

Яндекс.Практикум, Алгоритмы: Спринт 1.

Финальные задачи

🕒 7 авг 2020, 02:02:32
старт: 5 авг 2020, 01:56:18

С. Стековая Очередь

Ограничение времени	1 секунда
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Рита вчера по дороге на работу задумалась: можно ли реализовать очередь с использованием стеков?

На самом деле можно, и вам предстоит это сделать.

Очередь должна поддерживать методы:

put - добавляет элемент

get - извлекает самый ранее добавленный элемент

get_size - возвращает текущий размер очереди

Формат ввода

В первой строке записано n - количество команд, оно не превосходит 5000. В каждой из следующих n строк записана одна из команд:

put value

get

get_size

value - целое число, по модулю не превосходящее 1000.

Формат вывода

Выведите результат вызова методов. Если метод get вызывается для пустой очереди, нужно напечатать 'error'.

Пример 1

Ввод	Вывод
5	-9
put -9	2
get	
put -3	
put 2	
get_size	

Пример 2

Ввод	Вывод
------	-------

Ввод	Вывод
8	-7
put -7	1
get	-2
put 1	1
get	
put -2	
get	
put 2	
get_size	

Пример 3

Ввод	Вывод
9	4
put -3	5
put 0	-3
put 7	4
put 4	
get_size	
put -4	
get_size	
get	
get_size	

Примечания

Основные требования к выполнению задания:


Должен быть соблюден принцип FIFO.

В реализации очереди используются только стеки.

Готовый стек из стандартной библиотеки языка программирования использовать нельзя.

Постарайтесь, чтобы получившаяся очередь была эффективной.

Количество стеков, которое вы можете задействовать в решении, не ограничено!

Язык GNU c++17 7.3 

Набрать здесь

Отправить файл

```

1 #include <iostream>
2 #include <string>
3 #include <sstream>
4
5 using namespace std;
6
7 /**
8  * @brief структура элемента связного списка
9  */
10 struct Node
11 {
12     Node( int value ) : mValue( value ), mNext( nullptr ) {}
13     Node( int value, Node* next ) : mValue( value ), mNext( next ) {}
14
15     int mValue; // значение элемента
16     Node* mNext; // указатель на следующий элемент
17 };
18
19 /**
20  * @brief структура стека
21  */
22 struct Stack
23 {
24     Stack() : mTop( nullptr ), mSize( 0 ) {}
25
26     /**
27      * @brief определить, пустой ли стек
28      * @return true / false
29      */
30     bool Empty()
31     {
32         return mSize == 0 ? true : false;
33     }
34
35     /**
36      * @brief положить в стек новый элемент
37      * @param value значение элемента
38      */

```

Отправить

Предыдущая