Indian Institute of Information Technology Sri City, Chittoor

(An Institute of National Importance under An Act of Parliament)

Name: **DSA Lab - 7** Duration: **3 Hrs**

Date: **27th May, 2021** Maximum Marks: **10**

INSTRUCTIONS:

1. Please carefully read all assignment problems and complete the function prototypes given to solve the problems. Do not change the function prototypes.
2. Write only a single main function. You can call the required functions from the main function with static input or input provided by the user. Do not ask for any user input within the functions.
3. Name the file as follows: S2020xxxxx_A7.c
4. DO NOT zip. Upload a single .c file directly to your submission in the common Google classroom.

If you do not follow the above-mentioned instructions, a suitable penalty would be imposed.

ASSIGNMENT PROBLEMS

1. A structure stud_record has the following definition

```
struct stud_record
{
    int roll_no;
    char name[20];
    float marks[3];
};
```

Assume that there exists an array of structures of type stud_record which has been sorted in decreasing order of roll_no. Complete the function below in the most efficient manner, which takes as argument a pointer to the array of structures, the number of student records present in the array and the roll number of the student to be searched. If a student record with the searched roll number exists, then the function returns a pointer to the record. Otherwise the function returns NULL.

*struct stud_record * record_search(struct stud_record * record_array[], int n, int roll){ }* **[2 marks]**

2. Consider a character array which contains only 0s and 1s. Write a function that returns 1 if all the 1s within the string occur together in one undivided block, and returns 0 otherwise. You can just complete the function below. **[3 marks]**

```
int check_string(char arr[])
{
}
```

Example:

Input	Output
001111100	1
1111000000000	1
00000011	1
01	1
01110110	0
01010101	0
00000000	0 [if the string consists of only 0s, the function returns 0]

3. Binary search algorithm is designed to search sorted arrays efficiently. Consider an array such as: 3 4 5 1 2. This array is not sorted, but it is not completely unsorted as well. The given array is actually a sorted array (1 2 3 4 5) that has been rotated cyclically three positions to the right. Modify the binary search algorithm to search for an element within this array. You just have to complete the function given below:

```
int modified_binary_search(int  arr[], int n, int key, int k)
{                                                    }
```

In the given function prototype, *arr[]* is the rotated array, *n* is the number of elements in the array, *key* is the value to be searched and *k (k<n)* is the number of positions by which the array has been shifted cyclically to the right. The function returns the index of *key* if it exists within the array, and otherwise it returns -1. Keep in mind that the complexity of your function should still be *O(log n)*. **[5 marks]**

PRACTICE PROBLEM (UNGRADED):

1. Consider the third question once again. Imagine that you are not given any information about *k*, the number of times the array has been rotated. In this case, how will your function need to be changed? Can you still complete the task at hand in O(log n)?