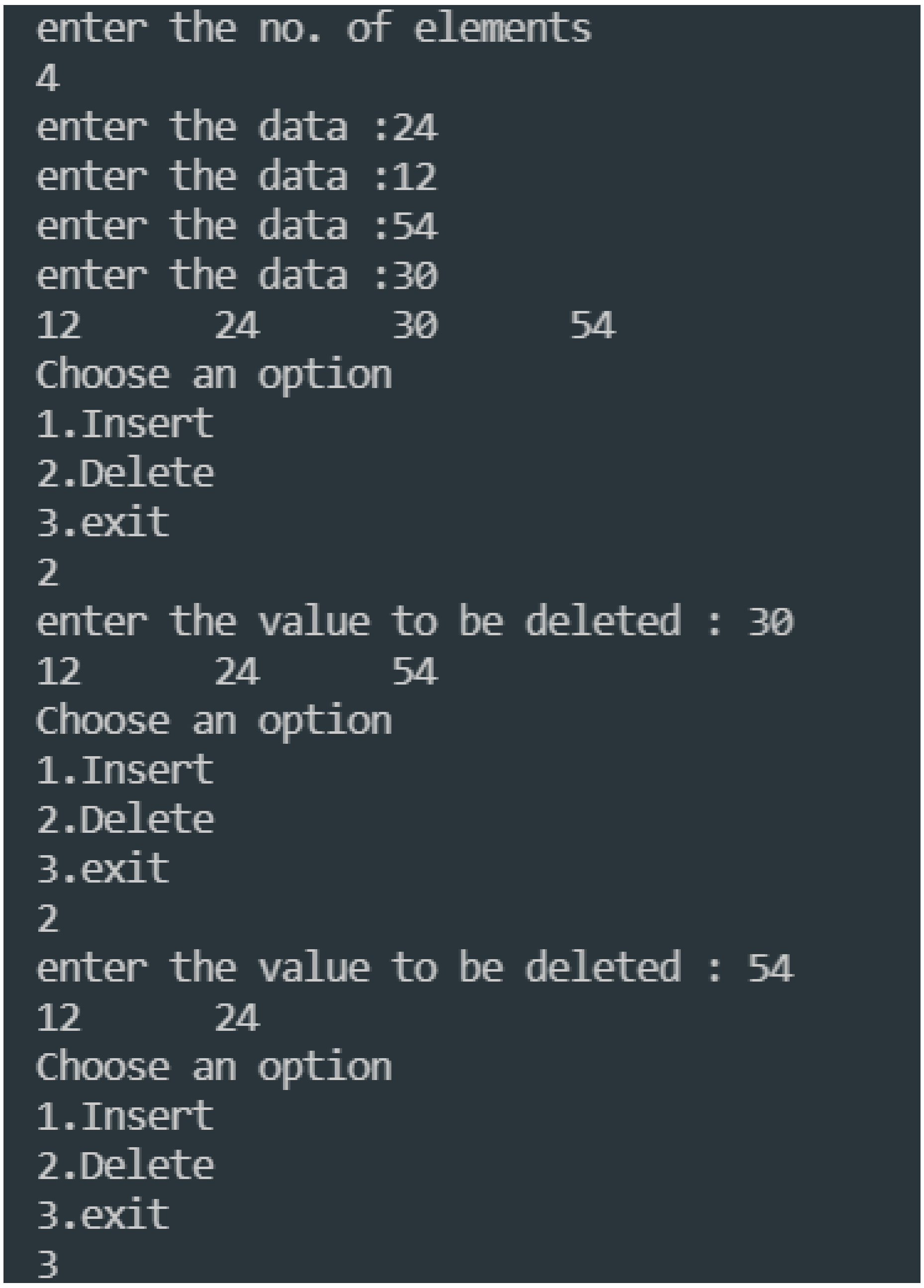

```
Enter the number of elements:
4
Enter the elements :
34
67
89
56
Choose an option :
1.PUSH
2.POP
3.DISPLAY
3
56
89
67
34
Continue ? y/n:
y
Choose an option :
1.PUSH
2.POP
3.DISPLAY
1
Enter the value to be pushed :
45
Continue ? y/n:
y
Choose an option :
1.PUSH
2.POP
3.DISPLAY
3
45
56
89
67
34
Continue ? y/n:
y
Choose an option :
1.PUSH
2.POP
3.DISPLAY
```



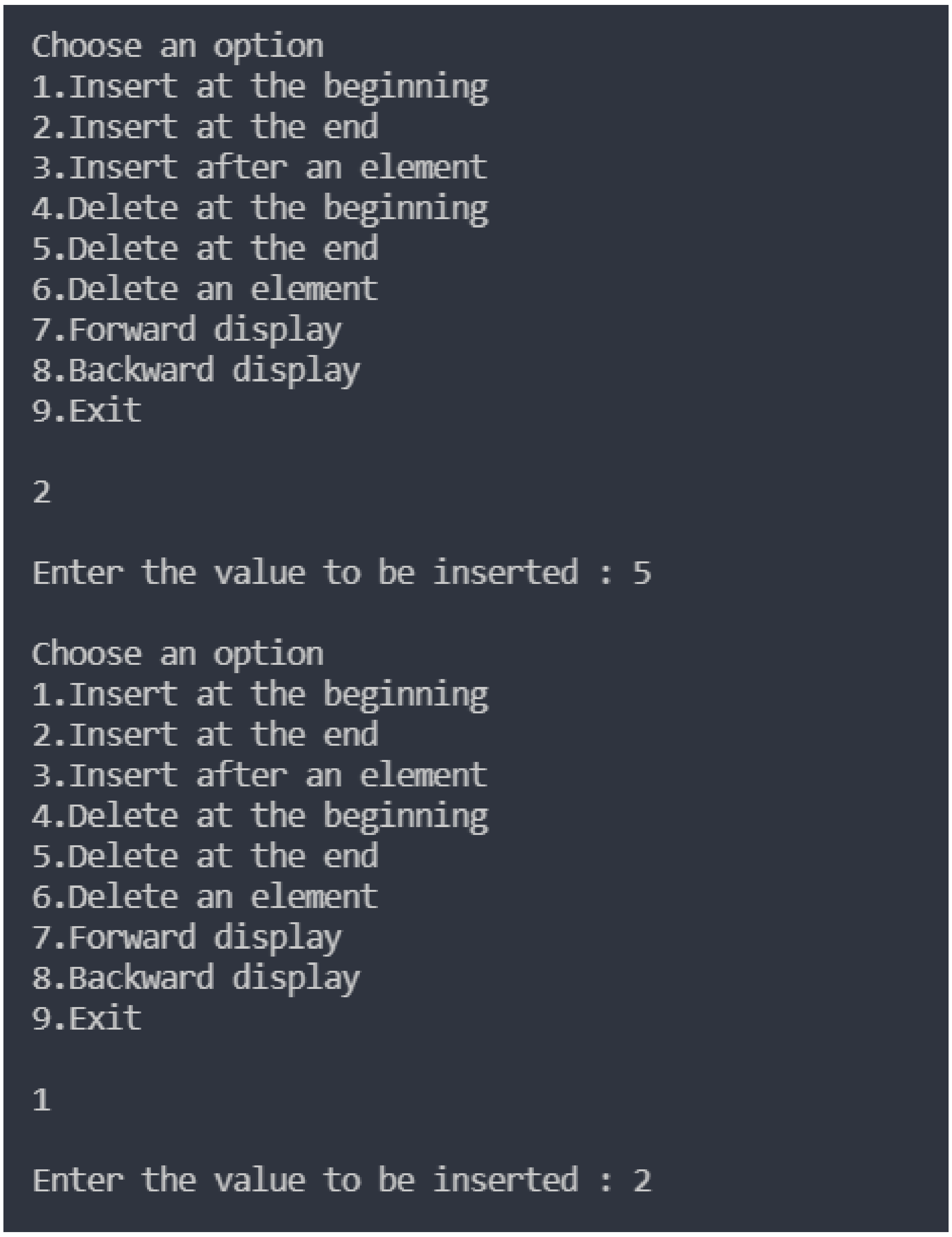
```
Choose an option :
1.PUSH
2.POP
3.DISPLAY
2
Continue ? y/n:
y
Choose an option :
1.PUSH
2.POP
3.DISPLAY
3
56
89
67
34
Continue ? y/n:
n
```



