ITU Computer Engineering Department
BLG 223E Data Structures, Summer 2021
Homework #1
Due July 15, 2021 23:59

Definition

You are asked to implement a class that encapsulates an integer array that can grow and shrink its capacity as elements are being added to/removed from the structure. Please pay attention to the following details inyour implementations:

- Your structure should be coded as a class
- Your class should contain a constructor that initializes the wrapped array with a capacity of its argument
- Your class should contain method(s) that may add a single element or a bunch of elements at once from your structure. Duplicates are allowed.
- Your class should contain method(s) that may remove a single element or a bunch of elements at once from your structure. In case of duplicates, always the first occurence of an element should be removed. Nothing to be done if the element to be removed is not contained.
- The array in your structure should increase its capacity once the number of elements reaches the array capacity.
- The amount of increase in the capacity is determined to be one of the following:
    - Double the capacity when needed
    - Increase the capacity by log2 of its current capacity
    - Increase the capacity by the amount of the initial capacity
- The array in your structure should decrease its capacity once the number of elements reaches the appropriate capacity fort he next shrink
- The amount of decrease in the capacity is determined to be one of the following:
    - Half of the capacity when needed (For instance when the current capacity is 40 and the number of elements in the structure is 20, the capacity should be shrinked to 20)
    - Decrease the capacity by log2 of its current capacity
    - Decrease the capacity by the amount of the initial capacity
- Your class should contain a method to query if a number is contained in your structure. If the answer is yes, the indice of the first occurence should be returned. Otherwise -1 should be returned.
- Your class should contain a method to query its current capacity
- Your class should contain a method to query its current size (number of elements in the structure)
- Your class should contain a method to query the grow/shrink strategy. This method should output one of the following strings accordingly: TWICE, LOG, CAP_BY_CAP
- Your class should contain a method to print all the numbers in the wrapped array in a single line seperated by spaces

Input-Output

Your program should accept a filename as a command line parameter. In the accepted file, each line will contain a distinct command and parameters that should be processed by your program. In the input, the following command may be given:

- The first line is always a single number indicating the initial capacity of the structure.
- SS <strategy>: Sets the grow/shrink stretegy to one of the following: TWICE, LOG, CAP_BY_CAP.
- ADD <numbers separated with spaces>: Add all the numbers in the line to your structure
- REMOVE<numbers separated with spaces>: Remove all the numbers in the line to your structure.
- GC: Get current capacity
- GS: Get current size (number of elements)
- PRINT: Print the contents of your structure
- CONTAINS <number>: Print the indice of the first occurence of the single <number>, if return -1
- CLEAR: Set the structure to its initial state, with no elements in it.

Only the GC, GS, PRINT and REMOVE commands should produce output. The output they need to produce is explained in the previous definitions.

The input file would be error-free syntactically. It might contain semantic errors (e.g. try to remove element from an empty structure)

Deliver

Please zip and deliver the directory structure defined below:

- HW1 : Topmost folder, that will contain all the folders in your submission. No other files should be present under this folder in your submission.
- HW1/src: Contains all the *.cpp files
- HW1/src/main.cpp: Contains you main function.
- HW1/src/Expandingarray.cpp: Contains your array wrapper.
- HW1/include: Contains all the header files you use
- HW1/bin: An empty directory that will contain objective files when your project is compiled

Please check the calico test file in the homework definition to see how your files will be compiled and tested.

Restrictions and Guidelines

- Compilation environment: Only the code that can be compiled by the environment of the container definition provided in ninova will be accepted.
- Testing of your program will be performed using Calico (https://calico.readthedocs.io/en/latest/). Test cases that will be used to test your homework is provided as an attachment in ninova.
- **STL usage is forbidden.**
- **This homework is for individual submissions**. Any kind of code sharing or code adaptation from an external source is strictly forbidden. Submitted code will undergo a plagiarism check

process and plagiarsim penalties may be given if the submitted code's similarity is approved by the instuctor.

- Make sure you write your name and number in all of the files of your project, in the following format:
  /* @Author
  Student Name:<studentname>
  Student ID :<studentid>
  Date:<date>*/
- Only electronic submissions through Ninova will be accepted no later than deadline
- Use comments wherever necessary in your code to explain what you did.