Computer Engineering Department

TED University

CMPE 252 - C Programming, Spring 2023-2024

Lab 1

In this lab, you will use the following functions to be implemented for the prelab.

```
void readInput(int arr[], int *nPtr); // reads numbers from the standard input into
arr, and stores the number of read elements in the memory cell pointed to by nPtr

void printNumbers(const int arr[], int n); // prints the elements in arr[0..(n-1)]
```

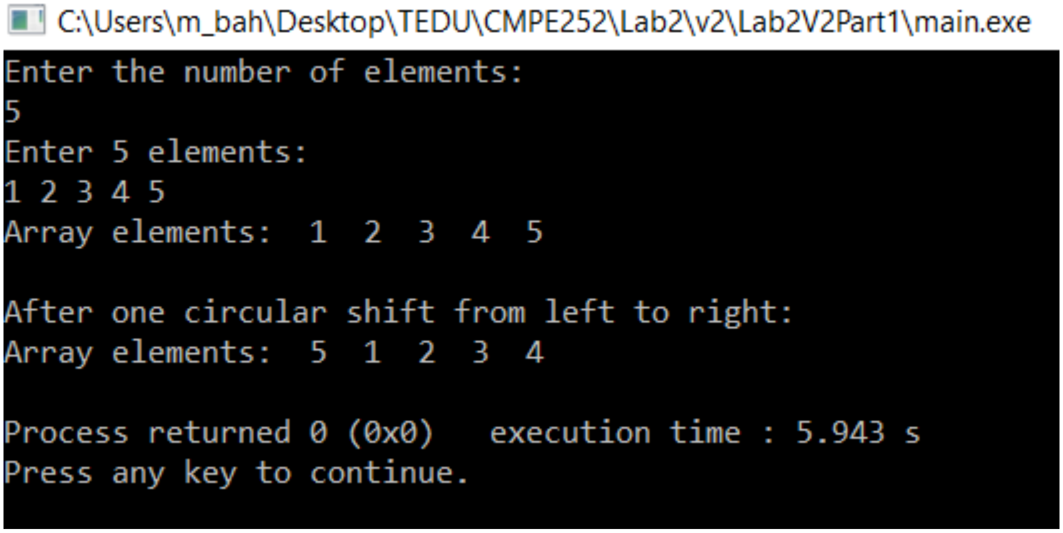
Part 1 (30 points)

Implement the following function in skeleton code `lab1part1.c`:

```
// Circularly shift elements of arr from left to right where last element of arr is
// shifted the first position of arr.
// Size of arr is n.
void circularShiftFromLeftToRight(int arr[], int n);
```

Your task in this part to fill in the missing function definitions in skeleton code `lab1part1.c`.
`main` function will stay as it is.

Sample Run:



C:\Users\m_bah\Desktop\TEDU\CMPE252\Lab2\v2\Lab2V2Part1\main.exe

```
Enter the number of elements:
5
Enter 5 elements:
1 2 3 4 5
Array elements:  1  2  3  4  5

After one circular shift from left to right:
Array elements:  5  1  2  3  4

Process returned 0 (0x0)   execution time : 5.943 s
Press any key to continue.
```

