Source Code

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

void merge(int *arr, int left, int mid, int right)
{
    int i, j, k;
    int left_size = mid - left + 1;
    int right_size = right - mid;
    int L[left_size], R[right_size];

    for (i = 0; i < left_size; i++)
        L[i] = arr[left + i];
    for (j = 0; j < right_size; j++)
        R[j] = arr[mid + 1 + j];

    i = 0;
    j = 0;
    k = left;

    while (i < left_size && j < right_size)
    {
        if (L[i] <= R[j])
        {
            arr[k] = L[i];
            i++;
        }
        else
        {
            arr[k] = R[j];
            j++;
        }
        k++;
    }

    while (i < left_size)
    {
        arr[k] = L[i];
        i++;
        k++;
    }

    while (j < right_size)
    {
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergeSort(int arr[], int left, int right)
```

```
{
    if (left < right)
    {
        int mid = (right + left) / 2;

        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);

        merge(arr, left, mid, right);
    }
}

int binarySearch(int arr[], int left, int right, int x)
{
    while (left <= right)
    {
        int mid = left + (right - left) / 2;

        if (arr[mid] == x)
            return mid;

        if (arr[mid] < x)
            left = mid + 1;
        else
            right = mid - 1;
    }
    return -1;
}

int main()
{
    int choice, size, *arr, search_data, result;
    while (1)
    {
        printf("\nEnter option:\n (0) merge sort\n (1) binary search\n");
        scanf("%d", &choice);

        printf("Enter size of array to sort/search\n");
        scanf("%d", &size);
        arr = (int *)malloc(sizeof(int) * size);
        printf("Enter array in unsorted form\n");
        for (int i = 0; i < size; i++)
            scanf("%d", &arr[i]);

        switch (choice)
        {
        // Cases 0 and 1 both use merge sort, so I merged them ;)
        case 0:
        case 1:
            mergeSort(arr, 0, size - 1);
            printf("Sorted array using Merge Sort:\n");
            for (int i = 0; i < size; i++)
```

```
                printf("%d ", arr[i]);
            printf("\n");
            if (choice == 0)
                break;

            printf("Enter data to search: ");
            scanf("%d", &search_data);
            result = binarySearch(arr, 0, size - 1, search_data);
            if (result != -1)
                printf("Element %d is present at index %d\n", search_data, result);
            else
                printf("Element %d is not present in the array\n", search_data);
            break;
        }
        free(arr);
    }
    return 0;
}
```

Output

```
Enter option:
 (0) merge sort
 (1) binary search
0
Enter size of array to sort/search
4
Enter array in unsorted form
3 5 2 1
Sorted array using Merge Sort:
1 2 3 5

Enter option:
 (0) merge sort
 (1) binary search
1
Enter size of array to sort/search
6
Enter array in unsorted form
4 3 2 5 1 6
Sorted array using Merge Sort:
1 2 3 4 5 6
Enter data to search: 6
Element 6 is present at index 5
```