UTTARANCHAL UNIVERSITY

(Established vide Uttaranchal University Act, 2012)
(Uttarakhand Act No. 11 of 2013)
Arcadia Grant, P.O. Chandanwari, Premnagar, Dehradun, Uttarakhand

UTTARANCHAL
UNIVERSITY

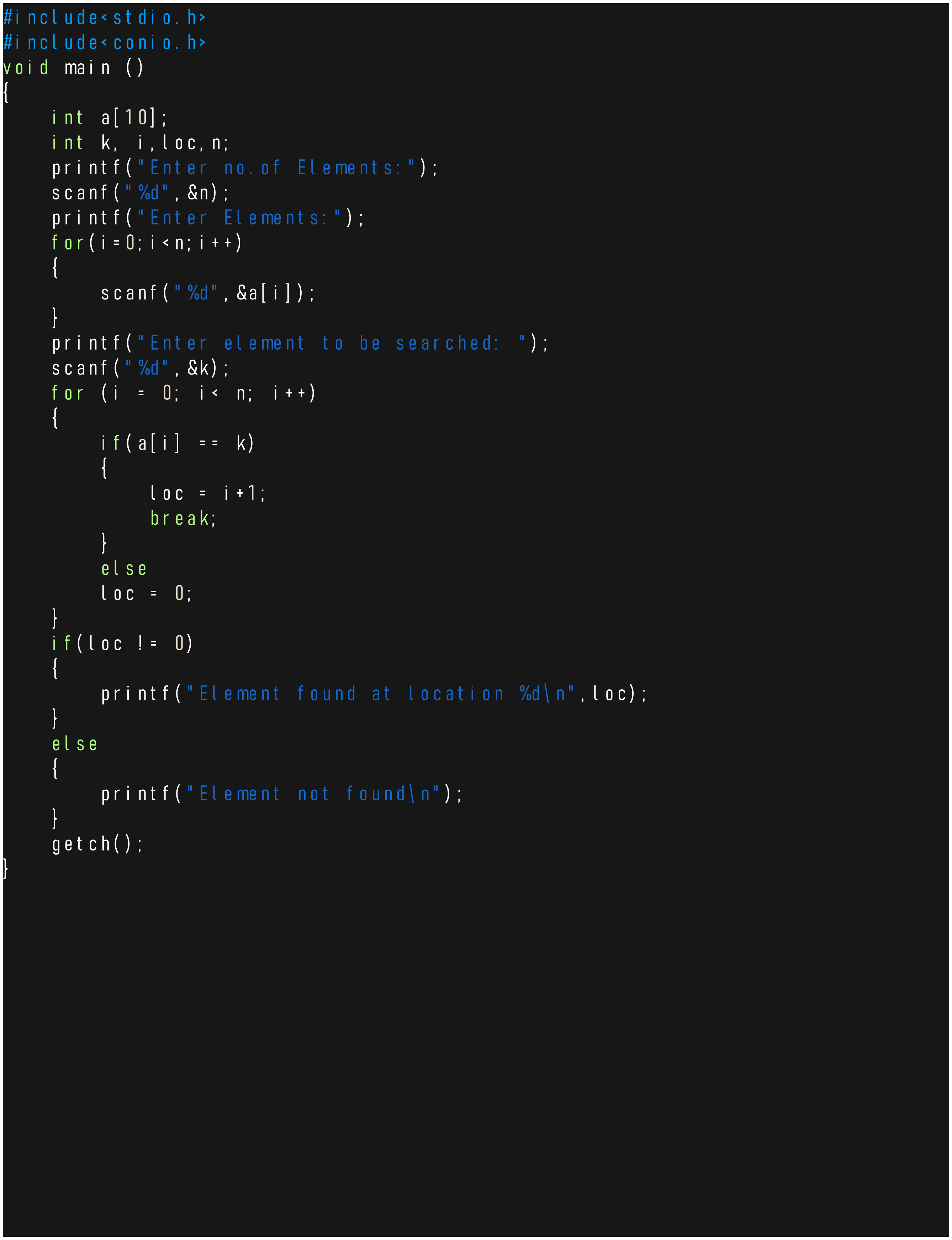
Academic Session 2020-21 (Even Semester)

Student Name	:	**Piyush Bhatt**
University Roll Number	:	1901090008
Lab Name	:	**Design and analysis of algorithm Lab**
Lab Code	:	**PCS - 403**
Session Year	:	**2020-21**
Semester	:	**4th**
Course	:	**Bachelor of Technology (B.Tech-CSE)**
Submitted to	:	**Ms. Madhu Kirola**

