

S.No: 1

Exp. Name: **C program to insert and delete the element of one dimensional array.**

Date: 2023-04-27

Aim:

Write C program to insert and delete the element of one dimensional array

Source Code:

array.c

```
#include<stdio.h>
void main()
{
    int a[30],i,n,pos,val;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    printf("Enter the elements of the array:\n");
    for(i=0;i<n;i++)
        scanf("%d",&a[i]);
    printf("Enter the position where you want to insert an element: ");
    scanf("%d",&pos);
    printf("Enter the value to insert: ");
    scanf("%d",&val);
    for(i=n-1;i>=pos-1;i--)
    {
        a[i+1]=a[i];
    }
    a[pos]=val;
    printf("Element inserted successfully!\n");
    printf("Enter the position of the element you want to delete: ");
    scanf("%d",&pos);
    for(i=pos;i<n;i++)
    {
        a[i]=a[i+1];
    }
    printf("Element deleted successfully!\n");
    printf("Updated array:\n");
    for(i=0;i<n;i++)
    {
        printf("%d ",a[i]);
    }
    printf("\n");
}
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter the size of the array:

4

Enter the elements of the array:

1 5 2 3

Enter the position where you want to insert an element:

2

Enter the value to insert:

11

Element inserted successfully!

Enter the position of the element you want to delete:

4

Element deleted successfully!

Updated array:

1 5 11 2

Test Case - 2

User Output

Enter the size of the array:

7

Enter the elements of the array:

11 22 33 44 55 66 77

Enter the position where you want to insert an element:

1

Enter the value to insert:

88

Element inserted successfully!

Enter the position of the element you want to delete:

7

Element deleted successfully!

Updated array:

11 88 22 33 44 55 66

S.No: 4

Exp. Name: **Write a C program to Search an element using Linear Search process**

Date: 2023-03-25

Aim:

Write a program to **search** a key element with in the given array of elements using **linear search** process.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 89
Enter element for a[1] : 33
Enter element for a[2] : 56

Next, the program should print the message on the console as:

Enter key element :

if the user gives the **input** as:

Enter key element : 56

then the program should **print** the result as:

The key element 56 is found at the position 2

Similarly if the key element is given as **25** for the above one dimensional array elements then the program should print the output as "**The Key element 25 is not found in the array**".

Note: Do use the **printf()** function with a **newline** character (**\n**) at the end.

Source Code:

Program509.c

```

#include<stdio.h>
int main()
{
    int a[10],n,i,pos=-1,key;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Enter key element : ");
    scanf("%d",&key);
    for(i=0;i<n;i++)
    {
        if(a[i]==key)
        {
            pos=i;
            break;
        }
    }
    if (pos== -1)
    {
        printf("The key element %d is not found in the
array\n",key,pos);
    }
    else
    {
        printf("The key element %d is found at the position
%d\n",key,pos);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter element for a[0] :
45
Enter element for a[1] :
67
Enter element for a[2] :
35
Enter element for a[3] :
28
Enter element for a[4] :
16
Enter key element :

28

The key element 28 is found at the position 3

Test Case - 2

User Output

Enter value of n :

5

Enter element for a[0] :

2

Enter element for a[1] :

7

Enter element for a[2] :

5

Enter element for a[3] :

1

Enter element for a[4] :

4

Enter key element :

2

The key element 2 is found at the position 0

Test Case - 3

User Output

Enter value of n :

4

Enter element for a[0] :

452

Enter element for a[1] :

356

Enter element for a[2] :

754

Enter element for a[3] :

127

Enter key element :

127

The key element 127 is found at the position 3

Test Case - 4

User Output

Enter value of n :

3

Enter element for a[0] :

5

Enter element for a[1] :

7
Enter element for a[2] :
3
Enter key element :
4
The key element 4 is not found in the array

Test Case - 5	
User Output	
Enter value of n :	
3	
Enter element for a[0] :	
11	
Enter element for a[1] :	
45	
Enter element for a[2] :	
37	
Enter key element :	
25	
The key element 25 is not found in the array	

