

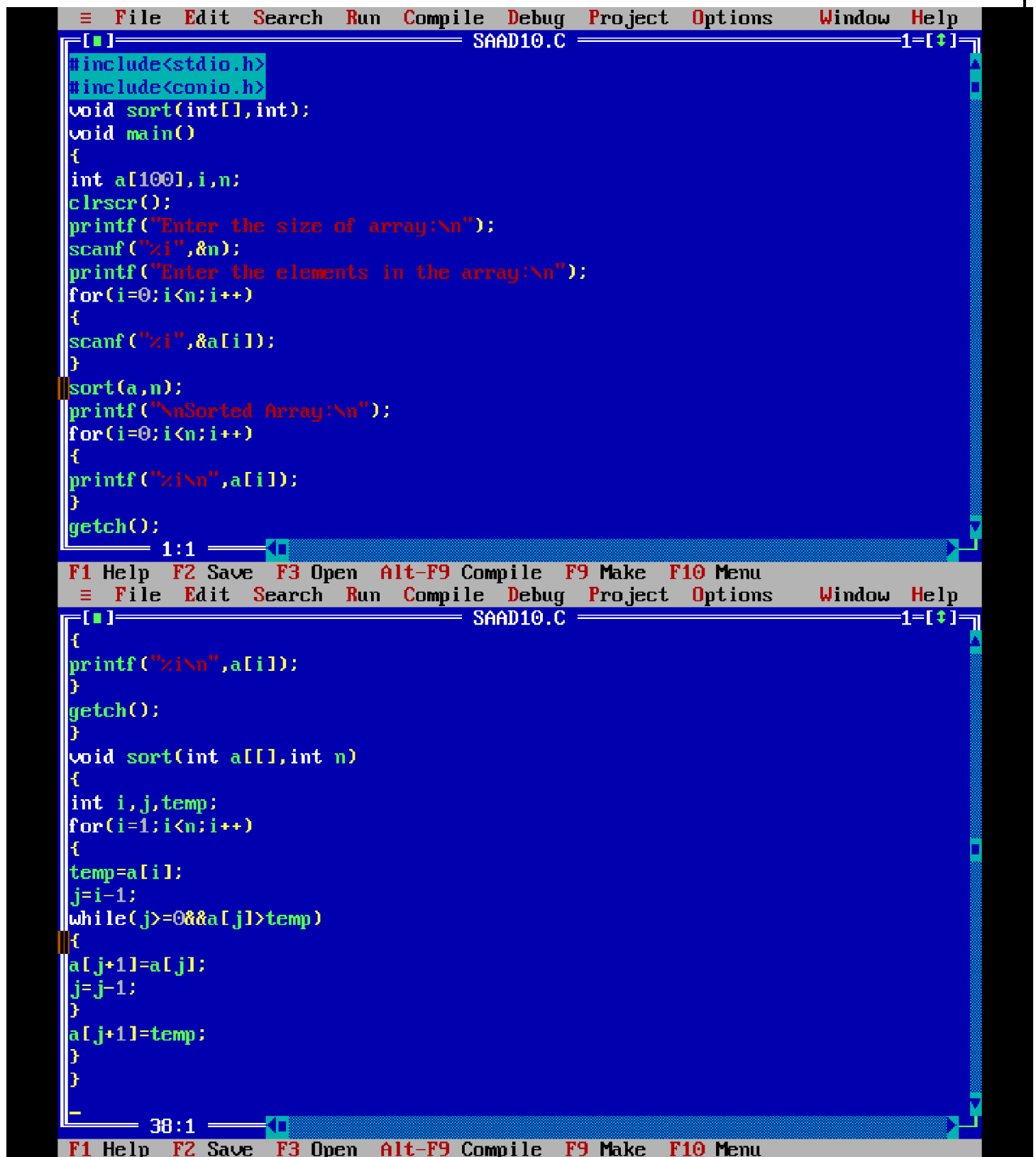
Subject: - DATA STRUCTURE	Subject Code: 313 301
Semester: - III	Course: DATA STRUCTURES
Laboratory No: L001	Name of Subject Teacher: Prof. Imraan S.
Name of Student: Saad Sharif Kazi	Roll Id: - 24203A0013
Experiment No:	10
Title of Experiment	* Write a 'C' Program to Sort an Array of numbers using Insertion Sort Method

Aim: * Write a 'C' Program to Sort an Array of numbers using Insertion Sort Method.