Department of Computer Science & Engineering
Uttaranchal Institute of Technology

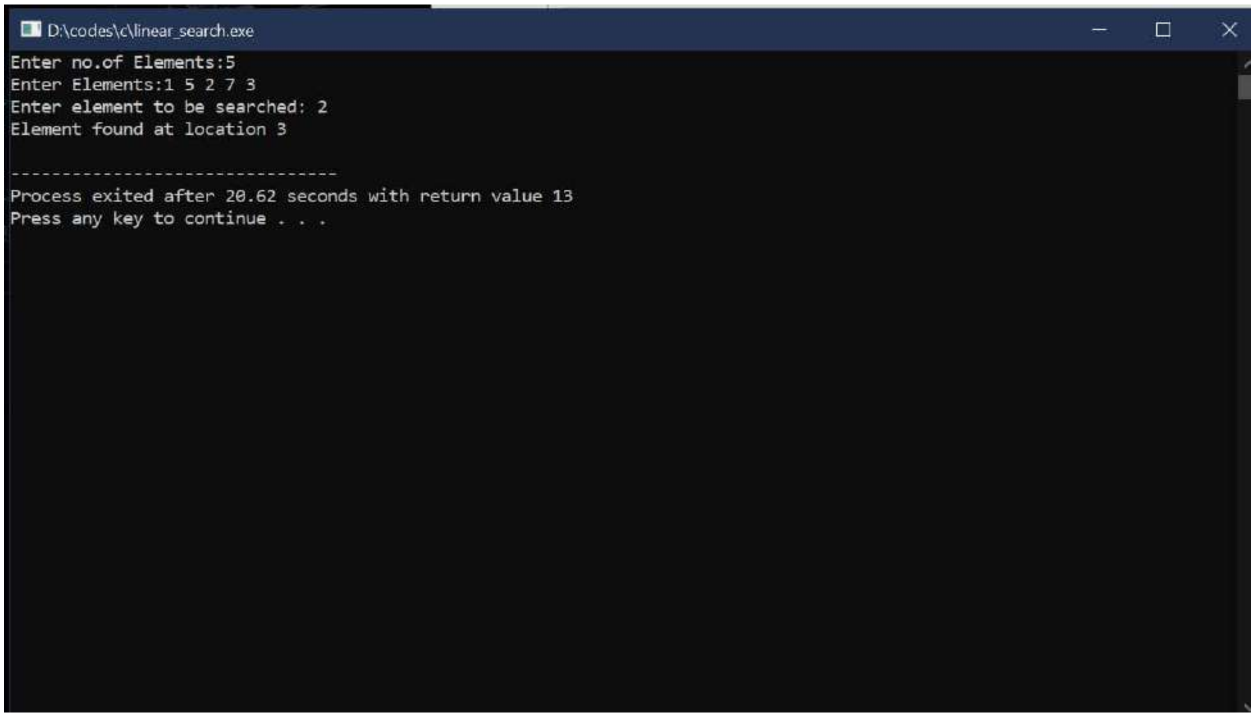
UTTARANCHAL
INSTITUTE OF
TECHNOLOGY

INDEX



```c
#include<stdio.h>
#include<conio.h>
void main ()
{
    int a[10];
    int k, i,loc,n;
    printf("Enter no.of Elements:");
    scanf("%d",&n);
    printf("Enter Elements:");
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    printf("Enter element to be searched: ");
    scanf("%d",&k);
    for (i = 0; i< n; i++)
    {
        if(a[i] == k)
        {
            loc = i+1;
            break;
        }
        else
        loc = 0;
    }
    if(loc != 0)
    {
        printf("Element found at location %d\n",loc);
    }
    else
    {
        printf("Element not found\n");
    }
    getch();
}
```

Output:



