



Day 5

Student Name: Sweta singh

UID: 22BCS10664

Branch: CSE 3rdyear

Section/Group: KPIT-901 A

Q1. Searching a Number

Given an integer k and array arr. Your task is to return the position of the first occurrence of k in the given array and if element k is not present in the array then return -1.

Code :

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

```
int searchNumber(int arr[], int size, int k) {  
    for (int i = 0; i < size; i++) {  
        if (arr[i] == k) {  
            return i;  
        }  
    }  
    return -1;  
}
```

```
int main() {  
    int n, k;  
    printf("Enter the size of the array: ");  
    scanf("%d", &n);  
    int arr[n];  
    printf("Enter the elements of the array:\n");  
    for (int i = 0; i < n; i++) {  
        scanf("%d", &arr[i]);  
    }  
    printf("Enter the number to search: ");  
    scanf("%d", &k);  
    int position = searchNumber(arr, n, k);
```



```
if(position != -1) {  
    printf("The first occurrence of %d is at position: %d\n", k, position);  
} else {  
    printf("Element %d is not present in the array.\n", k);  
}  
  
return 0;  
}
```

Output :

Q2. Sorted array Search.

Given an array, arr[] sorted in ascending order and an integer k. Return true if k is present in the array, otherwise, false.

Code:

```
#include <stdio.h>  
#include <stdbool.h>  
  
bool binarySearch(int arr[], int size, int k) {  
    int left = 0, right = size - 1;  
  
    while (left <= right) {  
        int mid = left + (right - left) / 2;  
  
        if (arr[mid] == k) {  
            return true;  
        }  
  
        if (arr[mid] < k) {  
            left = mid + 1;  
        }  
    }  
}
```



DEPARTMENT OF

Discover. Learn. Empower.

```
} else {  
    right = mid - 1;  
}  
}  
return false;  
}  
  
int main() {  
    int n, k;  
    printf("Enter the size of the array: ");  
    scanf("%d", &n);  
  
    int arr[n];  
    printf("Enter the elements of the sorted array:\n");  
    for (int i = 0; i < n; i++) {  
        scanf("%d", &arr[i]);  
    }  
    printf("Enter the number to search: ");  
    scanf("%d", &k);  
    if (binarySearch(arr, n, k)) {  
        printf("Element %d is present in the array.\n", k);  
    } else {  
        printf("Element %d is not present in the array.\n", k);  
    }  
  
    return 0;  
}
```

Output:



```
input
Enter the size of the array: 4
Enter the elements of the sorted array:
1
2
3
4
Enter the number to search: 3
Element 3 is present in the array.

...Program finished with exit code 0
Press ENTER to exit console.
```

Q3. Find Target Indices After Sorting Array.

You are given a 0-indexed integer array `nums` and a target element `target`.

A target index is an index `i` such that `nums[i] == target`.

Return a list of the target indices of `nums` after sorting `nums` in non-decreasing order. If there are no target indices, return an empty list. The returned list must be sorted in increasing order.

Code:

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

int compare(const void *a, const void *b) {
    return (*(int *)a - *(int *)b);
}

void findTargetIndices(int nums[], int size, int target) {

    qsort(nums, size, sizeof(int), compare);

    int found = 0;

    printf("Target indices: ");

    for (int i = 0; i < size; i++) {
        if (nums[i] == target) {
            printf("%d ", i);
            found = 1;
        }
    }

    if (!found) {
        printf("No target indices found.");
    }
}
```



DEPARTMENT OF

Discover. Learn. Empower.

```
printf("\n");  
}  
  
int main() {  
    int n, target;  
  
    printf("Enter the size of the array: ");  
    scanf("%d", &n);  
  
    int nums[n];  
    printf("Enter the elements of the array:\n");  
    for (int i = 0; i < n; i++) {  
        scanf("%d", &nums[i]);  
    }  
    printf("Enter the target element: ");  
    scanf("%d", &target);  
    findTargetIndices(nums, n, target);  
  
    return 0;  
}
```

Output:

A screenshot of a console window titled 'Input'. The window has a black background with white text. The text shows the program's execution: 'Enter the size of the array: 4', 'Enter the elements of the array: 3267', '4', '6', '7', 'Enter the target element: 7', 'Target indices: 2'. At the bottom, it says '...Program finished with exit code 0' and 'Press ENTER to exit console.' in green text.