S.No: 5

Exp. Name: **Write a Program to Search an element using Linear Search and Recursion**

Date: 2023-04-14

Aim:

Write a program to **search** the given element from a list of elements with **linear search** technique using **recursion**.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 6

Next, the program should print the message on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 12 54 32 9 26

Next, the program should print the message on the console as:

Enter a key element :

if the user gives the **input** as:

Enter a key element : 9

then the program should **print** the result as:

The key element 9 is found at position : 3

Similarly, if the key element is given as **18** for the above example then the program should print the output as:

The key element 18 is not found

Note: Write the functions **read()** and **linearSearch()** in [Program911a.c](#)

Source Code:

[Program911.c](#)

```

#include <stdio.h>
#include "Program911a.c"
void main() {
    int a[20], n, pos, key;
    printf("Enter n value : ");
    scanf("%d", &n);
    read(a, n);
    printf("Enter a key element : ");
    scanf("%d", &key);
    pos = linearSearch(a, 0, n - 1, key);
    if (pos == -1) {
        printf("The key element %d is not found\n", key);
    } else {
        printf("The key element %d is found at position : %d\n", key, pos);
    }
}

```

Program911a.c

```

#include<stdio.h>
int read(int a[],int n)
{
    int i;
    printf("Enter %d elements : ",n);
    for(i=0;i<n;i++)
        scanf("%d",&a[i]);
}
int linearSearch(int a[], int i,int n,int key)
{
    int pos=0;
    if(i>=n)
    {
        return -1;
    }
    else if(a[i]==key)
    {
        pos=i;
        return pos;
    }
    else
    {
        linearSearch(a,i+1,n,key);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter n value :

4

Enter 4 elements :

10 20 15 12

Enter a key element :

15

The key element 15 is found at position : 2

Test Case - 2

User Output

Enter n value :

6

Enter 6 elements :

2 6 4 1 3 7

Enter a key element :

5

The key element 5 is not found

Test Case - 3

User Output

Enter n value :

5

Enter 5 elements :

11 44 33 55 22

Enter a key element :

11

The key element 11 is found at position : 0

Test Case - 4

User Output

Enter n value :

5

Enter 5 elements :

99 65 78 34 27

Enter a key element :

26

The key element 26 is not found

S.No: 6

Exp. Name: **Write a C program to Search an element using Binary Search process**

Date: 2023-04-14

Aim:

Write a program to **search** a key element in the given array of elements using **binary search**.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 89
Enter element for a[1] : 33
Enter element for a[2] : 56

Next, the program should print the message on the console as:

Enter key element :

if the user gives the **input** as:

Enter key element : 56

then the program should **print** the result as:

After sorting the elements in the array are
Value of a[0] = 33
Value of a[1] = 56
Value of a[2] = 89
The key element 56 is found at the position 1

Similarly if the key element is given as **25** for the above one dimensional array elements then the program should print the output as "**The Key element 25 is not found in the array**".

Note: Do use the **printf()** function with a **newline** character (**\n**) at the end.

Source Code:

Program510.c

```

#include<stdio.h>
void main()
{
    int a[20],n,low,high,key,mid,t,i,j;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Enter key element : ");
    scanf("%d",&key);
    for(i=0;i<n;i++)
    {
        for(j=0;j<n-i-1;j++)
        {
            if(a[j]>a[j+1])
            {
                t=a[j];
                a[j]=a[j+1];
                a[j+1]=t;
            }
        }
    }
    printf("After sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d\n",i,a[i]);
    }
    low=0,high=n-1;
    while(low<=high)
    {
        mid=(low+high)/2;
        if(key==a[mid])
        {
            printf("The key element %d is found at the position %d\n",key,low);
            break;
        }
        else if(key>a[mid])
        {
            low=mid+1;
        }
        else
        {
            high=mid-1;
        }
    }
    if(low>high)
    {
        printf("The Key element %d is not found in the array\n",key);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
Enter value of n :	
5	
Enter element for a[0] :	
4	
Enter element for a[1] :	
8	
Enter element for a[2] :	
6	
Enter element for a[3] :	
2	
Enter element for a[4] :	
1	
Enter key element :	
8	
After sorting the elements in the array are	
Value of a[0] = 1	
Value of a[1] = 2	
Value of a[2] = 4	
Value of a[3] = 6	
Value of a[4] = 8	
The key element 8 is found at the position 4	

