1. **Write a menu driven program in C to perform array operations (Insertion, Deletion, Reversing, Searching, Sorting, Modifying, Displaying) using user defined functions.**

Program: prg1.c

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

#define MAX 20

void display(int[], int);
int insert(int[], int, int, int);
int delete(int[], int, int);
void linearSearch(int[], int, int);
void binarySearch(int[], int, int);
void sort(int[], int);
void reverse(int[], int);

int main()
{
        int arr[MAX], size, i, choice, data, position;

        printf("\n\t--: ARRAY OPERATIONS :--\n\n");

        do
        {
                printf("How many elements you want to enter [<%d]: ", MAX);
                scanf("%d", &size);

                if (size < 0 || size > MAX)
                        printf("!!! Number of elements must be smaller than or equals
to %d !!!\nTry again\n\n", MAX);
        } while (size < 0 || size > MAX);

        printf("\nEnter data one by one for array elements:\n");
        for (i = 0; i < size; i++)
        {
                printf("\tEnter for Arr[%d] = ", i);
                scanf("%d", &arr[i]);
        }

        while (1)
        {
                printf("\n\nPress Enter to continue.... ");
                fflush(stdin);
                getchar();
                system("cls");
```

```
printf("\n\t: ARRAY OPERATIONS :\n\n");
		printf(" 1. Insert an Element\n 2. Delete an Element\n 3. Search for a
Element\n 4. Sort the array\n 5. Reverse the array\n 6. Display the whole array\n 0.
Exit\n");
		printf("\nEnter corresponding numbers of your choice : ");
		scanf("%d", &choice);

		switch (choice)
		{
		case 0:
			// Exit
			printf("\n\t--- THANK YOU FOR USING THE PROGRAM ---\n");
			exit(0);

		case 1:
			// Insertion
			system("cls");
			printf("\n\t--- ELEMENT INSERTION --- \n\n");

			do
			{
				printf("Enter the position you want to insert the data (1 to %d) :
", size);
				scanf("%d", &position);

				if (position < 1 || position > size)
					printf("!!! ERROR : Invalid Position. Try Again\n\n");
			} while (position < 1 || position > size);

			printf("Enter the new element you want to insert : ");
			scanf("%d", &data);

			display(arr, size);

			size = insert(arr, size, position, data);
			printf(">> New element %d successfuly entered at position %d\n\n",
data, position);

			display(arr, size);

			break;

		case 2:
			//	Deletion
			system("cls");
			printf("\n\t--- ELEMENT DELETION --- \n\n");
```

```
				do
				{
					printf("Enter the position of the data you want to delete (1 
to %d) : ", size);
					scanf("%d", &position);

					if (position < 1 || position > size)
						printf("!!! ERROR : Invalid Position. Try Again\n\n");
				} while (position < 1 || position > size);

				display(arr, size);

				size = delete (arr, size, position);
				printf(">> Element successfuly deleted from position %d\n\n", 
position);

				display(arr, size);

				break;

			case 3:
				//	Search a Element
				system("cls");
				printf("\n\t--- SEARCH ELEMENT ---\n\n");

				printf("Which type of search you want to perform ?\n");
				printf(" 1. Linear Search (Multiple Occurence)\n");
				printf(" 2. Binary Search (First Occurence, sorting required)\n=> ");
				scanf("%d", &choice);

				printf("Enter the item you want to search : ");
				scanf("%d", &data);

				display(arr, size);

				if (choice == 1)
					linearSearch(arr, size, data);
				else if (choice == 2)
					binarySearch(arr, size, data);

				break;

			case 4:
				//	Sort array
				system("cls");
				printf("\n\t--- SORT ARRAY ---\n\n");
				printf("Before Sorting : ");
				display(arr, size);

				sort(arr, size);

				printf("After Sorting : ");
				display(arr, size);

				break;

			case 5:
				//	Reverse
				system("cls");
				printf("\n\t--- ARRAY REVERSE --- \n\n");

				printf("\nThe array before reverse: \n");
				display(arr, size);

				reverse(arr, size);

				printf("\nThe array after reverse: \n");
				display(arr, size);

				break;

			case 6:
				display(arr, size);
				break;

			default:
				printf("\n\t!!! Wrong Choice. Please enter a correct option !!!\t");
			}
	}