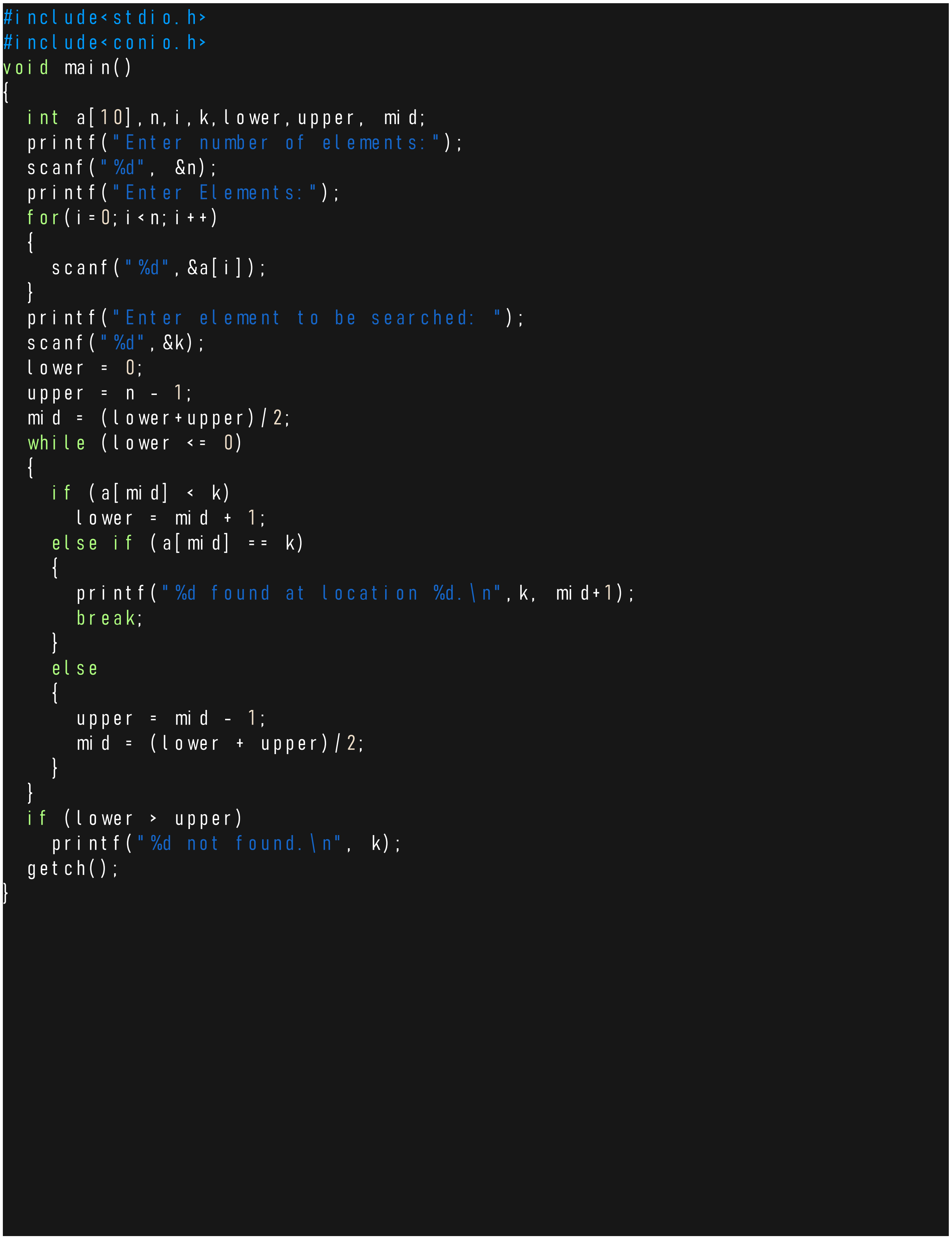
```
D:\codes\c\linear_search.exe
Enter no.of Elements:5
Enter Elements:1 5 2 7 3
Enter element to be searched: 2
Element found at location 3

--------------------------------
Process exited after 20.62 seconds with return value 13
Press any key to continue . . .
```

Complexity:

Best case : O(1)

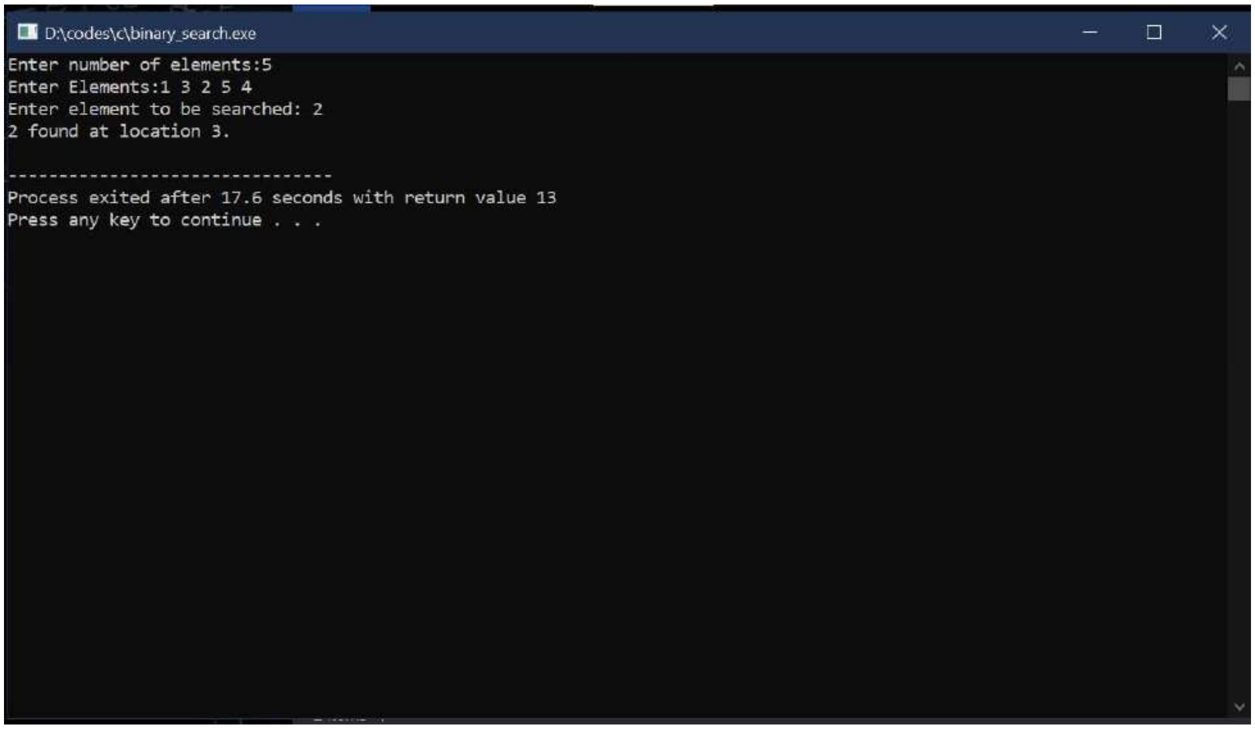
Average case : O(n)