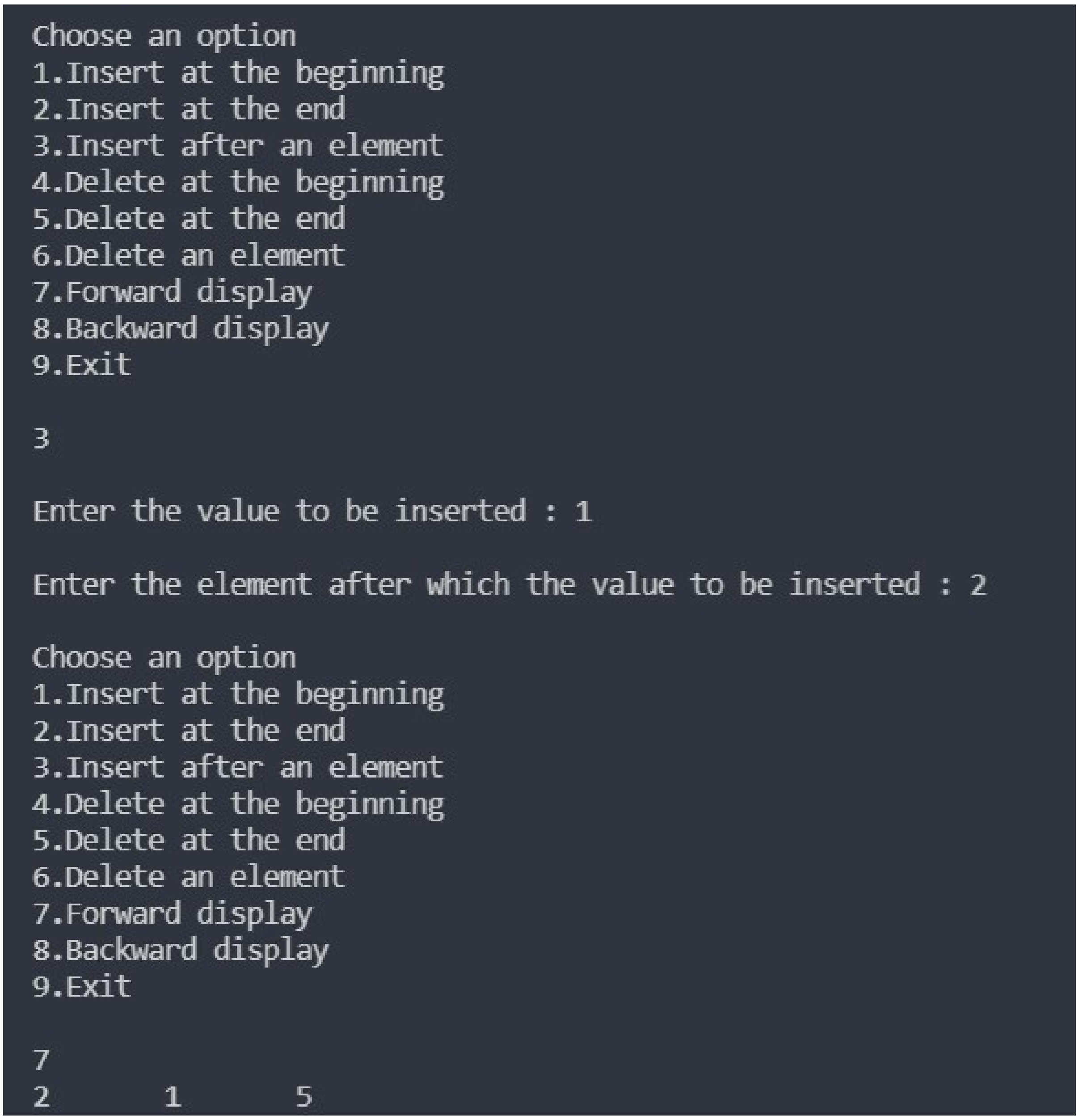
```
Enter the number of elements :
5
Enter the elements :
78
34
5
13
55
The Sorted Array is
5       13      34      55      78
```



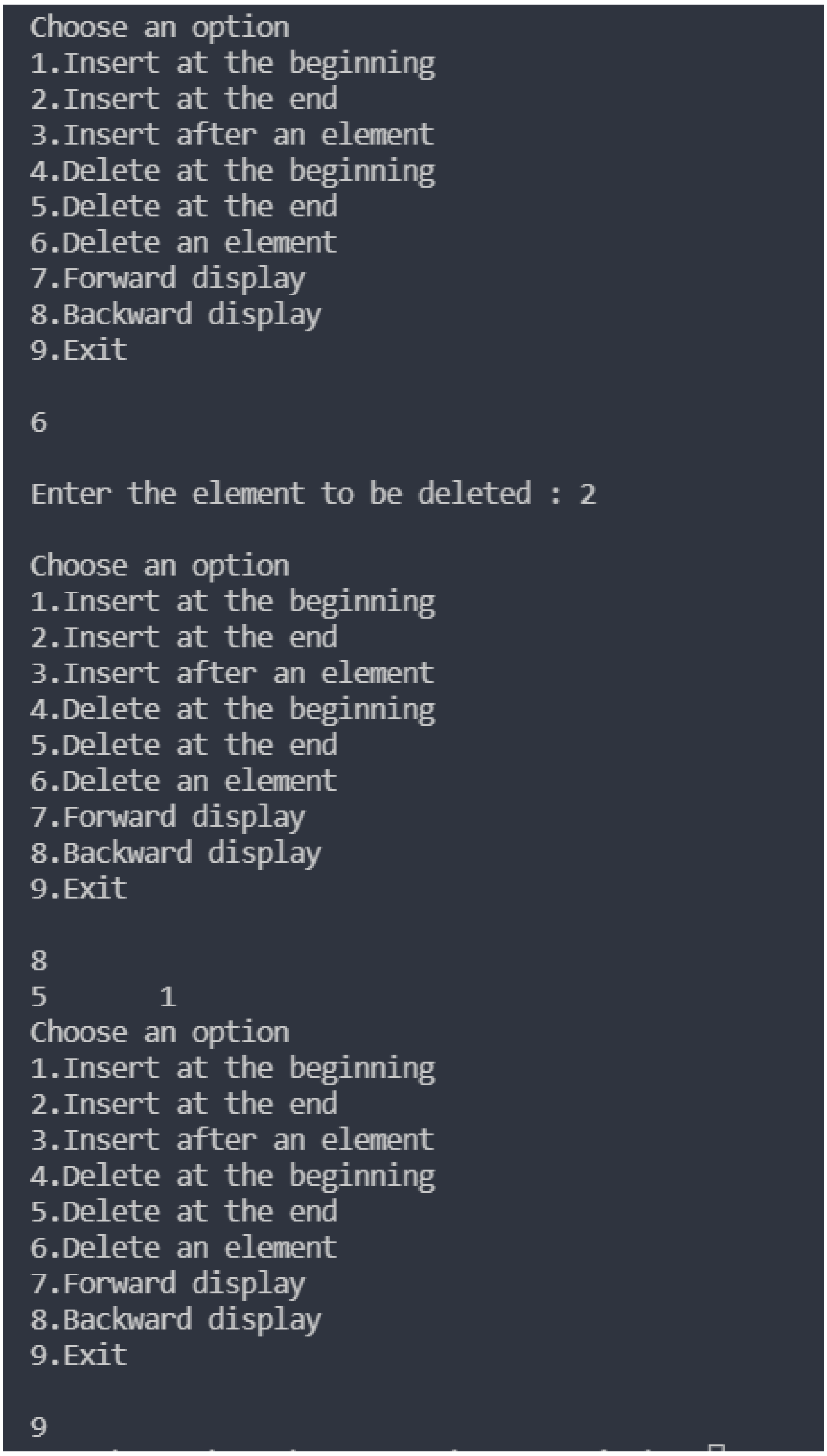
```
enter the no. of elements
4
enter the data :24
enter the data :12
enter the data :54
enter the data :30
12      24      30      54
Choose an option
1.Insert
2.Delete
3.exit
2
enter the value to be deleted : 30
12      24      54
Choose an option
1.Insert
2.Delete
3.exit
2
enter the value to be deleted : 54
12      24
Choose an option
1.Insert
2.Delete
3.exit
3
```