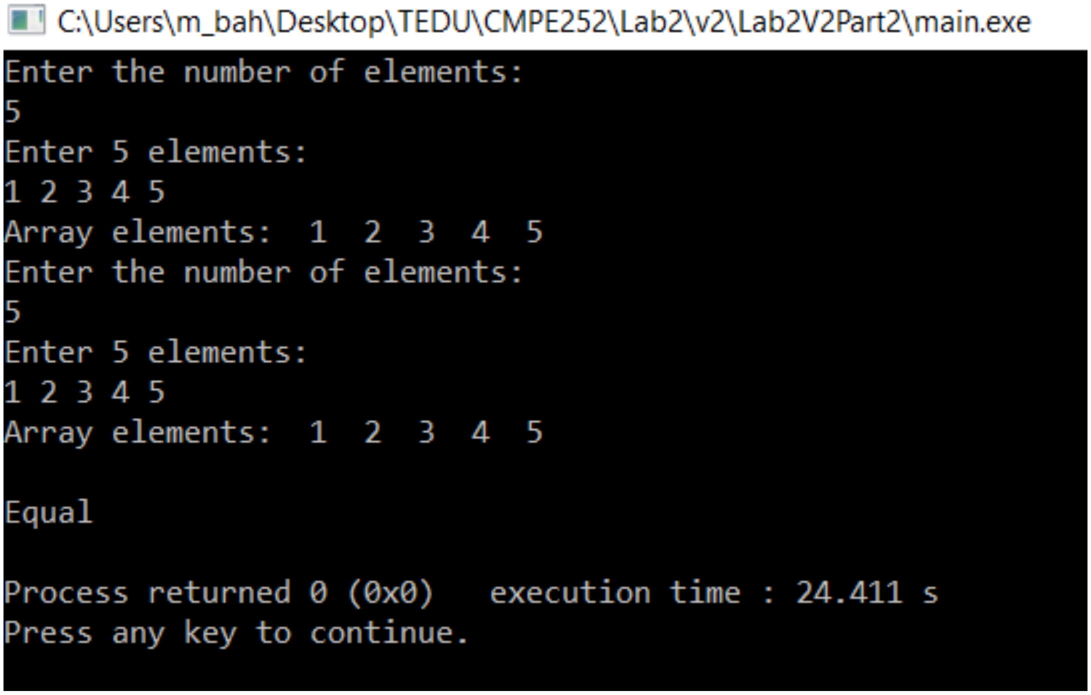
Part 2 (35 points)

Implement the following function in skeleton code `lab1part2.c`:

```
// Let n be the minimum of n1 and n2 where n1 and n2 are the number of elements in
// arr1 and arr2, respectively.
// Compare corresponding elements of arr1 and arr2 at each of the first n positions.
//
// If arr1 and arr2 has the same value in each position:
// Return 0 if n1 == n2
// Return 1 if n1 > n2
// Return -1 if n1 < n2
//
// If arr1 has a larger value than arr2 considering the first position from the
// beginning at which arr1 and arr2 have a different value:
// Return 1
//
// If arr1 has a smaller value than arr2 considering the first position from the
// beginning at which arr1 and arr2 have a different value:
// Return -1
int compareTwoArrays(const int arr1[], const int arr2[], int n1, int n2);
```

Your task in this part to fill in the missing function definitions in skeleton code `lab1part2.c`. `main` function will stay as it is.

Sample Run:

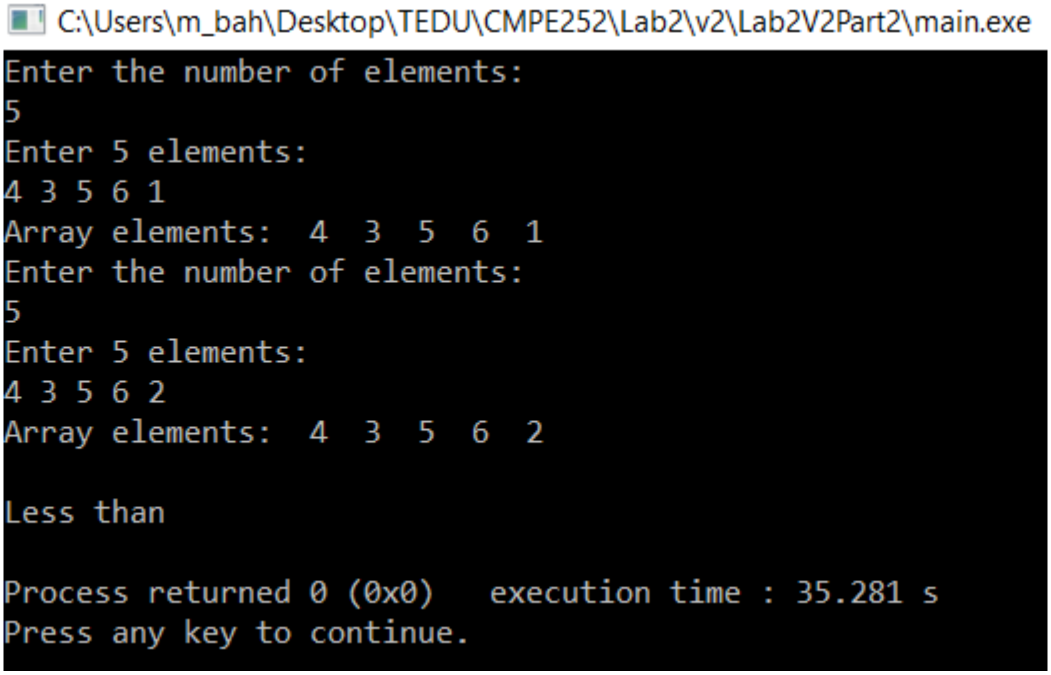


C:\Users\m_bah\Desktop\TEDU\CMPE252\Lab2\v2\Lab2V2Part2\main.exe

```
Enter the number of elements:
5
Enter 5 elements:
1 2 3 4 5
Array elements:  1   2   3   4   5
Enter the number of elements:
5
Enter 5 elements:
1 2 3 4 5
Array elements:  1   2   3   4   5

Equal

Process returned 0 (0x0)   execution time : 24.411 s
Press any key to continue.
```



C:\Users\m_bah\Desktop\TEDU\CMPE252\Lab2\v2\Lab2V2Part2\main.exe

```
Enter the number of elements:
5
Enter 5 elements:
4 3 5 6 1
Array elements:  4  3  5  6  1
Enter the number of elements:
5
Enter 5 elements:
4 3 5 6 2
Array elements:  4  3  5  6  2

Less than

Process returned 0 (0x0)   execution time : 35.281 s
Press any key to continue.
```

Part 3 (35 points)

Implement the following function in skeleton code `lab1part3.c`:

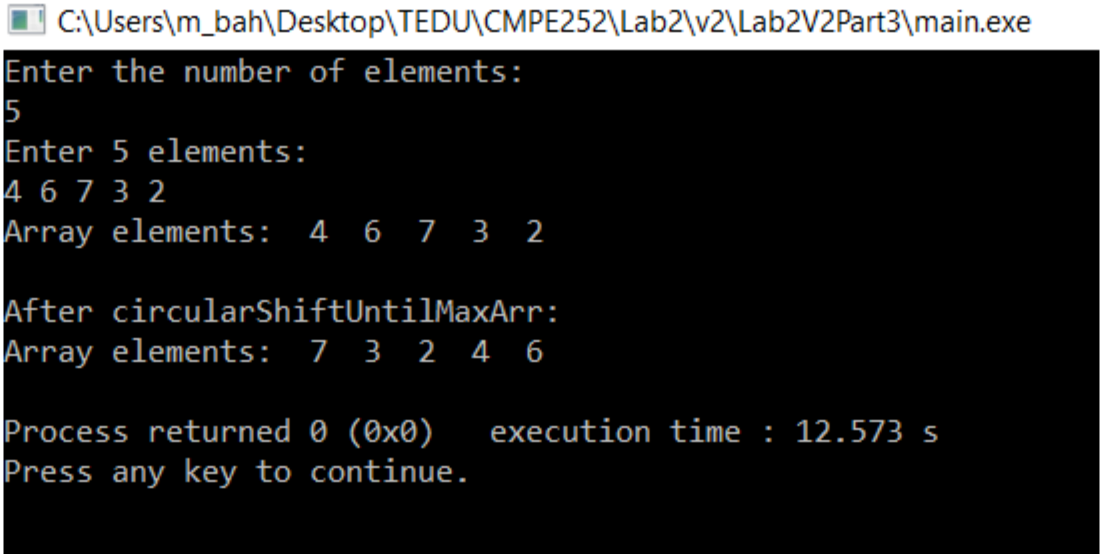
```
// Circularly shift elements of arr from left to right until sequence of values in
// arr becomes the largest considering all sequence of values obtained by circularly
// shifting elements in arr.
void circularShiftUntilMaxArr(int arr[], int n);
```

- You need to use `circularShiftFromLeftToRight` function and `compareTwoArrays` function while implementing `circularShiftUntilMaxArr` function.

Assume that arr can have at most 500 elements in it. **Hint:** If you need to declare a local array in function `circularShiftUntilMaxArr`, you can set its size as 500.

Your task in this part to fill in the missing function definitions in skeleton code `lab1part3.c`. `main` function will stay as it is.

Sample Run:



C:\Users\m_bah\Desktop\TEDU\CMPE252\Lab2\v2\Lab2V2Part3\main.exe

```
Enter the number of elements:
5
Enter 5 elements:
4 6 7 3 2
Array elements:  4  6  7  3  2

After circularShiftUntilMaxArr:
Array elements:  7  3  2  4  6

Process returned 0 (0x0)   execution time : 12.573 s
Press any key to continue.
```