Worst case : O(n)



```c
#include<stdio.h>
#include<conio.h>
void main()
{
  int a[10],n,i,k,lower,upper, mid;
  printf("Enter number of elements:");
  scanf("%d", &n);
  printf("Enter Elements:");
  for(i=0;i<n;i++)
  {
    scanf("%d",&a[i]);
  }
  printf("Enter element to be searched: ");
  scanf("%d",&k);
  lower = 0;
  upper = n - 1;
  mid = (lower+upper)/2;
  while (lower <= 0)
  {
    if (a[mid] < k)
      lower = mid + 1;
    else if (a[mid] == k)
    {
      printf("%d found at location %d.\n",k, mid+1);
      break;
    }
    else
    {
      upper = mid - 1;
      mid = (lower + upper)/2;
    }
  }
  if (lower > upper)
    printf("%d not found.\n", k);
  getch();
}
```

Output :



D:\codes\c\binary_search.exe

```
Enter number of elements:5
Enter Elements:1 3 2 5 4
Enter element to be searched: 2
2 found at location 3.

--------------------------------
Process exited after 17.6 seconds with return value 13
Press any key to continue . . .
```

Complexity:

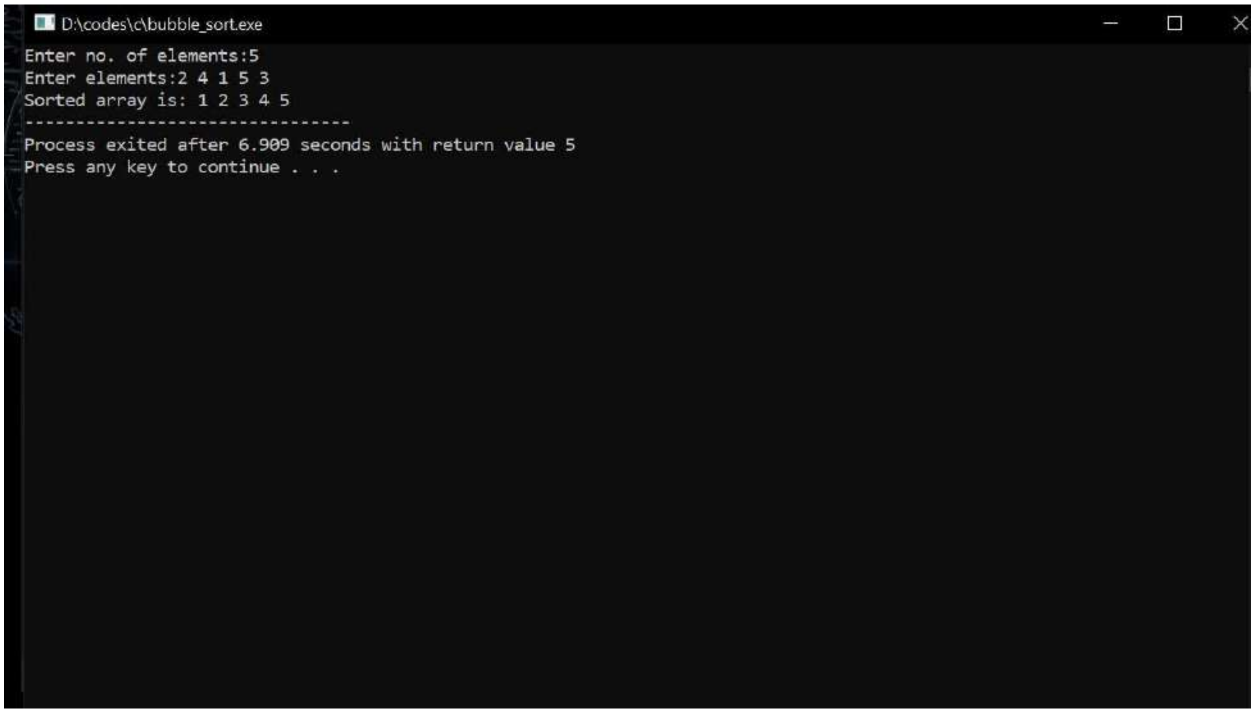
Best case : O(n)