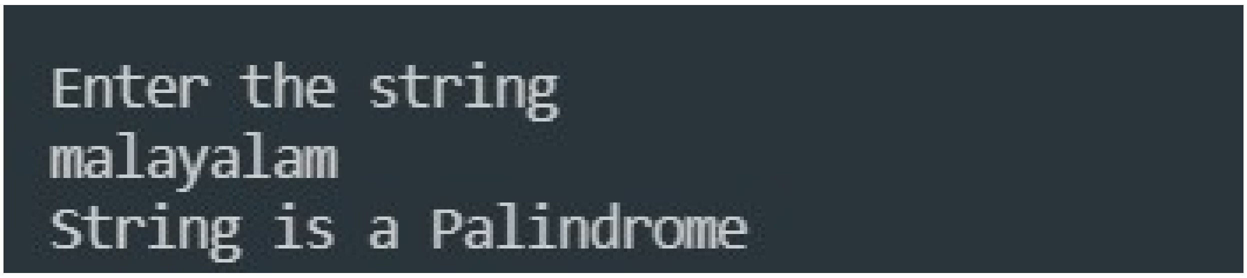
```
Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

2

Enter the value to be inserted : 5

Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

1

Enter the value to be inserted : 2
```



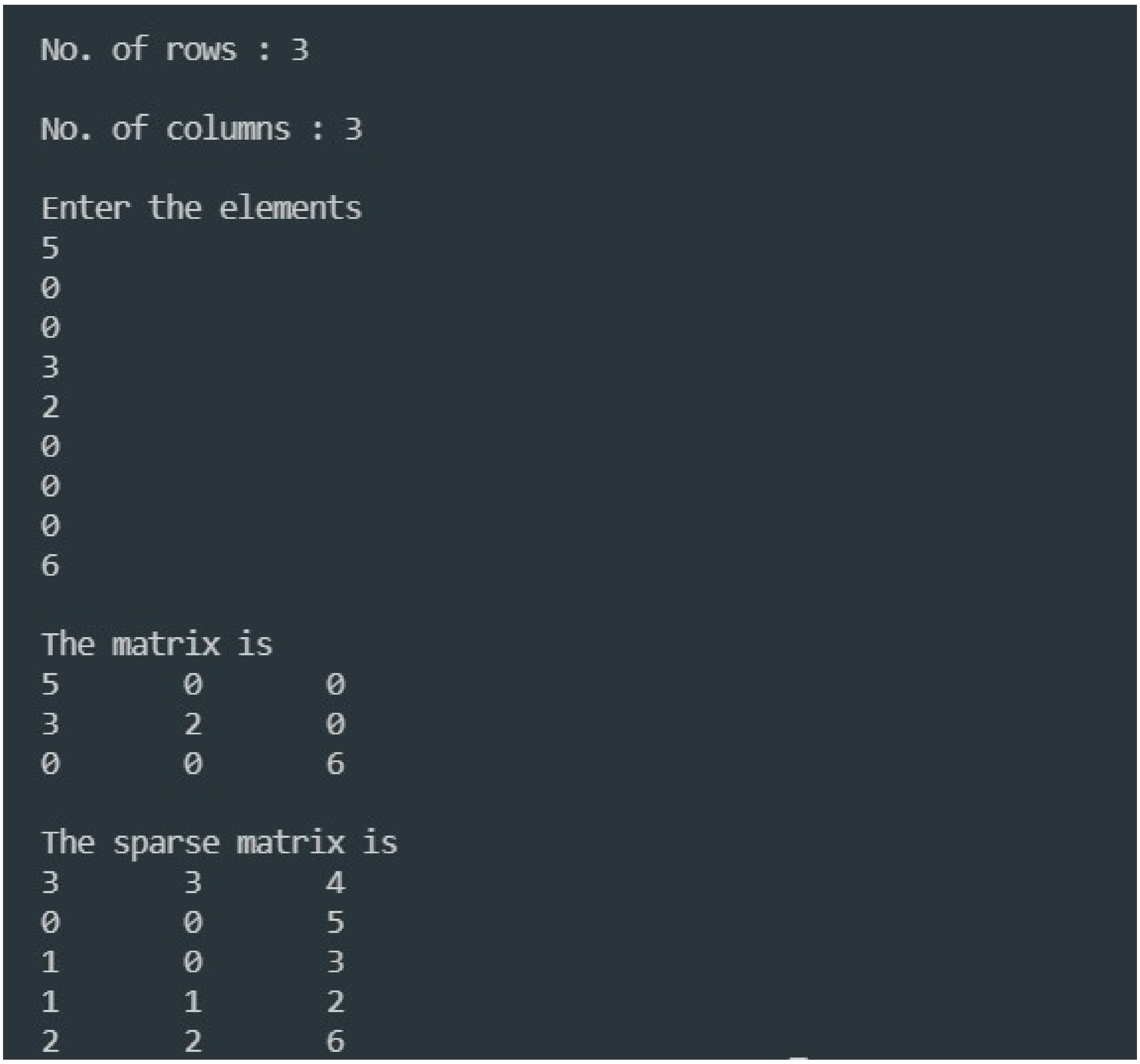
```
Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

3

Enter the value to be inserted : 1

Enter the element after which the value to be inserted : 2

Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

7
2       1       5
```



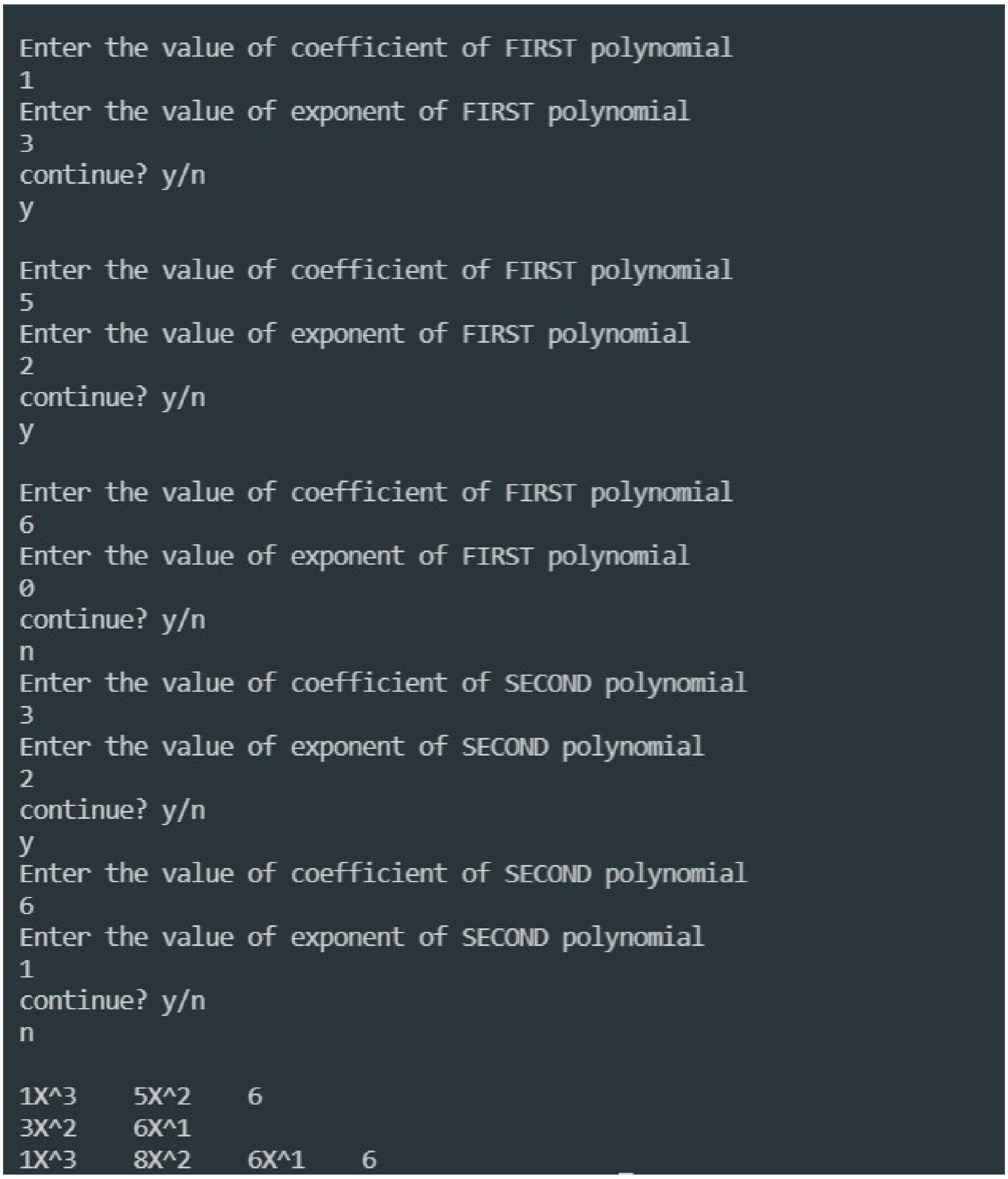
```
Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

6

Enter the element to be deleted : 2

Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

8
5       1
Choose an option
1.Insert at the beginning
2.Insert at the end
3.Insert after an element
4.Delete at the beginning
5.Delete at the end
6.Delete an element
7.Forward display
8.Backward display
9.Exit

9
```



```
Enter the string
malayalam
String is a Palindrome
```