Algorithm:

Step 1: Start
 Step 2: Declare an integer array a[100] and variables i, n
 Step 3: Clear screen using clrscr()
 Step 4: Print "Enter the size of the array:"
 Step 5: Scan the value of n from keyboard
 Step 6: Print "Enter the elements in the array:"
 Step 7: Run a loop from i = 0 to i < n
 Step 7.1: Scan each element and store it in a[i]
 Step 8: Call the function sort(a, n)
 Step 9: Inside the sort() function
 Step 9.1: Declare integer variables i, j, temp
 Step 9.2: Run a loop from i = 1 to i < n
 Step 9.2.1: Set temp = a[i]
 Step 9.2.2: Set j = i - 1
 Step 9.2.3: While j >= 0 and a[j] > temp, repeat
 Step 9.2.3.1: Set a[j + 1] = a[j]
 Step 9.2.3.2: Decrement j by 1
 Step 9.2.4: Set a[j + 1] = temp
 Step 10: After returning from function, print "Sorted Array:"
 Step 11: Run a loop from i = 0 to i < n
 Step 11.1: Print a[i]
 Step 12: Stop

Code:



```
File Edit Search Run Compile Debug Project Options Window Help
SAAD10.C 1=1
#include<stdio.h>
#include<conio.h>
void sort(int[],int);
void main()
{
    int a[100],i,n;
    clrscr();
    printf("Enter the size of array:\n");
    scanf("%d",&n);
    printf("Enter the elements in the array:\n");
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    sort(a,n);
    printf("\nSorted Array:\n");
    for(i=0;i<n;i++)
    {
        printf("%d\n",a[i]);
    }
    getch();
}

1:1

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu
File Edit Search Run Compile Debug Project Options Window Help
SAAD10.C 1=1
{
    printf("%d\n",a[i]);
}
getch();
}
void sort(int a[],int n)
{
    int i,j,temp;
    for(i=1;i<n;i++)
    {
        temp=a[i];
        j=i-1;
        while(j>=0&& a[j]>temp)
        {
            a[j+1]=a[j];
            j=j-1;
        }
        a[j+1]=temp;
    }
}
```

OUTPUT: -

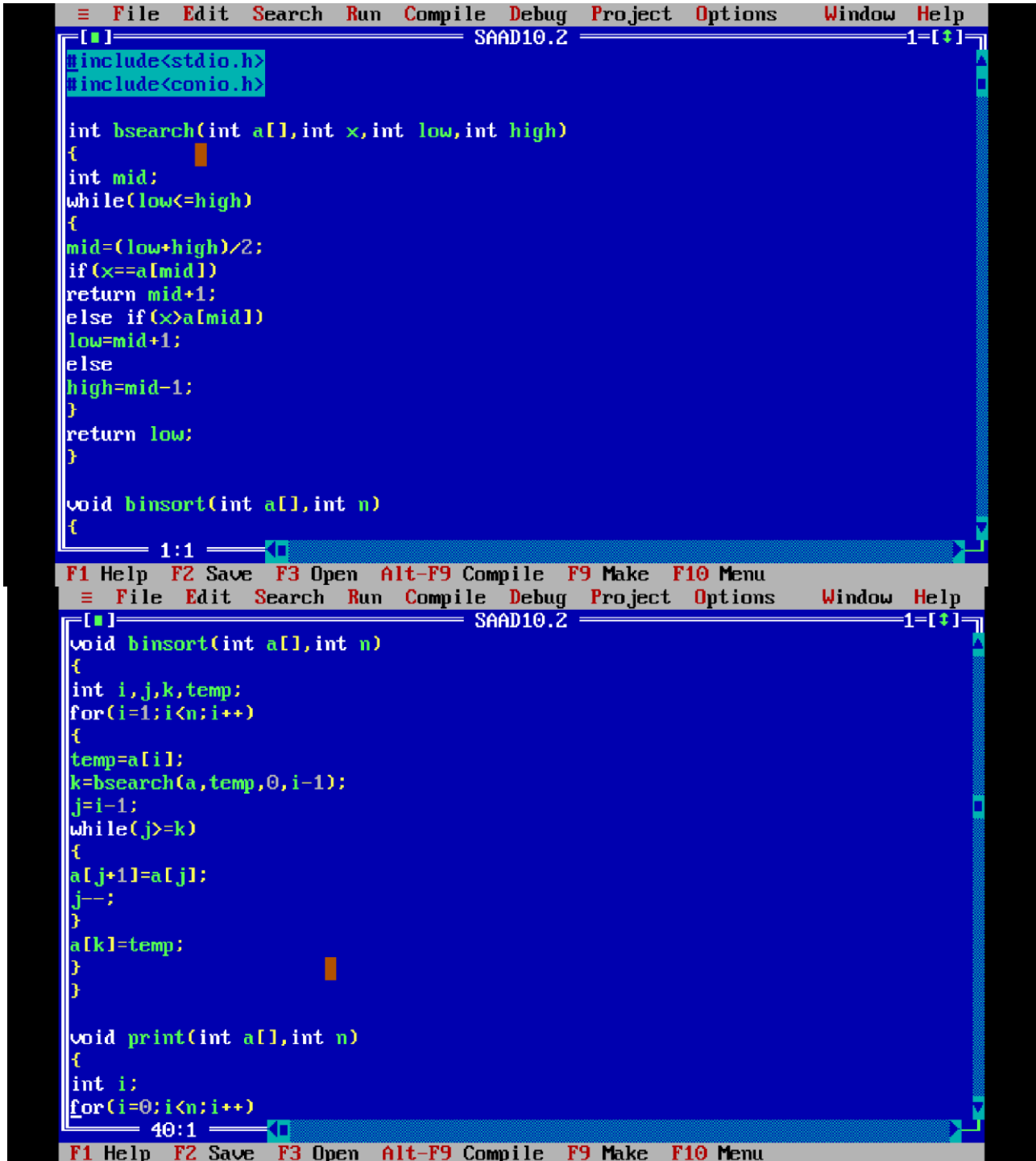
```
Enter the size of array:
5
Enter the elements in the array:
34
54
12
76
45

Sorted Array:
12
34
45
54
76
-
```