Average case : O(n^2)

Worst case : O(n^2)

```c
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[10],swap,i,j,n;
    printf("Enter no.of Elements:");
    scanf("%d",&n);
    printf("Enter Elements:");
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    for(i=0;i<n-1;j++)
    {
        for(j=0;j<n-i-1;j++)
        {
            if(a[j]>a[j+1])
            {
                swap=a[j];
                a[j]=a[j+1];
                a[j+1]=swap;
            }
        }
    }
    printf("Sorted list is:");
    for(i=0;i<n;i++)
    {
         printf("%d ",a[i]);
    }
    getch();
}
```

Output:



```
D:\codes\c\bubble_sort.exe
Enter no. of elements:5
Enter elements:2 4 1 5 3
Sorted array is: 1 2 3 4 5
--------------------------------
Process exited after 6.909 seconds with return value 5
Press any key to continue . . .
```

Complexity:

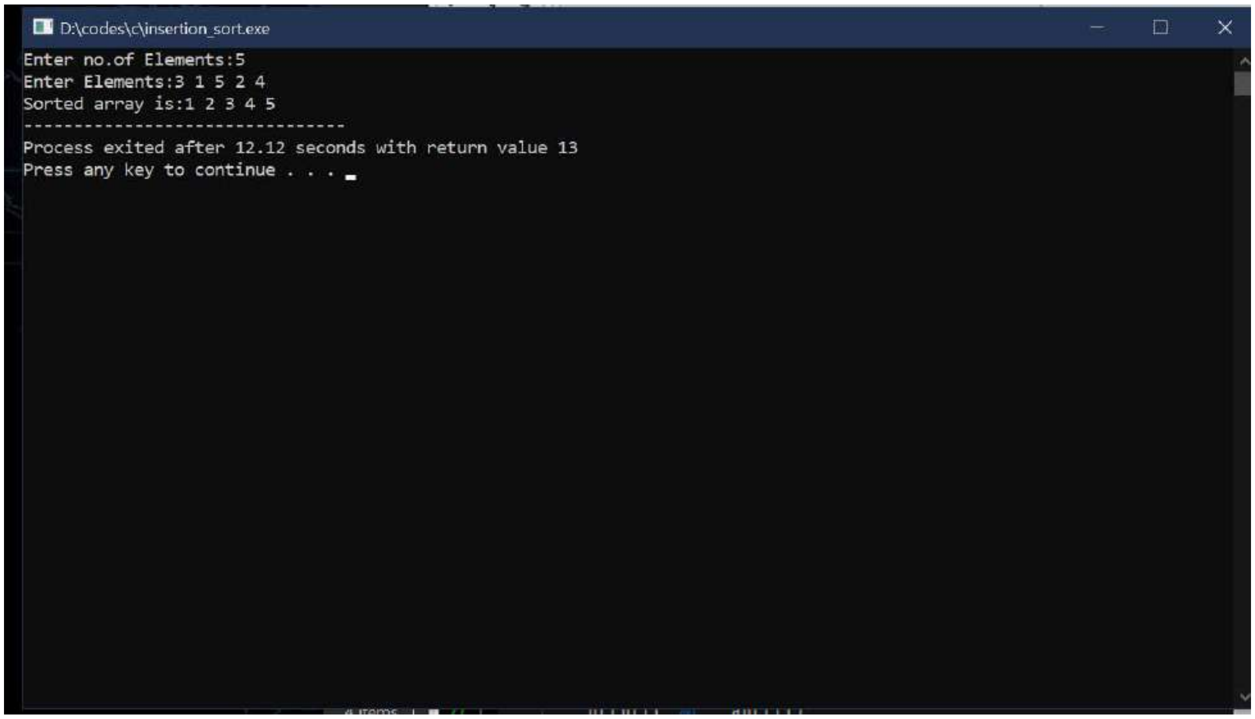
Best case : O(n)

Average case : O(n^2)

Worst case : O(n^2)

```c
#include<stdio.h>
#include<conio.h>
void main()
{
    int abc[10],i,j,k,n;
    printf("Enter no.of Elements:");
    scanf("%d",&n);
    printf("Enter Elements:");
    for(i=0;i<n;i++)
    {
        scanf("%d",&abc[i]);
    }
    for(i=1;i<n;i++)
    {
        k=abc[i];
        j=i-1;
        while(j>=0&&abc[j]>k)
        {
            abc[j+1]=abc[j];
            j=j-1;
        }
        abc[j+1]=k;
    }
    printf("Sorted array is:");
    for(i=0;i<n;i++)
    {
        printf("%d ",abc[i]);
    }
    getch();
}
```