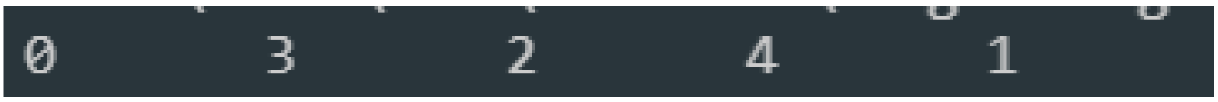


```
No. of rows : 3

No. of columns : 3

Enter the elements
5
0
0
3
2
0
0
0
6

The matrix is
5       0       0
3       2       0
0       0       6

The sparse matrix is
3       3       4
0       0       5
1       0       3
1       1       2
2       2       6
```

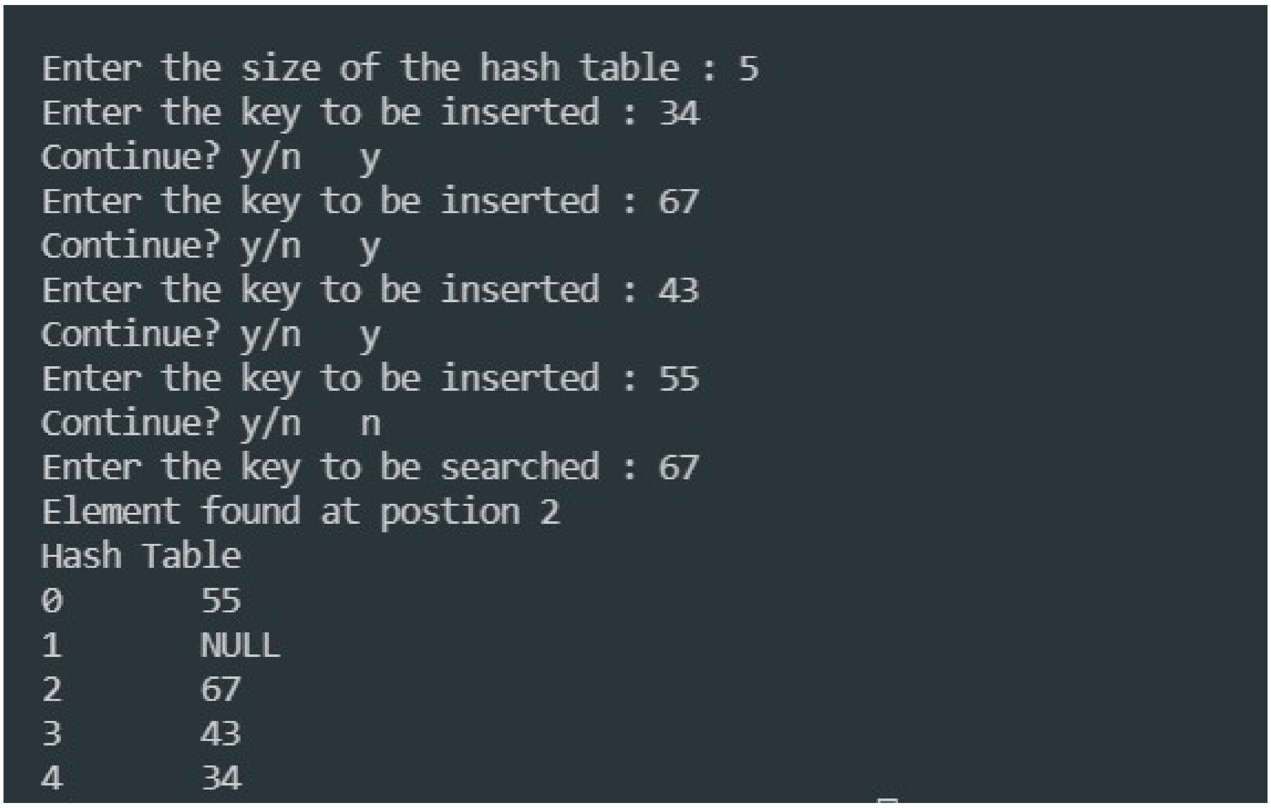


```
Enter the value of coefficient of FIRST polynomial
1
Enter the value of exponent of FIRST polynomial
3
continue? y/n
y

Enter the value of coefficient of FIRST polynomial
5
Enter the value of exponent of FIRST polynomial
2
continue? y/n
y

Enter the value of coefficient of FIRST polynomial
6
Enter the value of exponent of FIRST polynomial
0
continue? y/n
n
Enter the value of coefficient of SECOND polynomial
3
Enter the value of exponent of SECOND polynomial
2
continue? y/n
y
Enter the value of coefficient of SECOND polynomial
6
Enter the value of exponent of SECOND polynomial
1
continue? y/n
n

1X^3    5X^2    6
3X^2    6X^1
1X^3    8X^2    6X^1    6
```

```
 0      1      2      3      4
```



```
0         3         2         4         1
```



```
Enter the size of the hash table : 5
Enter the key to be inserted : 34
Continue? y/n   y
Enter the key to be inserted : 67
Continue? y/n   y
Enter the key to be inserted : 43
Continue? y/n   y
Enter the key to be inserted : 55
Continue? y/n   n
Enter the key to be searched : 67
Element found at postion 2
Hash Table
0       55
1       NULL
2       67
3       43
4       34
```