Test Case - 2	
User Output	
Enter value of n :	
7	
Enter element for a[0] :	
56	
Enter element for a[1] :	
89	
Enter element for a[2] :	
63	
Enter element for a[3] :	
215	
Enter element for a[4] :	
325	
Enter element for a[5] :	
156	
Enter element for a[6] :	
256	
Enter key element :	
458	
After sorting the elements in the array are	

Value of a[0] = 56

Value of a[1] = 63

Value of a[2] = 89

Value of a[3] = 156

Value of a[4] = 215

Value of a[5] = 256

Value of a[6] = 325

The Key element 458 is not found in the array

S.No: 7

Exp. Name: **Write a Program to Search an element using Binary Search and Recursion**

Date: 2023-04-14

Aim:

Write a program to **search** the given element from a list of elements with **binary search** technique using **recursion**.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 5

Next, the program should print the following messages one by one on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 33 55 22 44 11

then the program should **print** the result as:

After sorting the elements are : 11 22 33 44 55

Next, the program should print the message on the console as:

Enter key element :

if the user gives the **input** as:

Enter key element : 11

then the program should **print** the result as:

The given key element 11 is found at position : 0

Similarly, if the key element is given as **18** for the above example then the program should print the output as:

The given key element 18 is not found

Note: Write the functions **read()**, **bubbleSort()**, **display()** and **binarySearch()** in [Program912a.c](#)

Source Code:

[Program912.c](#)

```
#include <stdio.h>
#include "Program912a.c"
void main() {
    int a[20], n, key, flag;
    printf("Enter value of n : ");
    scanf("%d", &n);
    read(a, n);
    bubbleSort(a, n);
    printf("After sorting the elements are : ");
    display(a, n);
    printf("Enter key element : ");
    scanf("%d", &key);
    flag = binarySearch(a, 0, n - 1, key);
    if (flag == -1) {
        printf("The given key element %d is not found\n", key);
    } else {
        printf("The given key element %d is found at position : %d\n", key,
flag);
    }
}
```

Program912a.c

```

int*read(int a[],int n)
{
    int i;
    printf("Enter %d elements : ",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    return a;
}
void bubbleSort(int a[],int n)
{
    int i,j,temp;
    for(i=0;i<n-1;i++)
    {
        for(j=0;j<n-i-1;j++)
        {
            if(a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
}
int*display(int a[],int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        printf("%d ",a[i]);
    }
    printf("\n");
}
int binarySearch(int a[],int low,int high,int key)
{
    int mid;
    if(low>high)
    return -1;
    else
    {
        mid=(low+high)/2;
        if(a[mid]==key)
        {
            return mid;
        }
        else if(a[mid]>key)
        {
            return binarySearch(a,low,mid-1,key);
        }
        else
        {
            return binarySearch(a,mid+1,high,key);
        }
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
Enter value of n :	
5	
Enter 5 elements :	
33 55 22 44 11	
After sorting the elements are : 11 22 33 44 55	
Enter key element :	
11	
The given key element 11 is found at position : 0	

Test Case - 2	
User Output	
Enter value of n :	
4	
Enter 4 elements :	
23 67 45 18	
After sorting the elements are : 18 23 45 67	
Enter key element :	
24	
The given key element 24 is not found	

Test Case - 3	
User Output	
Enter value of n :	
6	
Enter 6 elements :	
10 20 18 9 11 15	
After sorting the elements are : 9 10 11 15 18 20	
Enter key element :	
18	
The given key element 18 is found at position : 4	

S.No: 8

Exp. Name: **Write a C program to Sort the given elements in Ascending order using Bubble Sort**

Date: 2023-04-27

Aim:

Write a program to **sort** (**Ascending order**) the given elements using **bubble sort technique**.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 22
Enter element for a[1] : 33
Enter element for a[2] : 12

then the program should **print** the result as:

Before sorting the elements in the array are
Value of a[0] = 22
Value of a[1] = 33
Value of a[2] = 12
After sorting the elements in the array are
Value of a[0] = 12
Value of a[1] = 22
Value of a[2] = 33

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

Program504.c

