Online Assignment 2: Stack (07/12/2021 Tuesday-Morning)

In your offline component you have already implemented Arr (ArrStack) and LL implementation of Stack ADT as specified in the offline specification document. Now it has been found that there are some bugs in your Arr implementation (don't be offended; this is just a story!). In particular, the dynamic allocation is not working properly. So, you now have to create another stack class (OuterStack), which have the same methods and constructors as in ArrStack but no actual storage for stack of its own. And, you will have to handle possible overflow in OuterStack. Therefore, OuterStack will use ArrStack for the actual stack implementation, but, it must assume that the overflow handling (i.e., dynamic allocation) of ArrStack is faulty. So, OuterStack will have to implement all the methods of ArrStack principally by calling the corresponding methods of ArrStack. But, it will somehow ensure the following characteristic:

A. It must identify when there is an overflow.
B. When it identifies an overflow it creates a new (i+1th) stack of the same size and considers it as an extension of the old (ith) stack. So in this way there could be a series of stacks and you need to carefully manage the stack related operations across this series of stacks.
C. You can have private helper functions as well as private member variables to help you manage this; but no storage can be allocated as actual stack storage; for that you need to instantiate ArrStack objects.

When you write the main function to demonstrate the functionality of the OuterStack, you cannot use ArrStack. You can only instantiate OuterStack and use that.

Submission Guidelines:

1. **Create a directory with your 7-digit student id as its name**
2. **You need to create one file for the OuterStack implementation code (e.g. outer.cpp/outer.py). You will also include your previous ArrStack implementations as a separate file. Create a separate file for the main function to demonstrate the OuterStack implementation. You cannot use any DS apart from your ArrStack implementation.**

3. **Put all the source files only into the directory created in step 1. Also create a readme.txt file briefly explaining the main purpose of the source files.**
4. **Zip the directory (compress in .zip format. Any other format like .rar, .7z etc. is not acceptable)**
5. **Upload the .zip file on Moodle in the designated assignment submission link. For example, if your student id is 1905xxx, create a directory named 1905xxx. Put only your source files (.c, .cpp, .java, .h, etc.) into 1905xxx. Compress the directory 1905xxx into 1905xxx.zip and upload the 1905xxx.zip on Moodle.**

Failure to follow the above-mentioned submission guideline may result in upto 10% penalty.