Inherit

```
import java.util.*;
class Employee{
   String name;
   int age;
   long phone;
   String address;
   int salary;
   void printSalary(){
      System.out.println("Salary : "+salary);
   }
}
class Officer extends Employee{
   String special;
}
class Manager extends Employee{
   String depart;
}
class Inherit{
   public static void main(String args[]){
      Scanner in = new Scanner(System.in);
      Scanner ins = new Scanner(System.in);
      Officer obj1= new Officer();
      Manager obj2= new Manager();
      System.out.print("Enter the name of officer : ");
      obj1.name=ins.nextLine();
      System.out.print("Enter the age : ");
      obj1.age=in.nextInt();
      System.out.print("Enter the phone number : ");
      obj1.phone=in.nextLong();
      System.out.print("Enter the address : ");
```

```
        obj1.address=ins.nextLine();
        System.out.print("Enter the salary : ");
        obj1.salary=in.nextInt();
        System.out.print("Enter the specialization : ");
        obj1.special=ins.nextLine();
        obj1.printSalary();
        System.out.print("Enter the name of Manager : ");
        obj2.name=ins.nextLine();
        System.out.print("Enter the age : ");
        obj2.age=in.nextInt();
        System.out.print("Enter the phone number : ");
        obj2.phone=in.nextLong();
        System.out.print("Enter the address : ");
        obj2.address=ins.nextLine();
        System.out.print("Enter the salary : ");
        obj2.salary=in.nextInt();
        System.out.print("Enter the department : ");
        obj2.depart=ins.nextLine();
        obj1.printSalary();
    }
}
```

OUTPUT

```
Enter the name of officer : Milind Roy
Enter the age : 35
Enter the phone number : 8976453429
Enter the address : Amardeep Apt, Vadodara, Gujrat
Enter the salary : 30000
Enter the specialization : Artificial Intelligence
Salary : 30000
Enter the name of Manager : Silvie Mahdal
Enter the age : 40
Enter the phone number : 9845624927
Enter the address : Love Shore,Bangalore, Karnataka
Enter the salary : 47000
Enter the department : Computer Science
Salary : 47000
```

Exception Handling

```
import java.util.*;
class Excep{
  static void method() throws IllegalAccessException{
    System.out.println("Throws");
    throw new IllegalAccessException("throw demo");}
  public static void main(String args[]){
    try{
      method();
    }
    catch(IllegalAccessException e){
      System.out.println("IllegalAccessException"+e);
    }
    try{
      int a[]=new int[2];
      a[2]=3/0;
    }
    catch(ArithmeticException e){
      System.out.println("ArithmeticException"+e);
    }
    catch(ArrayIndexOutOfBoundsException e){
      System.out.println("ArrayIndexOutOfBoundsException"+e);
    }
    catch(Exception e){
      System.out.println("Parent Exception Occurs");
    }
    finally{
      System.out.println("This block will be printed");}
    try{
      try{
        System.out.println("First inner try");
```

```
          int b=3/0;
        }
        catch(ArithmeticException e){
          System.out.println("Arithmetic Exception");}
        try{
          System.out.println("Second inner try");
          int d[]=new int[3];
          d[3]=0; }
        catch(ArrayIndexOutOfBoundsException e){
          System.out.println("ArrayIndexOutOfBoundsException Occurs");
        }}
      catch(Exception e){
        System.out.println("Outer catch");
      }
      finally{
        System.out.println("Second finally");
      }}}
```

OUTPUT

```
Throws
IllegalAccessExceptionjava.lang.IllegalAccessException: throw demo
ArithmeticExceptionjava.lang.ArithmeticException: / by zero
This block will be printed
First inner try
Arithmetic Exception
Second inner try
ArrayIndexOutOfBoundsException Occurs
Second finally
```

Matrix Multiplication

```
import java.util.*;
public class Matrix {
  public static void main(String[] args){
    Scanner s1=new Scanner(System.in);
    int a[][]=new int[10][10];
    int b[][]=new int[10][10];
    int c[][]=new int[10][10];
    int m,n,p,q,i,j,k;
    System.out.println("Enter the size of first matrix : ");
    m=s1.nextInt();
    n=s1.nextInt();
    System.out.println("Enter the size of second matrix : ");
    p=s1.nextInt();
    q=s1.nextInt();
    if(n==p){
      System.out.println("Enter the elements into first matrix : ");
      for(i=0;i<m;i++){
        for(j=0;j<n;j++){
          a[i][j]=s1.nextInt();}}
      System.out.println("Enter the elements of second matrix : ");
      for(i=0;i<p;i++){
        for(j=0;j<q;j++){
          b[i][j]=s1.nextInt();}}
      for(i=0;i<m;i++){
        for(j=0;j<q;j++){
          c[i][j]=0;
          for(k=0;k<n;k++){
            c[i][j]=c[i][j]+(a[i][k]*b[k][j]); }}}
      System.out.println("Matrix after multiplication ");
      for(i=0;i<m;i++){
            System.out.println();
            for(j=0;j<q;j++){
               System.out.println(c[i][j]+" ");}}}
      else{
         System.out.println("Matrix multiplication not possible");}}
}
```

OUTPUT 1

```
Enter the size of first matrix :
2
3
Enter the size of second matrix :
3
3
Enter the elements into first matrix :
1
2
3
4
5
6
Enter the elements of second matrix :
7
4
3
6
7
3
6
8
3
```

```
Matrix after multiplication


37

42

18


94

99

45
```

OUTPUT 2

```
Enter the size of first matrix :

2

3

Enter the size of second matrix :

2

2

Matrix multiplication not possible
```

Palindrome

```
import java.io.*;
import java.util.*;
public class Palindrome {
  public static void main(String args[]){
    String rev="";
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the string : ");
    String st=sc.nextLine();
    int len=st.length();
    for(int i=0;i<len;i++){
      rev=st.charAt(i)+rev;}
    if(st.equals(rev)){
      System.out.println("Palindrome");}
    else{
      System.out.println("Not Palindrome");
    }}
}
```

OUTPUT 1

```
Enter the string :
madam
Palindrome
```

OUTPUT 2

```
Enter the string :
english
Not Palindrome
```

Frequency of a given character in a string

```
import java.io.*;
import java.util.*;
public class Repeat {
  public static void main(String args[]){
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a string : ");
    String st=sc.nextLine();
    int len=st.length();
    int sum=0;
    Scanner sch=new Scanner(System.in);
    System.out.println("Enter a character : ");
    char ct=sch.next().charAt(0);
    for(int i=0;i<len;i++){
      if(st.charAt(i)==ct){
        sum++;
      }}
    System.out.println("Frequency is ");
    System.out.println(sum);
  }
}
```

OUTPUT

```
Enter a string :
malayalam
Enter a character :
a
Frequency is
4
```

Binary Search

```
import java.util.*;
import javax.lang.model.util.ElementScanner14;
public class Binary {
  public static void main(String args[]){
    int n,a,flag=0,mid,low=0,i;
    Scanner in = new Scanner(System.in);
    System.out.println("Enter the number of elements : ");
    n=in.nextInt();
    int A[]=new int[50];
    int high=n;
    System.out.println("Enter the elements : ");
    for(i=0;i<n;i++){
      A[i]=in.nextInt();
    }
    System.out.println("Enter the number to be searched : ");
    a=in.nextInt();
    while(low<=high){
      mid=(low+high)/2;
      if(A[mid]==a){
        System.out.println("The element found at position : "+(mid+1));
        flag++;
        break;}
      else if(a>A[mid]){
        low=mid+1;}
      else
      high=mid-1;
    }
    if(flag==0)
    System.out.println("Element abort");
  }}
```

OUTPUT 1

```
Enter the number of elements :
5
Enter the elements :
12
34
55
78
34
Enter the number to be searched :
55
The element found at position : 3
```

OUTPUT 2

```
Enter the number of elements :
5
Enter the elements :
12
34
55
78
34
Enter the number to be searched :
10
Element abort
```

Abstraction

```
abstract class Shape{
  abstract void numberOfSides();
}
class Triangle extends Shape{
  void numberOfSides(){
    System.out.println("No.of Sides of Triangle= 3");
  }}
class Rectangle extends Shape{
  void numberOfSides(){
    System.out.println("No.of Sides of Rectangle = 4");
  }}
class Hexagon extends Shape{
  void numberOfSides(){
    System.out.println("No.of Sides of Hexagon = 6");
  }}
class abs{
  public static void main(String args[]){
    Triangle tri =new Triangle();
    Rectangle rec =new Rectangle();
    Hexagon hex =new Hexagon();
    tri.numberOfSides();
    rec.numberOfSides();
    hex.numberOfSides();
  }
}
```