Output:



```
D:\codes\c\insertion_sort.exe
Enter no.of Elements:5
Enter Elements:3 1 5 2 4
Sorted array is:1 2 3 4 5
--------------------------------
Process exited after 12.12 seconds with return value 13
Press any key to continue . . .
```

Complexity:

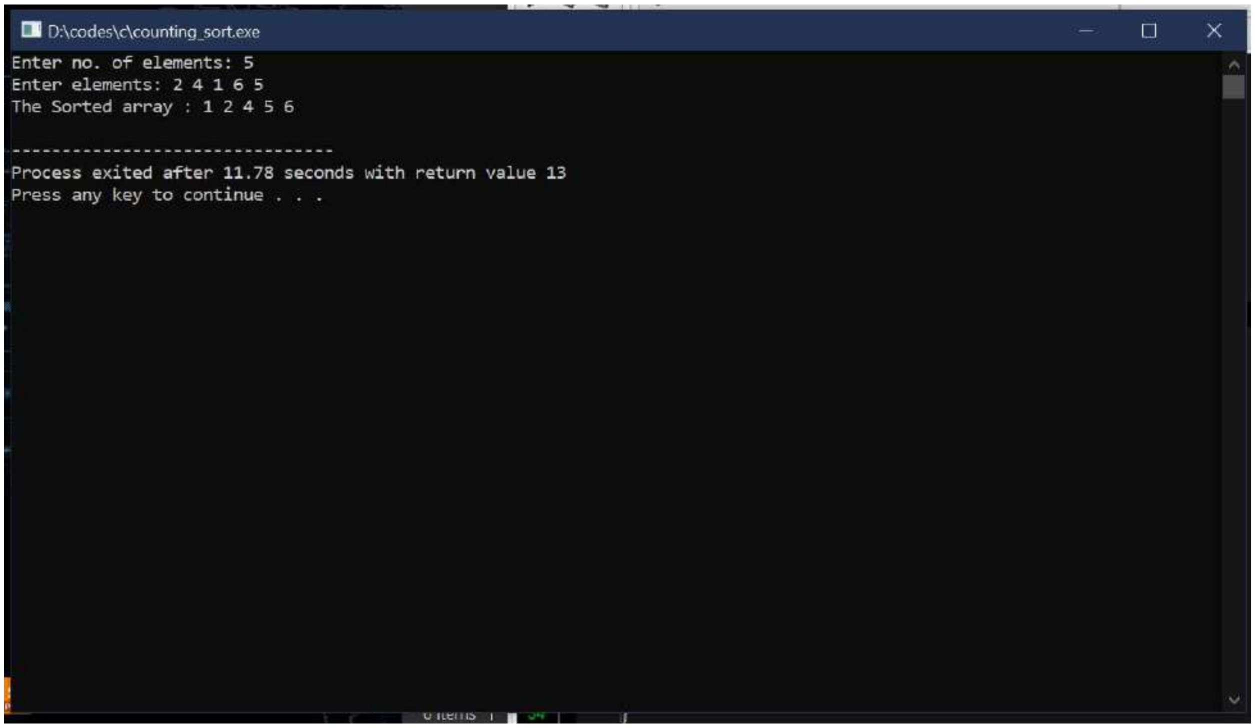
Best case : O(n)

Average case : O(n^2)

Worst case : O(n^2)

```c
#include <stdio.h>
#include <conio.h>
void  counting_sort(int a[], int k, int n)
{
  int i, j;
  int b[15], c[100];
  for (i = 0; i <= k; i++)
    c[i] = 0;
  for (j = 1; j <= n; j++)
    c[a[j]] = c[a[j]] + 1;
  for (i = 1; i <= k; i++)
    c[i] = c[i] + c[i-1];
  for (j = n; j >= 1; j--)
  {
    b[c[a[j]]] = a[j];
    c[a[j]] = c[a[j]] - 1;
  }
  printf("The Sorted array : ");
  for (i = 1; i <= n; i++)
    printf("%d ", b[i]);
}
void  main()
{
  int n, k = 0, a[10], i;
  printf("Enter no. of elements: ");
  scanf("%d", &n);
  printf("Enter elements: ");
  for (i = 1; i <= n; i++)
  {
    scanf("%d", &a[i]);
    if (a[i] > k)
    {
       k = a[i];
    }
  }
  counting_sort(a, k, n);
  printf("\n");
  getch();
}
```