Practical Related Questions:

1. Modify the Insertion Sort algorithm to use binary search for finding the correct position to insert the current element. Implement this modified algorithm and compare its performance with the standard Insertion Sort.

CODE:



```
SAAD10.2
#include<stdio.h>
#include<conio.h>

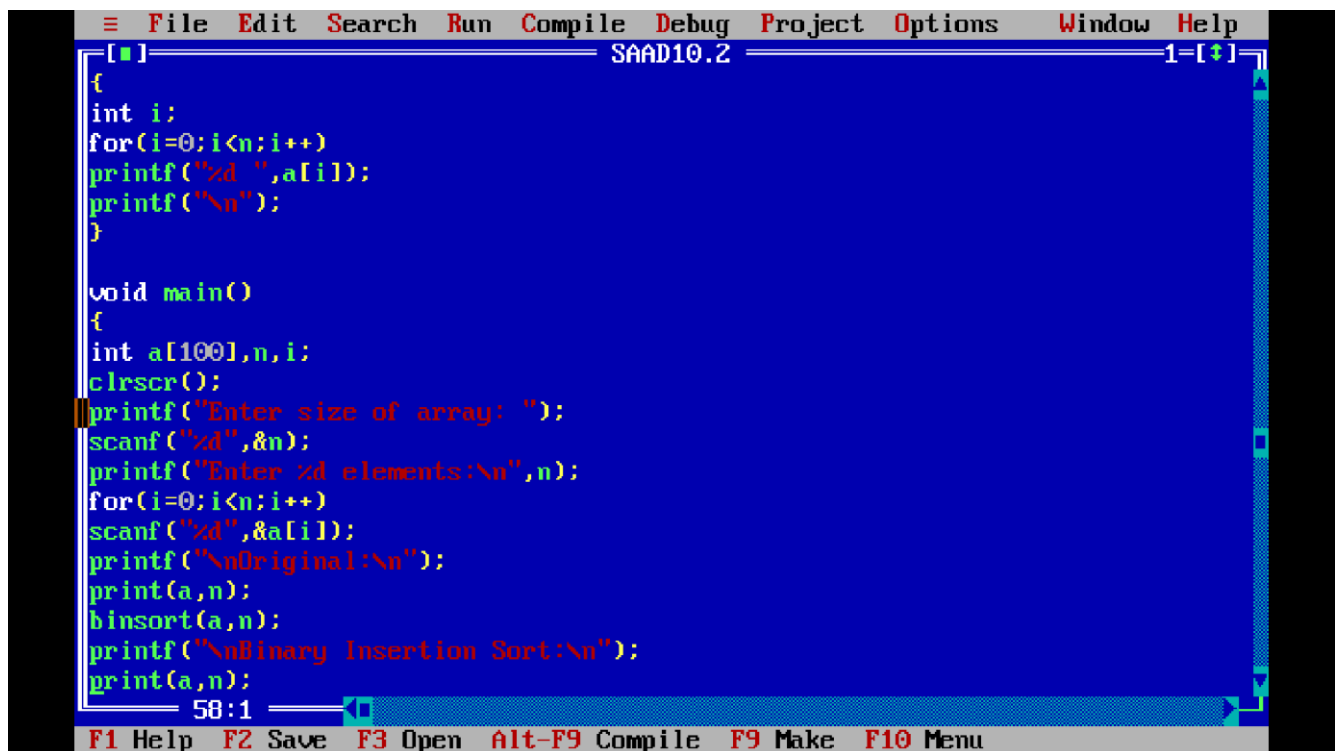
int bsearch(int a[],int x,int low,int high)
{
    int mid;
    while(low<=high)
    {
        mid=(low+high)/2;
        if(x==a[mid])
            return mid+1;
        else if(x>a[mid])
            low=mid+1;
        else
            high=mid-1;
    }
    return low;
}

void binsort(int a[],int n)
{
    1:1
}

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu
SAAD10.2
void binsort(int a[],int n)
{
    int i,j,k,temp;
    for(i=1;i<n;i++)
    {
        temp=a[i];
        k=bsearch(a,temp,0,i-1);
        j=i-1;
        while(j>=k)
        {
            a[j+1]=a[j];
            j--;
        }
        a[k]=temp;
    }
}

void print(int a[],int n)
{
    int i;
    for(i=0;i<n;i++)
    40:1
}

F1 Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu
```