	return 0;
}

int insert(int arr[], int size, int position, int item)
{
	int i;

	for (i = size - 1; i >= position - 1; i--)
	{
		arr[i + 1] = arr[i];
	}

	arr[position - 1] = item;

	return size + 1;
}

int delete(int arr[], int size, int position)
{
	int i;
```

```
        for (i = position - 1; i < size - 1; i++)
        {
                arr[i] = arr[i + 1];
        }

        return size - 1;
}

void linearSearch(int arr[], int size, int item)
{
        int i, count = 0;

        printf("\n[ LINEAR SEARCH FOR = %d ]\n\n");

        for (i = 0; i < size; i++)
        {
                if (arr[i] == item)
                {
                        printf(">> Found at position = %d\n", i + 1);
                        count++;
                }
        }

        if (count == 0)
                printf("\n!!! ELEMENT NOT FOUND IN THE ARRAY !!!\n");
}

void binarySearch(int arr[], int size, int item)
{
        int i, count = 0, beg, mid, end;

        printf("\n[ BINARY SEARCH FOR = %d ]\n\n");
        printf("\n>> Sorting the array before starting binary search...");
        sort(arr, size);

        display(arr, size);

        beg = 0;
        end = size - 1;
        mid = (beg + end) / 2;

        while ((item != arr[mid]) && (beg <= end))
        {

                mid = (beg + end) / 2;

                if (arr[mid] < item)
                        beg = mid + 1;
                else if (arr[mid] > item)
                        end = mid - 1;
                else if (arr[mid] == item)
                {
                        printf("Item %d found at Position %d\n", item, mid + 1);
                        break;
}
        }

        if (arr[mid] == item)
                printf("Item %d found at Position %d\n", item, mid + 1);
        else
                printf("Item %d not found in the array\n", item);
}

void sort(int arr[], int size)
{ // assending order sorting
        int i, j, temp;

        for (i = 0; i < size - 1; i++)
        {
                for (j = 0; j < size - i - 1; j++)
                {

                        if (arr[j] > arr[j + 1])
                        {
                                temp = arr[j];
                                arr[j] = arr[j + 1];
                                arr[j + 1] = temp;
                        }
                }
        }
}

void reverse(int arr[], int size)
{
        int i, temp;

        for (i = 0; i < size / 2; i++)
        {
                temp = arr[i];
                arr[i] = arr[size - i - 1];
                arr[size - i - 1] = temp;
        }
}

void display(int arr[], int size)
{
        int i;

        printf("\nThe Array is : \n\t");

        //      upper design bar
        printf("-");
        for (i = 0; i < size; i++)
        {
                printf("----");
        }
        printf("-\n\t|");

        for (i = 0; i < size; i++)
        {
                printf(" %d |", arr[i]);
        }
```

```
	//	lower design bar
	printf("\n\t--");
	for (i = 0; i < size; i++)
	{
		printf("----");
	}

	for (i = 0; i < size; i++)
	{
		printf("\n\tArr[%d] = %d", i, arr[i]);
	}
	printf("\n");
}
```

OUTPUT:

```
          --: ARRAY OPERATIONS :--

How many elements you want to enter [<20]: 5

Enter data one by one for array elements:
          Enter for Arr[0] = 1
          Enter for Arr[1] = 6
          Enter for Arr[2] = 8
          Enter for Arr[3] = 3
          Enter for Arr[4] = 4


Press Enter to continue....


          : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 1


          --- ELEMENT INSERTION ---

Enter the position you want to insert the data (1 to 5) : 3
Enter the new element you want to insert : 10

The Array is :
          ----------------------
          | 1 | 6 | 8 | 3 | 4 |
          ----------------------
          Arr[0] = 1
          Arr[1] = 6
          Arr[2] = 8
          Arr[3] = 3
```

```
                   Arr[4] = 4
>> New element 10 successfuly entered at position 3


The Array is :
        -------------------------
        | 1 | 6 | 10 | 8 | 3 | 4 |
        -------------------------
        Arr[0] = 1
        Arr[1] = 6
        Arr[2] = 10
        Arr[3] = 8
        Arr[4] = 3
        Arr[5] = 4


Press Enter to continue....


        : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 2


        --- ELEMENT DELETION ---

Enter the position of the data you want to delete (1 to 6) : 2

The Array is :
        -------------------------
        | 1 | 6 | 10 | 8 | 3 | 4 |
        -------------------------
        Arr[0] = 1
        Arr[1] = 6
        Arr[2] = 10
        Arr[3] = 8
        Arr[4] = 3
        Arr[5] = 4
        >> Element successfuly deleted from position 2


The Array is :
        ----------------------
        | 1 | 10 | 8 | 3 | 4 |
        ----------------------
        Arr[0] = 1
        Arr[1] = 10
        Arr[2] = 8
        Arr[3] = 3
        Arr[4] = 4


Press Enter to continue....



        : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 3

        --- SEARCH ELEMENT ---

Which type of search you want to perform ?
 1. Linear Search (Multiple Occurence)
 2. Binary Search (First Occurence, sorting required)
=> 1
Enter the item you want to search : 10

The Array is :
        ----------------------
        | 1 | 10 | 8 | 3 | 4 |
        ----------------------
        Arr[0] = 1
        Arr[1] = 10
        Arr[2] = 8
        Arr[3] = 3
        Arr[4] = 4
```

```
[ LINEAR SEARCH FOR = 5 ]

>> Found at position = 2


Press Enter to continue....


          : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 3


          --- SEARCH ELEMENT ---

Which type of search you want to perform ?
 1. Linear Search (Multiple Occurence)
 2. Binary Search (First Occurence, sorting required)
=> 2
Enter the item you want to search : 4

The Array is :
          ----------------------
          | 1 | 10 | 8 | 3 | 4 |
          ----------------------
          Arr[0] = 1
          Arr[1] = 10
          Arr[2] = 8
          Arr[3] = 3
          Arr[4] = 4

[ BINARY SEARCH FOR = 5 ]


>> Sorting the array before starting binary search...
The Array is :
```

```
          ----------------------
          | 1 | 3 | 4 | 8 | 10 |
          ----------------------
          Arr[0] = 1
          Arr[1] = 3
          Arr[2] = 4
          Arr[3] = 8
          Arr[4] = 10

Item 4 found at Position 3

Press Enter to continue....


          : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 4


          --- SORT ARRAY ---

Before Sorting :
The Array is :
          ----------------------
          | 1 | 3 | 4 | 8 | 10 |
          ----------------------
          Arr[0] = 1
          Arr[1] = 3
          Arr[2] = 4
          Arr[3] = 8
          Arr[4] = 10

After Sorting :
The Array is :
          ----------------------
          | 1 | 3 | 4 | 8 | 10 |
          ----------------------
          Arr[0] = 1
          Arr[1] = 3
          Arr[2] = 4
          Arr[3] = 8
          Arr[4] = 10

Press Enter to continue....


          : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 5

          --- ARRAY REVERSE ---


The array before reverse:

The Array is :
          ----------------------
          | 1 | 3 | 4 | 8 | 10 |
          ----------------------
          Arr[0] = 1
          Arr[1] = 3
          Arr[2] = 4
          Arr[3] = 8
          Arr[4] = 10

The array after reverse:

The Array is :
          ----------------------
          | 10 | 8 | 4 | 3 | 1 |
          ----------------------
          Arr[0] = 10
          Arr[1] = 8
          Arr[2] = 4
```

```
        Arr[3] = 3
        Arr[4] = 1

Press Enter to continue....



        : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 6

The Array is :
        ----------------------
        | 10 | 8 | 4 | 3 | 1 |
        ----------------------
        Arr[0] = 10
        Arr[1] = 8
        Arr[2] = 4
        Arr[3] = 3
        Arr[4] = 1


Press Enter to continue....



        : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 7

        !!! Wrong Choice. Please enter a correct option !!!
```

```
Press Enter to continue....


        : ARRAY OPERATIONS :

 1. Insert an Element
 2. Delete an Element
 3. Search for a Element
 4. Sort the array
 5. Reverse the array
 6. Display the whole array
 0. Exit

Enter corresponding numbers of your choice : 0

        --- THANK YOU FOR USING THE PROGRAM ---

--------------------------------
Process exited after 2230 seconds with return value 0
Press any key to continue . . .
```