**POINTERS**

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

**CODE:**

**a)Call By Value**

#include <stdio.h>

void swapByValue(int a, int b) {

int temp = a;

a = b;

b = temp;

}

int main() {

int num1,num2;

printf("Enter number 1:");

scanf("%d",&num1);

printf("Enter number 2:");

scanf("%d",&num2);

printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

swapByValue(num1, num2);

printf("After swapping : num1 = %d, num2 = %d\n", num1, num2);

return 0;

}

**b)Call By Reference**

#include <stdio.h>

void swap(int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

int main() {

int num1;

int num2;

printf("Enter number 1:");

scanf("%d",&num1);

printf("Enter number 2:");

scanf("%d",&num2);

printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

swap(&num1, &num2);

printf("After swapping: num1 = %d, num2 = %d\n", num1, num2);

return 0;

}

**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>

void findDuplicates(int arr[], int n) {

int count[n];

for (int i = 0; i < n; i++) {

count[i] = 0;

}

for (int i = 0; i < n; i++) {

count[arr[i]]++;

}

int flag = 0;

printf("Duplicate elements: ");

for (int i = 0; i < n; i++) {

if (count[i] > 1) {

printf("%d ", i);

flag = 1;

}

}

if (!flag) {

printf("-1");

}

printf("\n");

}

int main() {

int n;

printf("Enter the size of the array: ");

scanf("%d", &n);

int arr[n];

printf("Enter the elements of the array (0 to %d): ", n - 1);

for (int i = 0; i < n; i++) {

scanf("%d", &arr[i]);

}

findDuplicates(arr, n);

return 0;

}

**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 printUnion(int arr1[], int n, int arr2[], int m) {

int i = 0, j = 0;

printf("Union of two arrays: ");

while (i < n && j < m) {

if (arr1[i] < arr2[j]) {

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

i++;

} else if (arr2[j] < arr1[i]) {

printf("%d ", arr2[j]);

j++;

} else {

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

i++;

j++;

}

}

while (i < n) {

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

i++;

}

while (j < m) {

printf("%d ", arr2[j]);

j++;

}

printf("\n");

}

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]);

}

printUnion(arr1, n, arr2, m);

return 0;

}