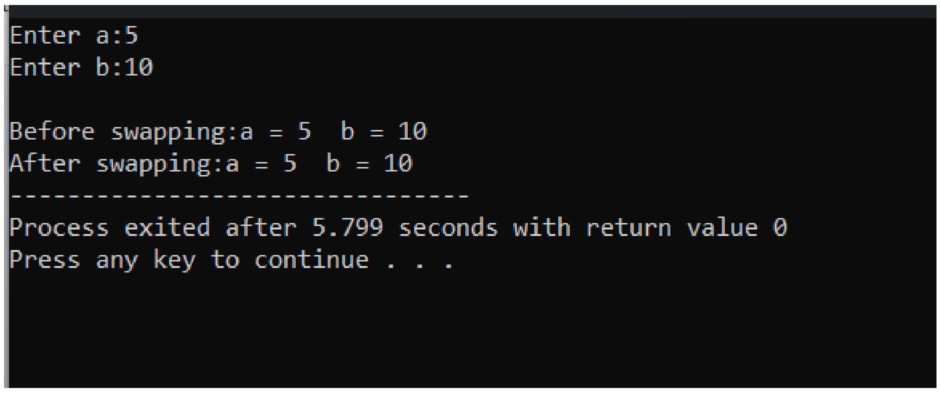
1.Swapping of two Numbers

a)Call By Value

```
#include<stdio.h>
void swap(int x , int y)
{
        int temp;
        temp = x;
        x = y;
        y = temp;
}
int main()
{
        int a,b;
        printf("Enter a:");
        scanf("%d",&a);
        printf("Enter b:");
        scanf("%d",&b);
        printf("\nBefore swapping:");
        printf("a = %d  b = %d",a,b);
        swap(a,b);
        printf("\nAfter swapping:");
        printf("a = %d  b = %d",a,b);
        return 0;
}
```

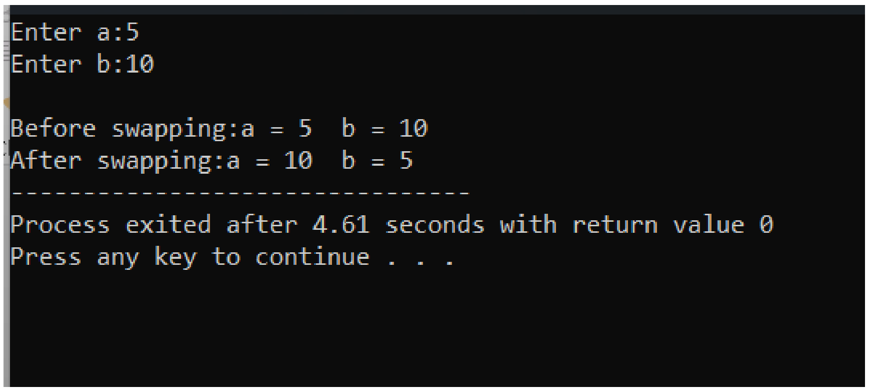


```
Enter a:5
Enter b:10

Before swapping:a = 5  b = 10
After swapping:a = 5  b = 10
--------------------------------
Process exited after 5.799 seconds with return value 0
Press any key to continue . . .
```

In call by value the values will not get swapped in the main function . The values will be swapped only in the swap() function.

b)Call By Reference

```
#include<stdio.h>
void swap(int *x , int *y)
{
	int temp;
	temp = *x;
	*x = *y;
	*y = temp;
}
int main()
{
	int a,b;
	printf("Enter a:");
	scanf("%d",&a);
	printf("Enter b:");
	scanf("%d",&b);
	printf("\nBefore swapping:");
	printf("a = %d  b = %d",a,b);
	swap(&a,&b);
	printf("\nAfter swapping:");
	printf("a = %d  b = %d",a,b);
	return 0;
}
```



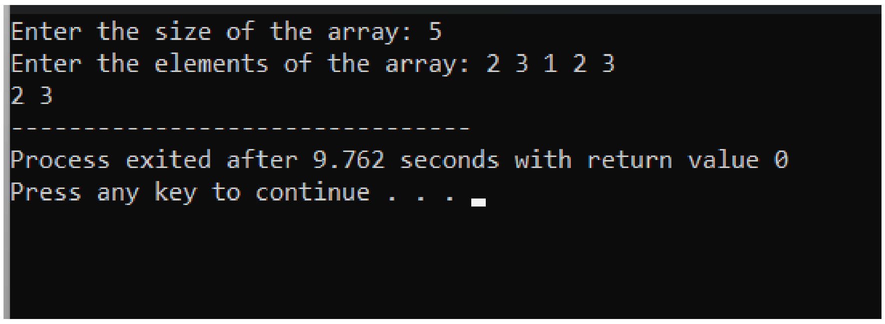
```
Enter a:5
Enter b:10

Before swapping:a = 5  b = 10
After swapping:a = 10  b = 5
--------------------------------
Process exited after 4.61 seconds with return value 0
Press any key to continue . . .
```

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.