```

#include<stdio.h>
void main()
{
    int a[30],i,j,n,temp;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Before sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = ",i);
        printf("%d\n",a[i]);
    }
    for(i=0;i<n;i++)
    {
        for(j=0;j<n-i-1;j++)
        {
            if(a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
    printf("After sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = ",i);
        printf("%d\n",a[i]);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter element for a[0] :
2
Enter element for a[1] :
7
Enter element for a[2] :
6
Enter element for a[3] :
4

Enter element for a[4] :

1

Before sorting the elements in the array are

Value of a[0] = 2

Value of a[1] = 7

Value of a[2] = 6

Value of a[3] = 4

Value of a[4] = 1

After sorting the elements in the array are

Value of a[0] = 1

Value of a[1] = 2

Value of a[2] = 4

Value of a[3] = 6

Value of a[4] = 7

Test Case - 2

User Output

Enter value of n :

4

Enter element for a[0] :

28

Enter element for a[1] :

34

Enter element for a[2] :

26

Enter element for a[3] :

29

Before sorting the elements in the array are

Value of a[0] = 28

Value of a[1] = 34

Value of a[2] = 26

Value of a[3] = 29

After sorting the elements in the array are

Value of a[0] = 26

Value of a[1] = 28

Value of a[2] = 29

Value of a[3] = 34

Test Case - 3

User Output

Enter value of n :

8

Enter element for a[0] :

7

Enter element for a[1] :

3

Enter element for a[2] :

9
Enter element for a[3] :
2
Enter element for a[4] :
5
Enter element for a[5] :
4
Enter element for a[6] :
6
Enter element for a[7] :
1
Before sorting the elements in the array are
Value of a[0] = 7
Value of a[1] = 3
Value of a[2] = 9
Value of a[3] = 2
Value of a[4] = 5
Value of a[5] = 4
Value of a[6] = 6
Value of a[7] = 1
After sorting the elements in the array are
Value of a[0] = 1
Value of a[1] = 2
Value of a[2] = 3
Value of a[3] = 4
Value of a[4] = 5
Value of a[5] = 6
Value of a[6] = 7
Value of a[7] = 9

Test Case - 4
User Output
Enter value of n :
4
Enter element for a[0] :
-23
Enter element for a[1] :
-14
Enter element for a[2] :
-56
Enter element for a[3] :
-35
Before sorting the elements in the array are
Value of a[0] = -23
Value of a[1] = -14
Value of a[2] = -56
Value of a[3] = -35
After sorting the elements in the array are

Value of a[0] = -56

Value of a[1] = -35

Value of a[2] = -23

Value of a[3] = -14

Test Case - 5

User Output

Enter value of n :

5

Enter element for a[0] :

28

Enter element for a[1] :

45

Enter element for a[2] :

-1

Enter element for a[3] :

-5

Enter element for a[4] :

2

Before sorting the elements in the array are

Value of a[0] = 28

Value of a[1] = 45

Value of a[2] = -1

Value of a[3] = -5

Value of a[4] = 2

After sorting the elements in the array are

Value of a[0] = -5

Value of a[1] = -1

Value of a[2] = 2

Value of a[3] = 28

Value of a[4] = 45

S.No: 9

Exp. Name: **Write a C program to Sort given elements using Quick sort**

Date: 2023-04-14

Aim:

Write a program to **sort** (**Ascending order**) the given elements using **quick sort** technique.

Note: Pick the first element as pivot. You will not be awarded marks if you do not follow this instruction.