The screenshot shows a Turbo C++ IDE window titled "SAAD10.2". The menu bar includes File, Edit, Search, Run, Compile, Debug, Project, Options, Window, and Help. The code is as follows:

```
[■] SAAD10.2 1-[+]  
{  
int i;  
for(i=0;i<n;i++)  
printf("%d ",a[i]);  
printf("\n");  
}  
  
void main()  
{  
int a[100],n,i;  
clrscr();  
printf("Enter size of array: ");  
scanf("%d",&n);  
printf("Enter %d elements:\n",n);  
for(i=0;i<n;i++)  
scanf("%d",&a[i]);  
printf("\nOriginal:\n");  
print(a,n);  
binsort(a,n);  
printf("\nBinary Insertion Sort:\n");  
print(a,n);  
} 58:1
```

The status bar at the bottom shows function key shortcuts: F1 Help, F2 Save, F3 Open, Alt-F9 Compile, F9 Make, and F10 Menu.

OUTPUT:

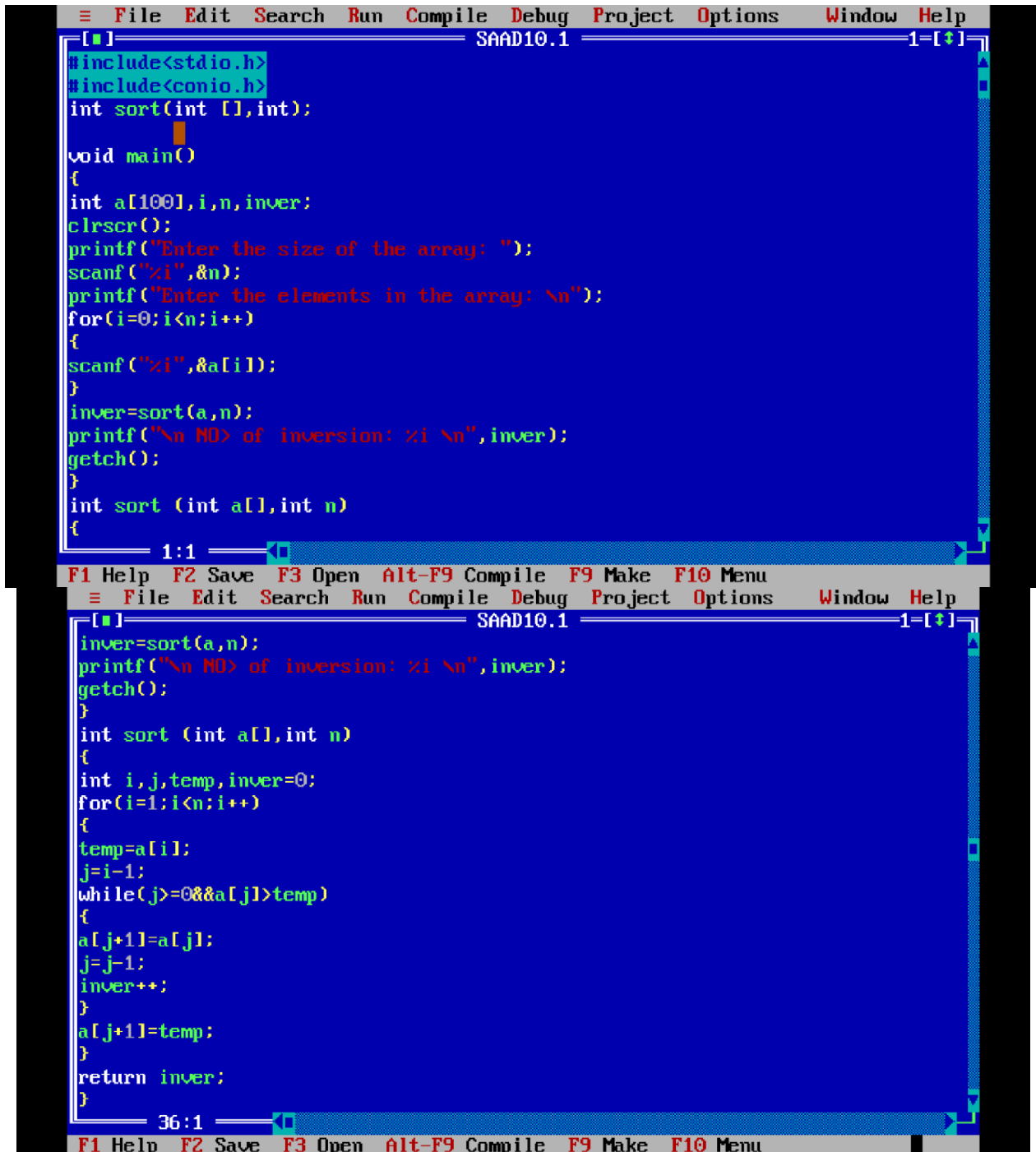


The output of the program is displayed on a black background with white text:

```
Enter size of array: 5  
Enter 5 elements:  
23  
54  
61  
12  
34  
  
Original:  
23 54 61 12 34  
  
Binary Insertion Sort:  
12 23 34 54 61
```

2. Use the Insertion Sort algorithm to count the number of inversions in an array. An inversion is a pair of elements where the earlier element is greater than the later element.

CODE:



```
#include<stdio.h>
#include<conio.h>
int sort(int [],int);

void main()
{
    int a[100],i,n,inver;
    clrscr();
    printf("Enter the size of the array: ");
    scanf("%i",&n);
    printf("Enter the elements in the array: \n");
    for(i=0;i<n;i++)
    {
        scanf("%i",&a[i]);
    }
    inver=sort(a,n);
    printf("\n NO> of inversion: %i \n",inver);
    getch();
}

int sort (int a[],int n)
{
    inver=sort(a,n);
    printf("\n NO> of inversion: %i \n",inver);
    getch();
}

int sort (int a[],int n)
{
    int i,j,temp,inver=0;
    for(i=1;i<n;i++)
    {
        temp=a[i];
        j=i-1;
        while(j>=0&& a[j]>temp)
        {
            a[j+1]=a[j];
            j=j-1;
            inver++;
        }
        a[j+1]=temp;
    }
    return inver;
}
```

OUTPUT:

```
Enter the size of the array: 5
Enter the elements in the array:
10
9
8
7
6
```

```
NO> of inversion: 10
```

Conclusion:

Marks Obtained			Dated signature of Teacher
Process Related (35)	Product Related (15)	Total (50)	