CS559: Computer Systems Lab

READ THE FOLLOWING INSTRUCTIONS CAREFULLY. FAILING TO ADHERE TO IT WILL LEAD TO DEDUCTION OF MARKS.

1. Make sure to give the correct information in your submission link. You will receive a response email for every submission. **You must save it for future reference for the whole semester as you will be asked to show it.**

2. Every assignment will have multiple questions. **For every question, you will save your solution (i.e. program) as a .c file.** So if "Assignment 01" has 4 questions (marked as 1, 2, 3 and 4), then you will save your solution program as 1.c, 2.c, 3.c and 4.c.

   Note: Your program will be compiled to check if your solution is correct and you will be marked accordingly. If any solution file is missing or not submitted, the marks for that solution will be zero.

3. Submit your files in a compressed folder (rar or zip) as per "submission instructions" given in your assignment question paper.

   Note: You must make sure to upload in .zip or .rar. **Folders submitted with any other extension or wrong extension will not be evaluated and awarded zero.**

4. The student must ensure that the submitted file is not corrupted and can be unzipped properly. **Corrupted files that cannot be opened will be given zero. Similarly bad filenames having any other extension than ".c" will not be evaluated and given zero.**

5. You will be able to upload your assignment multiple times within the deadline span, but only your last submission will be treated as final and considered for evaluation. **No plea/request to consider intermediate submissions for evaluation will be entertained.** So, make sure to carefully check that you are submitting the correct assignment.

6. The submission link will be automatically deactivated after the deadline, and no request/plea for extension will be accepted. **Non-submission /wrong submission will be automatically awarded zero.**

7. Your code will be checked for similarity and you will be penalized according to the following rule:

   Similarity above 75%: 50% deduction
   Similarity of 100%: full deduction

8. **Compiler Information: Program must be compiled using online gdb compiler. If your program can not be compiled using gdb compiler then marks will be reduced.**
   Compiler Link : `https://www.onlinegdb.com/online_c_compiler`

9. Marks for every assignment (other than practice assignments) will be mailed to you with remarks. **You will be allowed a deadline for rebuttal, after which no rebuttal will be entertained.**

Submission Instruction: Store your assignments in folder and compress it as a rar/zip file (filename should be in this format: roll-number_assign1.rar or roll-number_assign1.zip). For example, if your roll number is 2211CS01, store your assignment as 2211CS01_assign1.rar or 2211CS01_assign1.zip. Also, save each program in the format given beside each question. Upload the same at the below link:
`https://forms.gle/3mWFR41qvXrzwRiJ7`

Problem 1 (Save as 1.c) :- Write a c program to read an input from the text file. Find the gcd of the content of the text file. Finally append the pascals triangle for the gcd found. **[10 Points]**

Step 1: content of the text file - two integers (name the text file as input.txt)

i.e, input.txt
81
153
Read the content of the file, find the gcd.

Step 2: now the gcd is the length of pascal triangle. Append this to the same text file.(gcd = 9)

I.e, input.txt
81
153
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1

Problem 2 (Save as 2.c) :- Considering array representation and index of the array starting from 1, construct a binary search tree from n number of elements $a_1, a_2, a_3, ..., a_n$. Consider k as a key. Starting from the root node, find the path and sum of the path according to the following rules: **[10 Points]**

   a) If abs($a_i - k$) is an even number, add node a_i to the sum and traverse left sub-tree.

   b) If abs($a_i - k$) is an odd number, add node a_i to the sum and traverse right sub-tree.

   c) If a node has no subtree, add node a_i and return.

Note: Here, abs(x) is the absolute value of x.

First-line contains the number of test cases T. For each test case, the first line asks for the number of elements. Next line asks for n number of elements. Consider that elements are inserted in the same order in a Binary search tree (using search and insert operation). Next line asks for the value of k.

Sample Input:
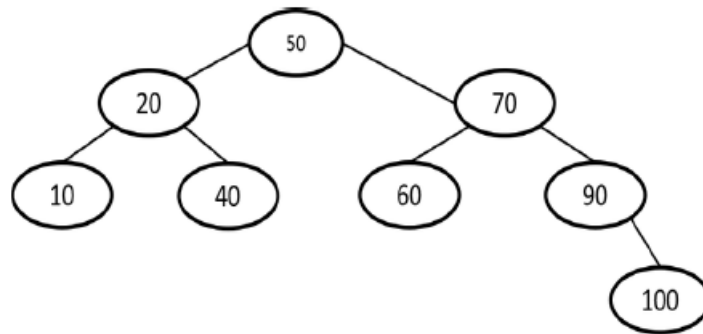Number of test cases: 1
Number of elements: 8
50 70 60 20 90 10 40 100
k=15

Expected Output:
Sum of the path=310
Path= 50 → 70 → 90 → 100

Explanation: If elements are inserted in the above order, we get the following binary search tree.

According to the above rules, since each node has an even number, their absolute difference with k is odd. Hence we keep traversing right node till we encounter a leaf node. At each node, we keep recording the path and adding the value of the node to the sum. Hence we get the final output as 310.

Problem 3 (Save as 3.c) :- There are 'n' number of elements which are to be inserted into a stack in sorted order. Find the total number of pop operations required to maintain the stack if elements are arranged in (from bottom of the stack to the top): **[10 Points]**

a) Ascending order

b) Descending order

Constraints:
Range of the elements: [1...N], where N=10000
Every element is an integer
$n <= 50$

First line of the program should take the number of test cases. For each test case, the first line should ask the number of elements to be inserted. Second line asks for n inputs and the third line asks whether input should be inserted in ascending/descending order.

Sample input:
1
6
45 32 100 79 88 11
Order (1 for ascending/2 for descending): 1

Expected output:
Number of pops: 8

Explanation:

No pop

45					

1 pop

32	45	100			

1 pop

32	45	79	100		

1 pop

32	45	79	88	100	

5 pop

11	32	45	79	88	100

Problem 1:

- Improper indentation: -1 points

- Compilation error: -2 points

- Input not taken(i.e hard-coded): -3 points

- Wrong logic/output: -4 points

Problem 2:

- Improper indentation: -1 points

- Compilation error: -2 points

- Input not taken(i.e hard-coded): -3 points

- Wrong logic/output: -4 points

Problem 3:

- Improper indentation: -1 points

- Compilation error: -2 points

- Input not taken(i.e hard-coded): -3 points

- Wrong logic/output: -4 points