OUTPUT

```
No.of Sides of Triangle= 3
No.of Sides of Rectangle = 4
No.of Sides of Hexagon = 6
```

Garbage Collection

```
public class garbage {
  int value;
  public void finalize(){
    System.out.println();
    System.out.println("The object is garbage collected");
  }
  public static void main(String args[]){
    System.out.println();
    garbage s1=new garbage();
    System.out.println("object 1 is created");
    garbage s2=new garbage();
    System.out.println("object 2 is created");
    garbage s3=new garbage();
    System.out.println("object 3 is created");
    new garbage();
    System.out.println("anonymous object is created");
    s1=null;
    s2=s3;
    System.gc();
  }
}
```

OUTPUT

```
object 1 is created
object 2 is created
object 3 is created
anonymous object is created
```

File Read Write

```
import java.io.*;
class FileRW{
 public static void main(String args[]) throws IOException {
  FileReader src = null;
  FileWriter des = null;
  FileWriter deslen = null;
  try {
   src = new FileReader("source.txt");
   des = new FileWriter("string.txt");
   int len=0,temp;
   while ((temp = src.read()) != -1) {
    des.write(temp);
    ++len;}
   des.write(" "+len+"");
   System.out.println("Successfully copied the contents of the given file to another
file character wise on the same location");
  }
  finally {
   if (src != null) {
    src.close();}
   if (des != null) {
    des.close();}
   if(deslen !=null){
     deslen.close();
   }}}
}
```

OUTPUT

```
Successfully copied the contents of the given file to another file character wise on
the same location
```

File Reverse

```
import java.io.*;
public class FileReverse {
  public static void main(String args[]) throws IOException {
    String rev="";
    try {
      FileInputStream fi = new FileInputStream("input.txt");
      FileOutputStream ft = new FileOutputStream("output.txt");
      int i = 0;
      while ((i = fi.read()) != -1) {
        char c = (char) i;
        rev=c+rev;}
      for(i=0;i<rev.length();i++){
        char ch =rev.charAt(i);
        byte b = (byte)ch;
        ft.write(b);
      }
    } catch (IOException e) {
      System.out.println("Exception " + e);
    }}
}
```

Input Files

input.txt

```
Hi Welcome
```

Output Files

output.txt

```
emocleW iH
```

File Tokens

```
import java.util.*;
class FileToken {
  public static void main(String args[]) {
    int n;
    int sum = 0;
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter integers with one space gap:");
    String s = sc.nextLine();
    StringTokenizer st = new StringTokenizer(s, " ");
    while (st.hasMoreTokens()) {
      String temp = st.nextToken();
      n = Integer.parseInt(temp);
      System.out.println(n);
      sum = sum + n;
    }
    System.out.println("sum of the integers is: " + sum);
    sc.close();
  }
}
```

OUTPUT

```
Enter integers with one space gap:
1 2 3 4
1
2
3
4
sum of the integers is: 10
```

Multi Thread

```
import java.util.*;
class MainThread extends Thread{
  public void run(){
    Random randomnumber = new Random();
    for(int i=1;i<=10;i++){
    int number =randomnumber.nextInt(100);
    System.out.println("Random number is "+ number);
    if(number%2==0){
      SquareThread even = new SquareThread(number);
      even.start();}
    else{
      CubeThread odd = new CubeThread(number);
      odd.start();
    }}
try{
  Thread.sleep(1000);
}catch(InterruptedException e){
  System.out.println(e);
}}}
class SquareThread extends Thread{
  int num;
  SquareThread(int number){
    num=number;}
  public void run(){
    System.out.println("Square of "+ num +" is "+ (num*num));
  }}
class CubeThread extends Thread{
  int num;
  CubeThread(int number){
    num=number;}
  public void run(){
    System.out.println("Cube of "+ num +" is "+ (num*num*num));
  }}
public class MultiThread extends MainThread{
  public static void main(String[] args){
    MainThread obj = new MainThread();
    obj.start();}
}
```

OUTPUT

```
Random number is 14
Random number is 23
Random number is 31
Random number is 42
Random number is 34
Random number is 31
Random number is 80
Random number is 30
Random number is 85
Random number is 14
Cube of 23 is 12167
Square of 14 is 196
Square of 34 is 1156
Square of 42 is 1764
Cube of 31 is 29791
Square of 14 is 196
Cube of 85 is 614125
Square of 30 is 900
Square of 80 is 6400
Cube of 31 is 29791
```

Doubly Linked List

```
import java.util.Scanner;
class Node{
  int data;
  Node prev;
  Node next;}
class CreateNode {
  Node head = null;
  Node tail = null;
  public void insertNode(int value) {
    Node newNode = new Node();
    newNode.data = value;
    newNode.prev = null;
    newNode.next = null;
    if(head == null) {
      head = tail = newNode;  }
    else {
      tail.next = newNode;
      newNode.prev = tail;
      tail = newNode;
    } }
  public void deleteNode(int item){
    Node current = new Node();
    current=head;
    if(head==null){
      System.out.println("List is empty");
      return;}
    if(item==head.data){
      head=head.next;
      head.prev=null;
      return;}
    if(item==tail.data){
      tail=tail.prev;
      tail.next=null;
      return;}
    while(current!=null){
      if(current.data == item){
        current.next.prev = current.prev;
        current.prev.next = current.next;
        current.prev=current.next=null;}
      current=current.next; }}
  public void display() {
    Node current = head;
    if(current == null) {
      System.out.println("List is empty");
      return;
    }
    System.out.print("Doubly linked list: ");
    while(current != null) {
      System.out.print(" "+ current.data + " ");
      current = current.next;
    }
    System.out.println();
  }}
public class DoublyLinkedList{
  public static void main(String[] args) {
    CreateNode obj = new CreateNode();
    Scanner sc = new Scanner(System.in);
    obj.insertNode(4);
    obj.insertNode(3);
    obj.insertNode(7);
    obj.insertNode(10);
```

```
        System.out.println("Doubly linked list before deletion");
        obj.display();
        System.out.print("Enter the data to be deleted : ");
        int item = sc.nextInt();
        obj.deleteNode(item);
        System.out.println("Doubly linked list after deletion");
        obj.display();
        sc.close();
    }
}
```

OUTPUT

```
Doubly linked list before deletion
Doubly linked list:  4  3  7  10
Enter the data to be deleted : 7
Doubly linked list after deletion
Doubly linked list:  4  3  10
```

Thread Synchronization

```
class PrintTable{
 synchronized void printTable(int n){
    System.out.println("Multiplication table of "+ n);
    for(int i=1;i<=10;i++){
    System.out.println(n+" X "+i+" = "+n*i);
    try{
     Thread.sleep(400);
    }catch(Exception e){System.out.println(e);} }
  System.out.println();
 }}
class MyThread extends Thread{
PrintTable t= new PrintTable();
int number;
MyThread(PrintTable tab,int num){
number=num;
t=tab;}
public void run(){
t.printTable(number);
}}
public class ThreadSynchronization{
 public static void main(String args[]){
    PrintTable obj = new PrintTable();
    for(int i=1;i<=10;i++){
    MyThread t1=new MyThread(obj,i);
    t1.start();
    try{
     t1.join();}
    catch(InterruptedException e){}
}}
}
```

OUTPUT

```
Multiplication table of 1
1 X 1 = 1
1 X 2 = 2
1 X 3 = 3
1 X 4 = 4
1 X 5 = 5
1 X 6 = 6
1 X 7 = 7
1 X 8 = 8
1 X 9 = 9
1 X 10 = 10

Multiplication table of 2
2 X 1 = 2
2 X 2 = 4
2 X 3 = 6
2 X 4 = 8
2 X 5 = 10
2 X 6 = 12
2 X 7 = 14
2 X 8 = 16
2 X 9 = 18
2 X 10 = 20

Multiplication table of 3
3 X 1 = 3
3 X 2 = 6
3 X 3 = 9
3 X 4 = 12
```