```
Input  
Enter the size of the array: 4  
Enter the elements of the array:  
3267  
4  
6  
7  
Enter the target element: 7  
Target indices: 2  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

Q4. . Search Insert Position.



Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order. You must write an algorithm with $O(\log n)$ runtime complexity.

Code:

```
#include <stdio.h>

int searchInsertPosition(int arr[], int size, int target) {
    int left = 0, right = size - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;

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

    return left;
}

int main() {
    int n, target;
    printf("Enter the size of the array: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter the sorted array elements:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
}
```



DEPARTMENT OF

Discover. Learn. Empower.

```
printf("Enter the target value: ");
scanf("%d", &target);

int position = searchInsertPosition(arr, n, target);

printf("The target would be inserted at index: %d\n", position);

return 0;
}
```

Output :

Q5. Relative Sort Array.

Given two arrays arr1 and arr2 the element of arr2 are distinct elements in arr2 are also in arr1.

Sort the Element of arr1 such that the relative ordering of item are the same as in arr2. Element that do not appear in arr2 should be placed at the end of arr1 in ascending order.

Code:

```
#include <stdio.h>

#include <stdlib.h>

int compare(const void *a, const void *b) {
    return (*(int *)a - *(int *)b);
}

void relativeSortArray(int arr1[], int n1, int arr2[], int n2) {
    int output[n1], visited[n1], index = 0;
    for (int i = 0; i < n1; i++) {
        visited[i] = 0;
```



DEPARTMENT OF

Discover. Learn. Empower.

```
for (int i = 0; i < n2; i++) {
    for (int j = 0; j < n1; j++) {
        if (!visited[j] && arr1[j] == arr2[i]) {
            output[index++] = arr1[j];
            visited[j] = 1;
        }
    }
}

for (int i = 0; i < n1; i++) {
    if (!visited[i]) {
        output[index++] = arr1[i];
    }
}

qsort(output + index - (n1 - index), n1 - index, sizeof(int), compare);

for (int i = 0; i < n1; i++) {
    arr1[i] = output[i];
}

int main() {
    int n1, n2;
    printf("Enter the size of arr1: ");
    scanf("%d", &n1);
    int arr1[n1];
    printf("Enter the elements of arr1:\n");
    for (int i = 0; i < n1; i++) {
        scanf("%d", &arr1[i]);
    }
    printf("Enter the size of arr2: ");
    scanf("%d", &n2);
    int arr2[n2];
    printf("Enter the elements of arr2:\n");
```




DEPARTMENT OF

Discover. Learn. Empower.

```
for (int i = 0; i < n2; i++) {  
    scanf("%d", &arr2[i]);  
}
```

```
relativeSortArray(arr1, n1, arr2, n2);
```

```
printf("The sorted array is:\n");
```

```
for (int i = 0; i < n1; i++) {
```

```
    printf("%d ", arr1[i]);
```

```
}
```

```
printf("\n");
```

```
return 0;
```

```
}
```

Output :

A screenshot of a terminal window titled 'input'. The window shows the execution of a C++ program. The user enters the size of array arr1 as 3, followed by its elements 1, 2, and 3. Then, the user enters the size of array arr2 as 4, followed by its elements 6, 7, 8, and 9. The program then prints 'The sorted array is:' followed by the elements 1, 2, and 3. At the bottom, it says '...Program finished with exit code 0' and 'Press ENTER to exit console.'

```
input  
Enter the size of arr1: 3  
Enter the elements of arr1:  
1  
2  
3  
Enter the size of arr2: 4  
Enter the elements of arr2:  
6  
7  
8  
9  
The sorted array is:  
1 2 3  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```