C++ Advanced – Exam 2 (21 Apr 2019)

Write C++ code for solving the tasks on the following pages.

Code should compile under the C++11 standard.

Submit your solutions here: https://judge.softuni.bg/Contests/1608/CPlusPlus-Advanced-Retake-21-Apr-2019

Any code files that are part of the task are provided under the folder **Skeleton**.

Please follow the exact instructions on uploading the solutions for each task.

Task 3 – Memory Nightmare

You are given 3 files: main.cpp, Defines.h and MemoryContainer.hpp.

You are given the **`main()`** function, which reads a single integer value of memory (N).

- The next N lines are special command strings;
  - "push containerType containerSize" command – creates a new container of type 'containerSize' with the containerSize as container size;
    where contaierType is:

    ```
    enum ContainerType
    {
        SHORT_CONTAINER       = 0, //container should have 'short' C++ primitive data type
        INT_CONTAINER         = 1, //container should have 'int' C++ primitive data type
        LONG_LONG_CONTAINER   = 2 //container should have 'long long' C++ primitive data type
    };
    ```

    And containerSize has a 'size_t' value in range **[1-SIZE_T_MAX]** inclusive;
    'size_t' and 'unsigned long long int' are the same thing.

  - "pop" command – removes the **last** added container (**if there is any**);

Your task is to study the provided Skeleton and implement the missing functionalities for ContainerInterface.hpp and a different .cpp file, which implement the methods **pushContainer()** and **popContainer()** defined in main.cpp (For example ContainerUtils.cpp).

Important note: Your implemented methods should not introduce memory leaks – otherwise some of your test cases will fail.

Keep in mind that the Judge system has a 64bit Little-endian architecture so:

- sizeof(short) is 2 bytes;
- sizeof(int) is 4 bytes;
- sizeof(long long) is 8 bytes;

At the end of each call to **pushContainer()** and **popContainer()** you should print "occupiedMemory: " followed by how many bytes of **dynamically allocated memory** requested by 'push' command is currently occupied by your program.

Example:

4 commands with output for each command:

- push 0 2 – “occupied memory 4” (2 short’s, 2 bytes each)
- push 1 2 – “occupied memory 12” (2 int’s, 4 bytes each)
- push 2 2 - “occupied memory 28” (2 long long’s, 8 bytes each)
- pop – “occupied memory 12” (the 2 long long’s were removed)

Your task is to study the code and implement the function so that the code accomplishes the task described.

You should submit a single **`.zip`** file for this task, containing **ONLY** the files you created.

The Judge system has a copy of the other files and will compile them, along with your file, in the same directory.

Restrictions

Your program has limit for 16MB of memory.

Your program should NOT introduce memory leaks – otherwise some of your test cases will fail.

Examples

Input	Output
5 pop pop push 0 20 push 1 20 push 2 20	occupiedMemory: 0 occupiedMemory: 0 occupiedMemory: 40 occupiedMemory: 120 occupiedMemory: 280
3 push 2 50 pop push 2 20	occupiedMemory: 400 occupiedMemory: 0 occupiedMemory: 160
4 push 0 1 push 1 2 push 2 3 pop	occupiedMemory: 2 occupiedMemory: 10 occupiedMemory: 34 occupiedMemory: 10