3 X 5 = 15

3 X 6 = 18

3 X 7 = 21

3 X 8 = 24

3 X 9 = 27

3 X 10 = 30

Multiplication table of 4

4 X 1 = 4

4 X 2 = 8

4 X 3 = 12

4 X 4 = 16

4 X 5 = 20

4 X 6 = 24

4 X 7 = 28

4 X 8 = 32

4 X 9 = 36

4 X 10 = 40

Multiplication table of 5

5 X 1 = 5

5 X 2 = 10

5 X 3 = 15

5 X 4 = 20

5 X 5 = 25

5 X 6 = 30

5 X 7 = 35

5 X 8 = 40

5 X 9 = 45

5 X 10 = 50

Multiplication table of 6

6 X 1 = 6

6 X 2 = 12

6 X 3 = 18

6 X 4 = 24

6 X 5 = 30

6 X 6 = 36

6 X 7 = 42

6 X 8 = 48

6 X 9 = 54

6 X 10 = 60

Multiplication table of 7

7 X 1 = 7

7 X 2 = 14

7 X 3 = 21

7 X 4 = 28

7 X 5 = 35

7 X 6 = 42

7 X 7 = 49

7 X 8 = 56

7 X 9 = 63

7 X 10 = 70

Multiplication table of 8

8 X 1 = 8

8 X 2 = 16

8 X 3 = 24

8 X 4 = 32

8 X 5 = 40

8 X 6 = 48

8 X 7 = 56

8 X 8 = 64

8 X 9 = 72

8 X 10 = 80

Multiplication table of 9

9 X 1 = 9

9 X 2 = 18

9 X 3 = 27

9 X 4 = 36

9 X 5 = 45

9 X 6 = 54

9 X 7 = 63

9 X 8 = 72

9 X 9 = 81

9 X 10 = 90

Multiplication table of 10

10 X 1 = 10

10 X 2 = 20

10 X 3 = 30

10 X 4 = 40

10 X 5 = 50

10 X 6 = 60

10 X 7 = 70

10 X 8 = 80

10 X 9 = 90

10 X 10 = 100

Quick Sort

```
import java.util.Arrays;
class Quicksort {
 static int partition(int array[], int low, int high) {
  int pivot = array[high];
  int i = (low - 1);
  for (int j = low; j < high; j++) {
   if (array[j] <= pivot) {
    i++;
    int temp = array[i];
    array[i] = array[j];
    array[j] = temp;
   }}
  int temp = array[i + 1];
  array[i + 1] = array[high];
  array[high] = temp;
  return (i + 1); }
 static void quickSort(int array[], int low, int high) {
  if (low < high) {
   int pi = partition(array, low, high);
   quickSort(array, low, pi - 1);
   quickSort(array, pi + 1, high);  } }
 public static void main(String args[]) {
  int[] data = { 8, 7, 2, 1, 0, 9, 6 };
  System.out.println("Unsorted Array");
  System.out.println(Arrays.toString(data));
  int size = data.length;
  Quicksort.quickSort(data, 0, size - 1);
  System.out.println("Sorted Array in Ascending Order ");
  System.out.println(Arrays.toString(data));
} }
```

OUTPUT

```
Unsorted Array

[8, 7, 2, 1, 0, 9, 6]

Sorted Array in Ascending Order

[0, 1, 2, 6, 7, 8, 9]
```

Traffic Light

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
public class TrafficLight
extends JFrame implements ItemListener {
JRadioButton jr1;
JRadioButton jr2;
JRadioButton jr3;
JTextField j1 = new JTextField(10);
ButtonGroup b = new ButtonGroup();
String msg = " ";
int x = 0, y = 0, z = 0;
public TrafficLight(String msg) {
super(msg);
setLayout(new FlowLayout());
jr1 = new JRadioButton("Red");
jr2 = new JRadioButton("Yellow");
jr3 = new JRadioButton("Green");
jr1.addItemListener(this);
jr2.addItemListener(this);
jr3.addItemListener(this);
add(jr1);
add(jr2);
add(jr3);
b.add(jr1);
b.add(jr2);
b.add(jr3);
add(j1); }
public void itemStateChanged(ItemEvent ie) {
```

```
if (ie.getSource() == jr1) {
if (ie.getStateChange() == 1) {
msg = "Stop!";
j1.setText(msg);
x = 1;
repaint(); }
else {
msg = "";
} }
if (ie.getSource() == jr2) {
if (ie.getStateChange() == 1) {
msg = "Ready to Go";
j1.setText(msg);
y= 1;
repaint(); }
else {
msg = ""; } }
if (ie.getSource() == jr3) {
if (ie.getStateChange() == 1) {
msg = "Go!";
j1.setText(msg);
z = 1;
repaint(); }
else {
msg = "";
} } }
public void paint(Graphics g) {
g.drawRect(100, 105, 110, 270);
g.drawOval(120, 150, 60, 60);
g.drawOval(120, 230, 60, 60);
g.drawOval(120, 300, 60, 60);
```

```
if (x == 1) {
g.setColor(Color.RED);
g.fillOval(120, 150, 60, 60);
g.setColor(Color.WHITE);
g.fillOval(120, 230, 60, 60);
g.setColor(Color.WHITE);
g.fillOval(120, 300, 60, 60);
x = 0; }
if (y == 1) {
g.setColor(Color.WHITE);
g.fillOval(120, 150, 60, 60);
g.setColor(Color.YELLOW);
g.fillOval(120, 230, 60, 60);
g.setColor(Color.WHITE);
g.fillOval(120, 300, 60, 60);
y = 0;  }
if (z == 1) {
g.setColor(Color.WHITE);
g.fillOval(120, 150, 60, 60);
g.setColor(Color.WHITE);
g.fillOval(120, 230, 60, 60);
g.setColor(Color.GREEN);
g.fillOval(120, 300, 60, 60);
z = 0;  } }
public static void main(String args[]) {
JFrame jf = new TrafficLight("Traffic Light");
jf.setSize(500, 500);
jf.setVisible(true);
}}
```

Calculator

```
import javax.swing.*;
import java.awt.event.*;
class Calc implements ActionListener {
JFrame f;
JTextField t;
JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,badd,bsub,bmul,bdiv,beq,bdot,bdel,bclr;
static double a=0,b=0,c=0;
static int operator;
Calc() {
f=new JFrame("Calc");
t=new JTextField();
b1=new JButton("1");
b2=new JButton("2");
b3=new JButton("3");
b4=new JButton("4");
b5=new JButton("5");
b6=new JButton("6");
b7=new JButton("7");
b8=new JButton("8");
b9=new JButton("9");
b0=new JButton("0");
badd=new JButton("+");
bsub=new JButton("-");
bmul=new JButton("*");
bdiv=new JButton("/");
beq=new JButton("=");
bdot=new JButton(".");
bdel=new JButton("DEL");
bclr=new JButton("CLR");
t.setBounds(30,40,370,30);
```

