Programming and Data Structures Laboratory | 2021-22 Autumn semester, Section 20
Assignment 4 | January 04, 2022

Submission instructions

* Submit one .c file for each part of the assignment. Each of your .c files should be named as:

<your roll number>_A<assignment number>_<part number>.c

For instance, if your roll number is 21CS10023, and if you are presently doing Assignment 4 which has 4 parts, then you should submit 4 separate .c files named as:
21CS10023_A4_1.c   21CS10023_A4_2.c   21CS10023_A4_3.c   21CS10023_A4_4.c

* Submissions must be through the course Moodle, before the end of Lab session (11:55 AM). Late submissions will be penalized / not accepted.

---

1. [10 Marks] Write a program to separate odd and even integers in separate arrays. Assume that the array can hold maximum 10 integers.
input: No. of elements in the array, the different integer values in the array
output: One array for odd integers and one array for even integers
Example output:

```
Enter the number of elements: 5
Enter the elements separated by space: 4 8 3 4 6
Even Integers: 4 8 4 6
Odd Integers: 3
```

2. [10 Marks] Write a program that will insert a new value to an already existing sorted array such that the sorting is maintained. You can assume
1. The original array is sorted in ascending order
2. That the user provides the elements of the original array in already sorted order
3. That the array can hold maximum 10 values
input: No. of elements in the array, the different values in the array in sorted order, the new value to be inserted
output: The original array, the modified array
Example output:

```
Enter the num of elements (Max Size =9): 5
Enter the elements in sorted order (separated by space): 5 10 15 20
25
Enter the new value to be inserted: 17
Old array: 5 10 15 20 25
Modified array: 5 10 15 17 20 25

Enter the num of elements (Max Size =9): 5
Enter the elements in sorted order (separated by space): 5 10 15 20
25
Enter the new value to be inserted: 30
Old array: 5 10 15 20 25
Modified array: 5 10 15 20 25 30
```

3. [15 Marks] Fill a single dimensional integer array of size 30 with random integral numbers in the range **5** to **50** by calling the **rand()** library function. Please visit the following link [http://www.cplusplus.com/reference/cstdlib/rand/] to know how to use **rand()** function to generate random integers between a lower and upper bound. It should then display the array

contents nicely formatted. Then reverse the elements of the integer array, so that the last element becomes the first, the second from last becomes the second, and so on. It should reverse the elements in place – that is, without using another array. It should display the contents of the array after reversal.

Example output:

```
The original array is: 22 18 32 35 17 13 49 23 30 22 41 40 11 13 42
8 28 24 5 31 30 34 12 38 29 10 16 11 35 14
The reversed array is: 14 35 11 16 10 29 38 12 34 30 31 5 24 28 8 42
13 11 40 41 22 30 23 49 13 17 35 32 18 22
```

4. [15 Marks] Write a program that takes as input an integer array a[10] that stores n (= at most 10) elements (scanned during execution) and finds how many elements are out of order. An element a[i] is said to be in order if it is not smaller than a[0], a[1], ..., a[i-1] and not larger than a[i+1], a[i+2], ..., a[n-1].

Example output:

```
Enter number of elements: 2
Enter 2 integers (separated by space): 5 7
Not in order = 0

Enter number of elements: 4
Enter 4 integers (separated by space): 5 6 7 9
Not in order = 0

Enter number of elements: 4
Enter 4 integers (separated by space): 5 5 7 5
Not in order = 2

Enter number of elements: 7
Enter 7 integers (separated by space): 2 4 2 2 4 2 2
Not in order = 6
```