At the time of execution, the program should print the message on the console as:

Enter array size :

For example, if the user gives the **input** as:

Enter array size : 5

Next, the program should print the following message on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 34 67 12 45 22

then the program should **print** the result as:

Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

QuickSortMain.c

```
#include <stdio.h>
#include "QuickSortFunctions.c"
void main() {
    int arr[15], i, n;
    printf("Enter array size : ");
    scanf("%d", &n);
    printf("Enter %d elements : ", n);
    for (i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Before sorting the elements are : ");
    display(arr, n);
    quickSort(arr, 0, n - 1);
    printf("After sorting the elements are : ");
    display(arr, n);
}
```

QuickSortFunctions.c

```

#include<stdio.h>
void display(int a[15], int n) {
    int i;
    for(i=0;i<n;i++)
    {
        printf("%d ",a[i]);
    }
    printf("\n");
}
int partition(int a[15], int lb, int ub){
    int i=lb+1,j,temp;
    int pos=a[lb];
    for(j=i;j<=ub;j++)
    {
        if(a[j]<pos)
        {
            temp=a[j];
            a[j]=a[i];
            a[i]=temp;
            i++;
        }
    }
    temp=a[lb];
    a[lb]=a[i-1];
    a[i-1]=temp;
    return i-1;
}
void quickSort(int a[15], int low, int high) {
    if(low<high)
    {
        int q=partition(a,low,high);
        quickSort(a,low,q-1);
        quickSort(a,q+1,high);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter array size :
5
Enter 5 elements :
34 67 12 45 22
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Test Case - 2
User Output
Enter array size :

8

Enter 8 elements :

77 55 22 44 99 33 11 66

Before sorting the elements are : 77 55 22 44 99 33 11 66

After sorting the elements are : 11 22 33 44 55 66 77 99

Test Case - 3

User Output

Enter array size :

5

Enter 5 elements :

-32 -45 -67 -46 -14

Before sorting the elements are : -32 -45 -67 -46 -14

After sorting the elements are : -67 -46 -45 -32 -14

S.No: 10

Exp. Name: **Write a C program to Sort given elements using Insertion sort**

Date: 2023-04-14

Aim:

Write a program to **sort** (**Ascending order**) the given elements using **insertion sort technique**.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 22
Enter element for a[1] : 33
Enter element for a[2] : 12

then the program should **print** the result as:

Before sorting the elements in the array are
Value of a[0] = 22
Value of a[1] = 33
Value of a[2] = 12
After sorting the elements in the array are
Value of a[0] = 12
Value of a[1] = 22
Value of a[2] = 33

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

Program505.c

```

#include<stdio.h>
void main()
{
    int a[30],i,j,n,k;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Before sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d\n",i,a[i]);
    }
    for(i=1;i<n;i++)
    {
        k=a[i];
        j=i-1;
        while(j>=0&&a[j]>k)
        {
            a[j+1]=a[j];
            j--;
        }
        a[j+1]=k;
    }
    printf("After sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    printf("Value of a[%d] = %d\n",i,a[i]);
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter element for a[0] :
7
Enter element for a[1] :
33
Enter element for a[2] :
12
Enter element for a[3] :
56
Enter element for a[4] :
9
Before sorting the elements in the array are
Value of a[0] = 7

Value of a[1] = 33

Value of a[2] = 12

Value of a[3] = 56

Value of a[4] = 9

After sorting the elements in the array are

Value of a[0] = 7

Value of a[1] = 9

Value of a[2] = 12

Value of a[3] = 33

Value of a[4] = 56

S.No: 11

Exp. Name: **Write a C program to Sort given elements using Merge sort**

Date: 2023-04-14

Aim:

Write a program to **sort** (**Ascending order**) the given elements using **merge sort** technique.

At the time of execution, the program should print the message on the console as:

Enter array size :

For example, if the user gives the **input** as:

Enter array size : 5

Next, the program should print the following message on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 34 67 12 45 22

then the program should **print** the result as:

Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

MergeSortMain.c

```
#include <stdio.h>
#include "MergeSortFunctions.c"
void main() {
    int arr[15], i, n;
    printf("Enter array size : ");
    scanf("%d", &n);
    printf("Enter %d elements : ", n);
    for (i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Before sorting the elements are : ");
    display(arr, n);
    splitAndMerge(arr, 0, n - 1);
    printf("After sorting the elements are : ");
    display(arr, n);
}
```