Output:



```
Enter no. of elements: 5
Enter elements: 2 4 1 6 5
The Sorted array : 1 2 4 5 6

--------------------------------
Process exited after 11.78 seconds with return value 13
Press any key to continue . . .
```

Complexity:

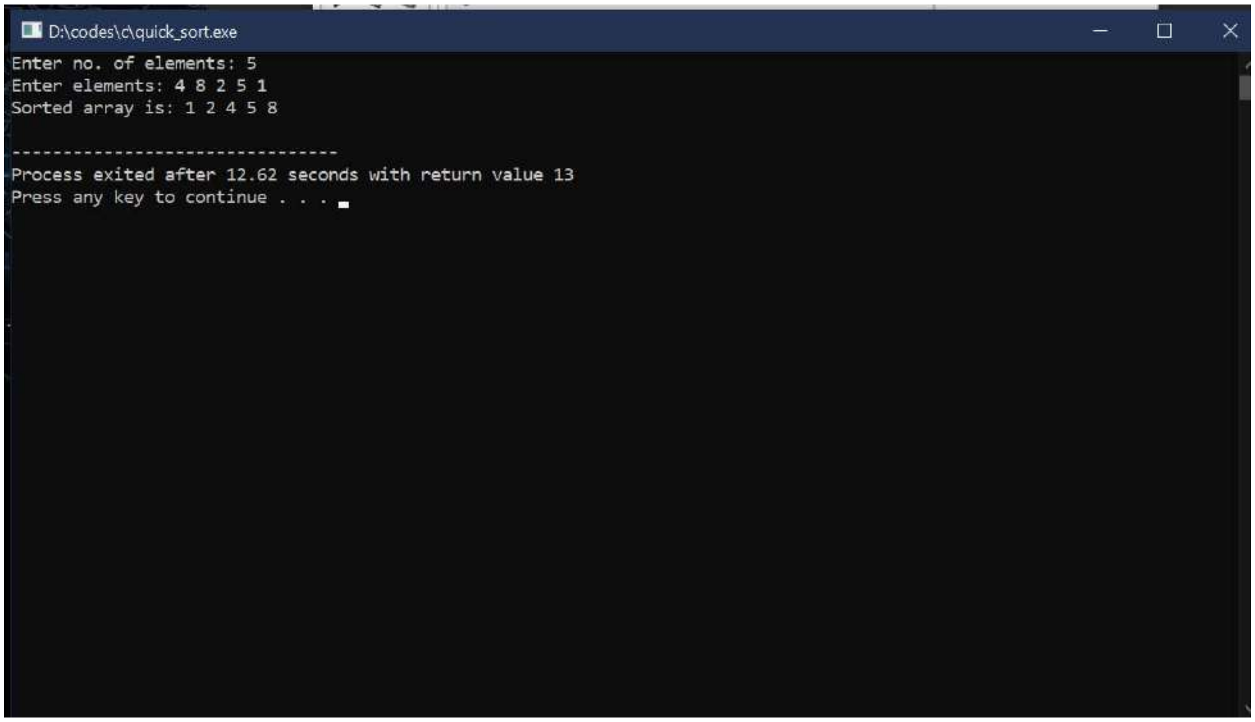
Best case : O(n + k)

Average case: O(n+k) where n is the number of elements in input array and k is the range of input.

Worst case : O(n+k)

```c
#include <stdio.h>
#include <conio.h>
void quicksort (int [], int, int);
int main()
{
    int a[10];
    int n, i;
    printf("Enter no. of elements: ");
    scanf("%d", &n);
    printf("Enter elements: ");
    for (i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
    quicksort(a, 0, n - 1);
    printf("Sorted array is: ");
    for (i = 0; i < n; i++)
    {
        printf("%d ",a[i]);
    }
    printf("\n");
    getch();
}
void quicksort(int a[], int low, int high)
{
    int pivot, i, j, temp;
    if (low < high)
    {
        pivot = low;
        i = low;
        j = high;
        while (i < j)
        {
            while (a[i] <= a[pivot] && i <= high)
            {
                i++;
            }
            while (a[j] > a[pivot] && j >= low)
            {
                j--;
            }
            if (i < j)
            {
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
            }
        }
        temp = a[j];
        a[j] = a[pivot];
        a[pivot] = temp;
        quicksort(a, low, j - 1);
        quicksort(a, j + 1, high);
    }
}
```

Output:



D:\codes\c\quick_sort.exe

```
Enter no. of elements: 5
Enter elements: 4 8 2 5 1
Sorted array is: 1 2 4 5 8

--------------------------------
Process exited after 12.62 seconds with return value 13
Press any key to continue . . .
```

Complexity:

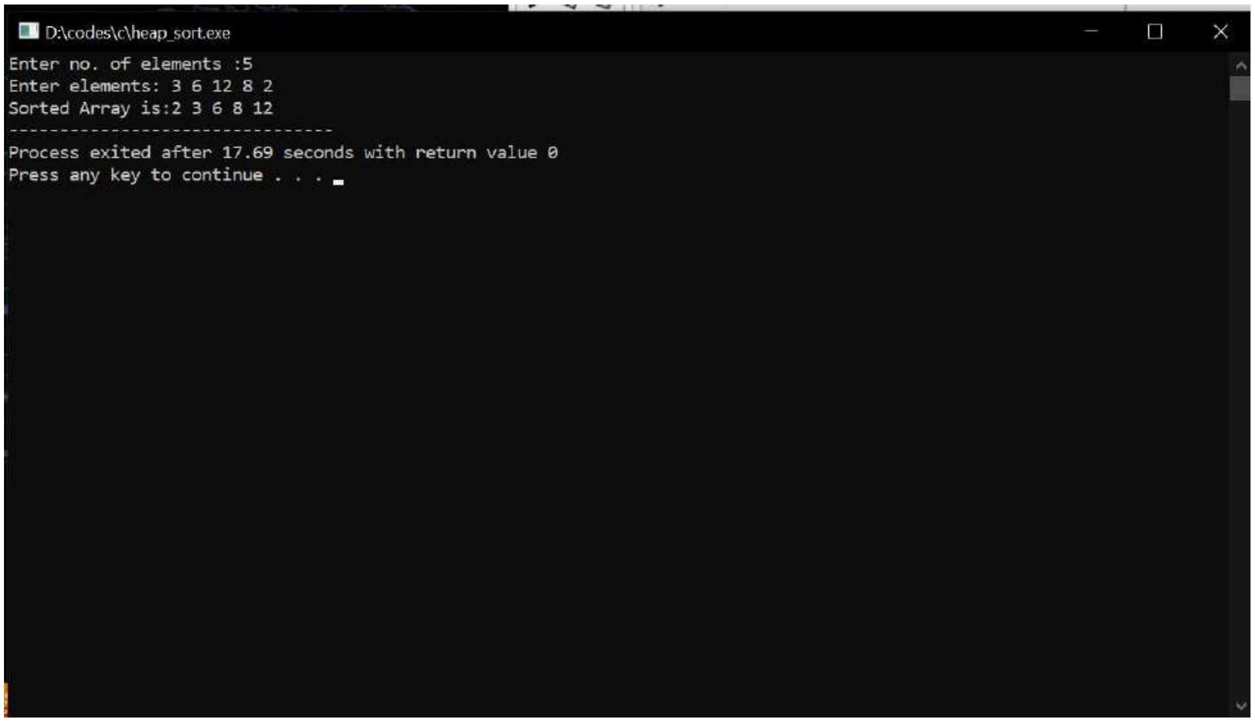
Best case : O(n log n)

Average case : O(n log n)

Worst case : O(n^2)

```c
#include<stdio.h>
#include <conio.h>
void Adjust(int Heap_of_Numbers[],int i)
{
int j,copy,Number,Reference = 1;
Number=Heap_of_Numbers[0];
while(2*i<=Number && Reference==1)
{
j=2*i;
if(j+1<=Number && Heap_of_Numbers[j+1] > Heap_of_Numbers[j])
j=j+1;
if( Heap_of_Numbers[j] < Heap_of_Numbers[i])
Reference=0;
else
{
copy=Heap_of_Numbers[i];
Heap_of_Numbers[i]=Heap_of_Numbers[j];
Heap_of_Numbers[j]=copy;
i=j;
}
}
}
void Make_Heap(int heap[])
{
int i;
int Number_of_Elements;
Number_of_Elements=heap[0];
for(i=Number_of_Elements/2;i>=1;i--)
Adjust(heap,i);
}
int main()
{
int heap[30];
int NumberofElements;
int i;
int LastElement;
int CopyVariable;
printf("Enter no. of elements :");
scanf("%d",&NumberofElements);
printf("Enter elements: ");
for(i=1;i<=NumberofElements;i++)
scanf("%d",&heap[i]);
heap[0]=NumberofElements;
Make_Heap(heap);
while(heap[0] > 1)
{
LastElement=heap[0];
CopyVariable=heap[1];
heap[1]=heap[LastElement];
heap[LastElement]=CopyVariable;
heap[0]--;
Adjust(heap,1);
}
printf("Sorted Array is:");
for(i=1;i<=NumberofElements;i++)
printf("%d ",heap[i]);
return 0;
}
```

Output :



```
D:\codes\c\heap_sort.exe
Enter no. of elements :5
Enter elements: 3 6 12 8 2
Sorted Array is:2 3 6 8 12
--------------------------------
Process exited after 17.69 seconds with return value 0
Press any key to continue . . .
```

Complexity:

Best case : O(n log n)

Average case : O(n log n)

Worst case : O(n log n)