CS 211 Data Structures and Algorithms Lab
Autumn 2022

Assignment no.	**1**
Objective	To implement Stacks using array and use it to solve the Tower of Hanoi/ Tower of Brahma/ Lucas' Tower
Total marks	10
Due date (without penalty)	22 August (Monday) 11:59 pm
Cut-off date (with penalty - 5%)	25 August (Thursday) 11:59 pm
Penalty for violating naming convention(s)	5%

There are two tasks in this assignment - *Task#1 and Task#2.* The solution for Task#1 must be used to solve Task#2 and the evaluation will be based on the output of Task#2.

Task#1:

The objective of this task is to implement Stacks using arrays. You need to implement PUSH and POP operations. There should be checks for Stack Overflow and Stack Underflow. The maximum size of stacks must be n, which will be given as a command-line argument.

Task#2:

*The objective of this task is to solve the Tower of Hanoi/ Tower of Brahma/ Lucas' Tower (ToH). In TOH, there are three poles A, B, and C. There are n disks each of them has a different diameter. Initially A contains all these n disks in the increasing order of diameter (**the largest disk is at the bottom of the pole**).*

Objective:

To move all the *n disks* to the *pole C* by following the below rules:

(i) only one disk can be moved at a time;

(ii) each move consists of taking the upper disk from one of the poles and placing it on top of another pole or on an empty pole;

(iii) a disk cannot be placed on top of a disk with less diameter.

While solving this problem, you have to use three stacks A, B, C in place of three poles and each move consists of popping one element from one stack and pushing that element to another stack. Initially all disks are to be pushed to the stack A.

Command-line argument:

Your program should receive one command-line argument: ***the number n of disks*.**
You can assume that the k^{th} disk has a diameter *k* units.

Output:

The output of your program should be in a file named '**toh.txt**'. Each line should denote a stack operation. The first n lines must be "Push disk <k> to Stack A", where k takes values n to 1 (in decreasing order). This is required to make sure that *all n disks are in Stack A* before solving the problem. Then the remaining stack operations must be written in order to achieve the goal, i.e., to place all disks in Stack C in the same order in which they present in Stack A initially. If the stack operation is Push the disk 3 to Stack B, then the corresponding line must be "Push disk 3 to Stack B". If the stack operation is Pop the disk 4 from Stack B, then the corresponding line must be "Pop disk 4 from Stack B".

In short, every line must have six words *where each two adjacent words are separated by a single white space*.

Word 1: "Push" or "Pop"
Word 2: "disk"
Word 3: <the-disk-number>
Word 4: "to" (if Word 1 is "Push") or "from" (if Word 1 is "Pop")
Word 5: "Stack"
Word 6: "A" or "B" or "C"

Submission:

- The program you submit should output: ***toh.txt*** when we run the program for evaluation.
- The main file of your program should be named as <roll no>.<extension>, where roll no. specifies your roll no. and the extension depends on the language you choose (in our case ".c" is a mandatory for this assignment). Ex: 210010001.c.
- Do the stress test of your program well before submission.
  (i) You may use the attached sample input files for testing; the corresponding output files are also attached;
  (ii) We have some hidden inputs with us to test your program. *The mark you obtain is purely based on whether your program correctly gives outputs for the hidden inputs*.
- If your program has only a single source file, please submit the file as it is. If your program has multiple source files, please submit your code as a zip file where the name of the zip file should be your roll number. It **is important that you follow the input/output conventions exactly** (including the naming scheme) as we may be doing an automated evaluation. ***There will be a penalty of 5% (on the mark you deserve otherwise) if you do not follow the naming conventions exactly.***
- Follow some coding style uniformly. Provide proper comments in your code.
- *Submit only through moodle*. ***Submit well in advance***. Any hiccups in the moodle at the last minute is never acceptable as an excuse for late submission. Submissions through email or any other means will be ignored.

- Acknowledge the people (other than the instructor and TA) who helped you to solve this assignment. The details of the help you received and the names of the people who helped you (including internet sources, if applicable) should come in the beginning of the main file as a comment. ***Copying others' programs and allowing others to copy your program are serious offences and a deserving penalty will be imposed if found.***

Evaluation:

- To consider for the first evaluation without penalty, you have to submit your program by the due date. If you submit after the ***due date*** but on or before the ***cut-off date***, ***there will be a penalty of 5% on the marks you deserve otherwise***.
- If you do not submit by the ***cut-off date,*** your program will not be considered for the first evaluation.
- We will do the first evaluation after the cut-off date. *The marks you obtain will be proportional to the number of correctly placed disks in Stack C* (starting from stack bottom) after the execution of the program. *Please note that the marks will be 0 if you do not write the first n lines that push n disks to Stack A*. For evaluation, we will simulate the output produced by your program. So, if there is any error being encountered while doing the evaluation (for example, your algorithm says to Pop from an empty Stack) then we will stop the simulation. Also, if a line is formatted inappropriately, then also we will stop the simulation.
- After the first evaluation, you will get a chance to improve your program. *For this, after modification, you can submit your code for* ***second evaluation****.* ***It comes with a 20% penalty.*** The due date for the second evaluation will be announced after the first evaluation. Those who submit their code after the cut-off date and before the due date for second evaluation will also be considered for the second evaluation. Submissions done after the due date of the second evaluation will be ignored.