**DAY-4** **USE POINTERS**

**Quiz-1**

**1.Swapping of two Numbers** by
a)Call By Value
b)Call By Reference

**a) Call By Value**

**CODE:**

```
#include <stdio.h>
void swapByValue(int a, int b) {
    int temp = a;
    a = b;
    b = temp;
}
int main() {
    int num1, num2;
    printf("Enter value for num1: ");
    scanf("%d", &num1);
    printf("Enter value for num2: ");
    scanf("%d", &num2);
    printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);
    swapByValue(num1, num2);
    printf("After swapping (Call By Value): num1 = %d, num2 = %d\n", num1, num2);
   return 0;
}
```

**OUTPUT:**

```
Enter value for num1: 3
Enter value for num2: 26
Before swapping: num1 = 3, num2 = 26
After swapping (Call By Value): num1 = 3, num2 = 26
```

**b) Call By Reference**

**CODE:**

```
#include <stdio.h>
void swapByReference(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

int main() {
    int num1, num2;
```

```
    printf("Enter value for num1: ");
    scanf("%d", &num1);
    printf("Enter value for num2: ");
    scanf("%d", &num2);
    printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);
    swapByReference(&num1, &num2);
    printf("After swapping (Call By Reference): num1 = %d, num2 = %d\n", num1, num2);
return 0;
}
```

**OUTPUT:**

```
Enter value for num1: 26
Enter value for num2: 3
Before swapping: num1 = 26, num2 = 3
After swapping (Call By Reference): num1 = 3, num2 = 26
```

**2.Find duplicates in an array**

Given an array a of size N which contains elements from 0 to N-1, you need to find all the elements occurring more than once in the given array. Return the answer in ascending order. If no such element is found, return list containing [-1].

Example 1:
Input:
N = 4
a[] = {0,3,1,2}
Output:
-1
Explanation: There is no repeating element in the array. Therefore output is -1.

Example 2:
Input:
N = 5
a[] = {2,3,1,2,3}
Output:
2 3
Explanation: 2 and 3 occur more than once in the given array.

**CODE:**

```
#include <stdio.h>
#include <stdlib.h>
int* findDuplicates(int arr[], int N, int* resultSize) {
    int* freq = (int*)calloc(N, sizeof(int));
    for (int i = 0; i < N; i++) {
        freq[arr[i]]++;
    }
```

```
    int countDuplicates = 0;
    for (int i = 0; i < N; i++) {
        if (freq[i] > 1) {
            countDuplicates++;
        }
    }

    if (countDuplicates == 0) {
        *resultSize = 1;
        int* result = (int*)malloc(sizeof(int));
        result[0] = -1;
        free(freq);
        return result;
    }
    int* result = (int*)malloc(countDuplicates * sizeof(int));
    int index = 0;
    for (int i = 0; i < N; i++) {
        if (freq[i] > 1) {
            result[index++] = i;
        }
    }
    *resultSize = countDuplicates;
    free(freq);

    return result;
}
int main() {
    int N;
    printf("Enter the size of the array: ");
    scanf("%d", &N);

    int arr[N];
    printf("Enter the elements of the array: ");
    for (int i = 0; i < N; i++) {
        scanf("%d", &arr[i]);
    }

    int resultSize;
    int* result = findDuplicates(arr, N, &resultSize);

    printf("Duplicate elements in ascending order: ");
    for (int i = 0; i < resultSize; i++) {
        printf("%d ", result[i]);
    }

    free(result);

    return 0;
}
```

**OUTPUT:**

**Example 1:**

```
Enter the size of the array: 4
Enter the elements of the array:
0
3
1
2
Duplicate elements in ascending order: -1
```

**Example 2:**

```
Enter the size of the array: 5
Enter the elements of the array:
2
3
1
2
3
Duplicate elements in ascending order: 2 3
```

**3.Union of Two Sorted Arrays**

Union of two arrays can be defined as the common and distinct elements in the two arrays. Given two sorted arrays of size n and m respectively, find their union.

Example 1:
Input:
n = 5, arr1[] = {1, 2, 3, 4, 5}
m = 3, arr2 [] = {1, 2, 3}
Output: 1 2 3 4 5
Explanation: Distinct elements including
both the arrays are: 1 2 3 4 5.

Example 2:
Input:
n = 5, arr1[] = {2, 2, 3, 4, 5}
m = 5, arr2[] = {1, 1, 2, 3, 4}
Output: 1 2 3 4 5
Explanation: Distinct elements including
both the arrays are: 1 2 3 4 5

**CODE:**

```
#include <stdio.h>
void findUnion(int arr1[], int n, int arr2[], int m) {
   int i = 0, j = 0;
  while (i < n && j < m) {
      if (i > 0 && arr1[i] == arr1[i - 1]) {
         // Skip duplicates in the first array
         i++;
         continue;
      }

      if (j > 0 && arr2[j] == arr2[j - 1]) {
         // Skip duplicates in the second array
         j++;
         continue;
      }
      if (arr1[i] < arr2[j]) {
         printf("%d ", arr1[i]);
         i++;
      } else if (arr1[i] > arr2[j]) {
         printf("%d ", arr2[j]);
         j++;
      } else {
         // Both elements are equal, print any and move both pointers
         printf("%d ", arr1[i]);
         i++;
         j++;
      }
   }
   while (i < n) {
      if (i == 0 || (i > 0 && arr1[i] != arr1[i - 1])) {
         printf("%d ", arr1[i]);
      }
      i++;
   }
   while (j < m) {
      if (j == 0 || (j > 0 && arr2[j] != arr2[j - 1])) {
         printf("%d ", arr2[j]);
      }
      j++;
   }
}

int main() {
   int n, m;

   printf("Enter the size of the first array: ");
   scanf("%d", &n);
   int arr1[n];
```

```
    printf("Enter the elements of the first array: ");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr1[i]);
    }

    printf("Enter the size of the second array: ");
    scanf("%d", &m);
    int arr2[m];
    printf("Enter the elements of the second array: ");
    for (int i = 0; i < m; i++) {
        scanf("%d", &arr2[i]);
    }

    printf("Union of the two arrays: ");
    findUnion(arr1, n, arr2, m);

    return 0;
}
```

**OUTPUT:**

**Example 1:**

```
Enter the size of the first array: 5
Enter the elements of the first array: 1 2 3 4 5
Enter the size of the second array: 3
Enter the elements of the second array: 1 2 3
Union of the two arrays: 1 2 3 4 5
```

**Example 2:**

```
Enter the size of the first array: 5
Enter the elements of the first array: 2 2 3 4 5
Enter the size of the second array: 5
Enter the elements of the second array: 1 1 2 3 4
Union of the two arrays: 1 2 3 4 5
```