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

int* find_duplicates(int arr[], int n, int* result_size)
{
   int* duplicates = NULL;
   int i;
   *result_size = 0;

   for (i = 0; i < n; ++i)
   {
      int index = abs(arr[i]);
      if (arr[index] >= 0)
      {
         arr[index] = -arr[index];
       }
      else
      {
         duplicates = realloc(duplicates, (*result_size + 1) * sizeof(int));
         duplicates[(*result_size)++] = index;
      }
   }

   return duplicates;
}

int main()
{
   int N, i ,result_size;
   printf("Enter the size of the array: ");
   scanf("%d", &N);
   int* a = (int*)malloc(N * sizeof(int));
   printf("Enter the elements of the array: ");
   for (i = 0; i < N; ++i)
   {
      scanf("%d", &a[i]);
   }
int* output = find_duplicates(a, N, &result_size);
if (result_size > 0)
{
    for (i = 0; i < result_size; ++i)
    {
        printf("%d ", output[i]);
    }

}
else
{
    printf("-1");
}

  return 0;
}
```



```
Enter the size of the array: 5
Enter the elements of the array: 2 3 1 2 3
2 3
--------------------------------
Process exited after 9.762 seconds with return value 0
Press any key to continue . . .
```

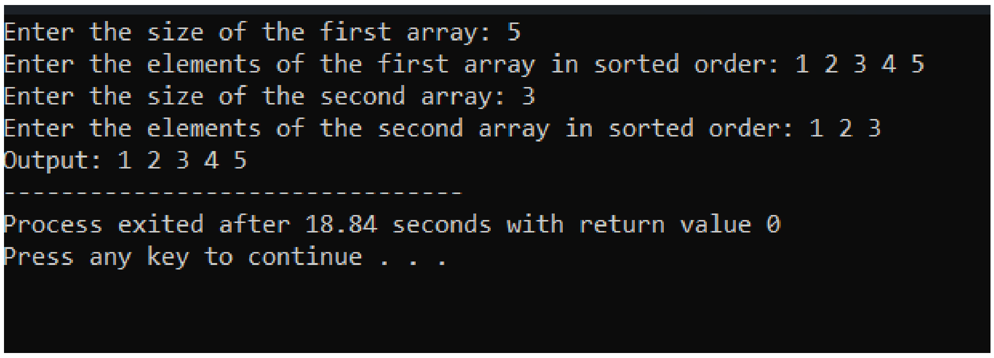
3.Union of Two Sorted Arrays

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.

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

void printUnion(int arr1[], int m, int arr2[], int n)
{
  int i;
  int *ptr1 = arr1;
  int *ptr2 = arr2;

  while (ptr1 - arr1 < m && ptr2 - arr2 < n)
  {
    if (*ptr1 < *ptr2)
    {
        printf("%d ", *ptr1);
        ptr1++;
    }
    else if (*ptr2 < *ptr1)
    {
        printf("%d ", *ptr2);
        ptr2++;
    }
    else
    {
        printf("%d ", *ptr1);
        ptr1++;
        ptr2++;
    }
  }


  while (ptr1 - arr1 < m)
  {
    printf("%d ", *ptr1);
    ptr1++;
  }

  while (ptr2 - arr2 < n)
  {
    printf("%d ", *ptr2);
    ptr2++;
  }
}

int main()
{
  int n, m, i;
  printf("Enter the size of the first array: ");
  scanf("%d", &m);

  int *arr1 = (int *)malloc(m * sizeof(int));

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

  printf("Enter the size of the second array: ");
  scanf("%d", &n);

  int *arr2 = (int *)malloc(n * sizeof(int));


  printf("Enter the elements of the second array in sorted order: ");
```

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

  printf("Output: ");
  printUnion(arr1, m, arr2, n);


  free(arr1);
  free(arr2);

  return 0;
}
```



```
Enter the size of the first array: 5
Enter the elements of the first array in sorted order: 1 2 3 4 5
Enter the size of the second array: 3
Enter the elements of the second array in sorted order: 1 2 3
Output: 1 2 3 4 5
--------------------------------
Process exited after 18.84 seconds with return value 0
Press any key to continue . . .
```