MergeSortFunctions.c

```

void display(int a[15], int n) {
    int i;
    for(i=0;i<n;i++)
    {
        printf("%d ",a[i]);
    }
    printf("\n");
}
void merge(int a[15], int low, int mid, int high) {
    int b[15],i,k,j;
    i=low;
    j=mid+1;
    k=low;
    while(i<=mid&&j<=high)
    {
        if(a[i]<=a[j])
        {
            b[k]=a[i];
            i++;
            k++;
        }
        else
        {
            b[k]=a[j];
            j++;
            k++;
        }
    }
    while(i<=mid)
    {
        b[k]=a[i];
        k++;
        i++;
    }
    while(j<=high)
    {
        b[k]=a[j];
        k++;
        j++;
    }
    for(i=low;i<=high;i++)
    {
        a[i]=b[i];
    }
}
void splitAndMerge(int a[15], int low, int high) {
    int mid;
    if(low<high)
    {
        mid=(low+high)/2;
        splitAndMerge(a,low,mid);
        splitAndMerge(a,mid+1,high);
        merge(a,low,mid,high);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
Enter array size :	
5	
Enter 5 elements :	
34 67 12 45 22	
Before sorting the elements are : 34 67 12 45 22	
After sorting the elements are : 12 22 34 45 67	

Test Case - 2	
User Output	
Enter array size :	
8	
Enter 8 elements :	
77 55 22 44 99 33 11 66	
Before sorting the elements are : 77 55 22 44 99 33 11 66	
After sorting the elements are : 11 22 33 44 55 66 77 99	

Test Case - 3	
User Output	
Enter array size :	
5	
Enter 5 elements :	
-32 -45 -67 -46 -14	
Before sorting the elements are : -32 -45 -67 -46 -14	
After sorting the elements are : -67 -46 -45 -32 -14	

S.No: 12

Exp. Name: **Write a C program to Create a Singly Linked List**

Date: 2023-04-27

Aim:

In the below `singly linked list` program we have two files, one file contains the **main program** and the other file contains the **functions**, to be implemented by the **user**.

Here the user has to implement the code for two functions `addNodes()` and `traverseList()`.

The `addNodes()` function creates a new list and adds elements to the list until delimiter `-1` is occurred.

Fill in the missing code in the below functions `addNodes(NODE first, int x)` and `traverseList(NODE first)` in the file `CreateAndAddNodes.c`.

Source Code:

`SingleLL1.c`

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

#include "CreateAndAddNodes.c"

void main() {
    NODE first = NULL;
    int x;
    printf("Enter elements up to -1 : ");
    scanf("%d", &x);
    while (x != -1) {
        first = addNodes(first, x);
        scanf("%d", &x);
    }
    if (first == NULL) {
        printf("Single Linked List is empty\n");
    } else {
        printf("The elements in SLL are : ");
        traverseList(first);
    }
}
```

`CreateAndAddNodes.c`

```

struct node {
    int data;
    struct node *next;
};

typedef struct node *NODE;

NODE createNode() {
    NODE temp;
    temp = (NODE)malloc(sizeof(struct node));
    temp -> next = NULL;
    return temp;
}

NODE addNodes(NODE first, int x) {
    NODE temp;
    temp=createNode();
    temp -> data=x;
    if(first==NULL)
    {
        first=temp;
    }
    else
    {
        NODE lastNode=first;
        while(lastNode->next!=NULL)
        {
            lastNode=lastNode->next;
        }
        lastNode->next=temp;
    }
    return first;
}
void traverseList(NODE first) {
    NODE temp = first;
    while(temp !=NULL)
    {
        printf("%d --> ",temp -> data);
        temp=temp->next;
    }
    printf("NULL\n");
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter elements up to -1 :
9 18 27 36 45 -1
The elements in SLL are : 9 --> 18 --> 27 --> 36 --> 45 --> NULL

Test Case - 2

User Output

Enter elements up to -1 :

12 14 19 23 -1

The elements in SLL are : 12 --> 14 --> 19 --> 23 --> NULL