```
b9.setBounds(180,100,50,40);
b8.setBounds(110,100,50,40);
b7.setBounds(40,100,50,40);
b4.setBounds(40,150,50,40);
b5.setBounds(110,150,50,40);
b6.setBounds(180,150,50,40);
b1.setBounds(40,200,50,40);
b2.setBounds(110,200,50,40);
b3.setBounds(180,200,50,40);
b0.setBounds(110,250,50,40);
beq.setBounds(180,250,50,40);
bdot.setBounds(40,250,50,40);
badd.setBounds(250,200,50,90);
bsub.setBounds(250,150,50,40);
bmul.setBounds(250,100,50,40);
bdiv.setBounds(320,100,60,40);
bdel.setBounds(320,150,60,40);
bclr.setBounds(320,200,60,90);
f.add(t);
f.add(b1);
f.add(b2);
f.add(b3);
f.add(b4);
f.add(b5);
f.add(b6);
f.add(b7);
f.add(b8);
f.add(b9);
f.add(b0);
f.add(beq);
f.add(bdot);
```

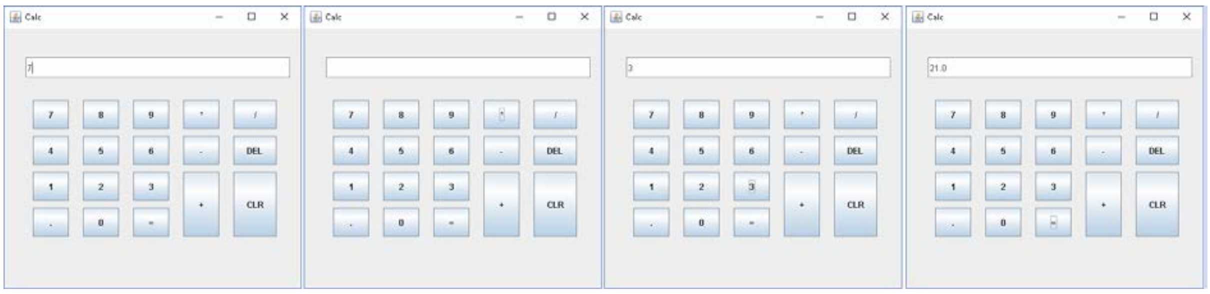
```
f.add(badd);
f.add(bsub);
f.add(bmul);
f.add(bdiv);
f.add(bdel);
f.add(bclr);
f.setLayout(null);
f.setVisible(true);
f.setSize(430,400);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
b5.addActionListener(this);
b6.addActionListener(this);
b7.addActionListener(this);
b8.addActionListener(this);
b9.addActionListener(this);
b0.addActionListener(this);
badd.addActionListener(this);
bsub.addActionListener(this);
bmul.addActionListener(this);
bdiv.addActionListener(this);
bdel.addActionListener(this);
bclr.addActionListener(this);
bdot.addActionListener(this);
beq.addActionListener(this); }
public void actionPerformed(ActionEvent e) {
if(e.getSource()==b1)
t.setText(t.getText().concat("1"));
```

```
if(e.getSource()==b2)
t.setText(t.getText().concat("2"));
if(e.getSource()==b3)
t.setText(t.getText().concat("3"));
if(e.getSource()==b4)
t.setText(t.getText().concat("4"));
if(e.getSource()==b5)
t.setText(t.getText().concat("5"));
if(e.getSource()==b6)
t.setText(t.getText().concat("6"));
if(e.getSource()==b7)
t.setText(t.getText().concat("7"));
if(e.getSource()==b8)
t.setText(t.getText().concat("8"));
if(e.getSource()==b9)
t.setText(t.getText().concat("9"));
if(e.getSource()==b0)
t.setText(t.getText().concat("0"));
if(e.getSource()==badd) {
a=Double.parseDouble(t.getText());
operator=1;
t.setText(""); }
if(e.getSource()==bsub) {
a=Double.parseDouble(t.getText());
operator=2;
t.setText(""); }
if(e.getSource()==bmul) {
a=Double.parseDouble(t.getText());
operator=3;
t.setText(""); }
if(e.getSource()==bdiv) {
```

```
a=Double.parseDouble(t.getText());
operator=4;
t.setText(""); }
if(e.getSource()==beq) {
b=Double.parseDouble(t.getText());
switch(operator) {
case 1:
c=a+b;
t.setText(""+c);
break;
case 2:
c=a-b;
t.setText(""+c);
break;
case 3:
c=a*b;
t.setText(""+c);
break;
case 4:
c=a/b;
t.setText(""+c);
break;
default:
c=0; } }
if(e.getSource()==bclr)
t.setText("");
if(e.getSource()==bdel) {
String s=t.getText();
t.setText("");
for(int i=0;i<s.length()-1;i++)
t.setText(t.getText()+s.charAt(i)); } }
```

```
public static void main(String args[]) {
Calc cal=new Calc();
} }
```

OUTPUT



<u>Fibonacci Series Using Packages</u>

```
// package
package FibSeries;
public class Series{
  public void printfibo(int n){
    int f1=0,f2=1,f3=0;
    System.out.println("Fibonacci Series with "+ n + " terms");
    System.out.print(f1 +" ");
    System.out.print(f2 + " ");
    for(int i=2;i<n;i++) {
      f3=f1+f2;
      f1=f2;
      f2=f3;
      System.out.print(f3 + " "); } }
}

//java program to display Fibonacci series by importing package ‘FibSeries’
import FibSeries.Series;
import java.util.Scanner;
public class Fibonacci {
  public static void main(String args[]) {
    Series obj = new Series();
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the no. of terms ");
    int n = sc.nextInt();
    obj.printfibo(n);
    sc.close(); }
}
```

OUTPUT

```
Enter the no. of terms
6
Fibonacci Series with 6 terms
0 1 1 2 3 5
```

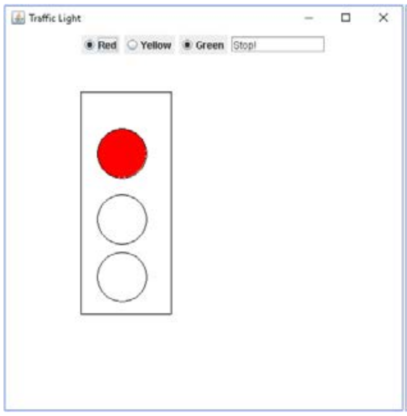
Quick Sort

```
import java.util.Scanner;
public class StringQuickSort {
  String names[];
  int length;
  public static void main(String[] args) {
    StringQuickSort sorts = new StringQuickSort();
    Scanner sc1=new Scanner(System.in);
    System.out.println("Enter the number of list elements");
    int n=sc1.nextInt();
    String[] words=new String[n];
    System.out.print("Enter the list elements \n");
    for(int i=0;i<n;i++){
      words[i]=sc1.next();  }
    sorts.sort(words);
    System.out.println("The sorted list ");
    for (String i : words) {
      System.out.print(i);
      System.out.print("\t");  } }
  void sort(String array[]) {
    if (array == null || array.length == 0) {
      return;  }
    this.names = array;
    this.length = array.length;
    quickSort(0, length - 1);}
  void quickSort(int low, int high) {
    int i = low;
    int j = high;
    String pivot = this.names[low + (high - low) / 2];
    while (i <= j) {
      while (this.names[i].compareToIgnoreCase(pivot) < 0) {
        i++; }
      while (this.names[j].compareToIgnoreCase(pivot) > 0) {
        j--; }
      if (i <= j) {
        swap(i, j);
        i++;
        j--; } }
    if (low < j) {
      quickSort(low, j);  }
    if (i < high) {
      quickSort(i, high);
    }}
  void swap(int i, int j) {
    String temp = this.names[i];
    this.names[i] = this.names[j];
    this.names[j] = temp; }
}
```

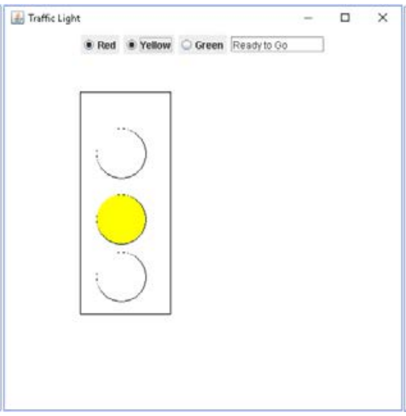
OUTPUT

```
Enter the number of list elements

4

Enter the list elements

Hina

Peter

Aby

Bob

The sorted list

Aby     Bob     Hina    Peter
```

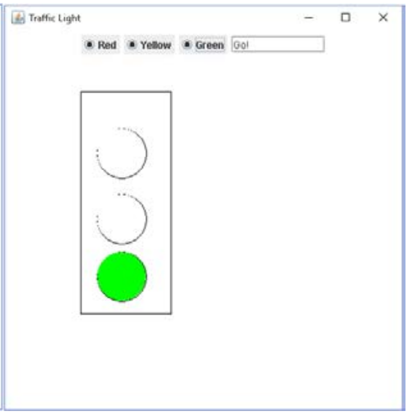
OUTPUT



Stop!



Ready to Go



Go!