**Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики**

**Алгоритмы и структуры данных**

**Лабораторная работа №4**

**Стек, очередь, связанный список**

Выполнил:

Бараканов Жаргал Мырзабекович

Факультет ИКТ

Группа K3121

Преподаватель:

Харьковская Татьяна Александровна

**Санкт-Петербург**

**09.12.2021**

**Задание 1.**

Реализуйте работу стека. Для каждой операции изъятия элемента выведите ее результат.

На вход программе подаются строки, содержащие команды. Каждая строка содержит одну команду. Команда — это либо “+ N”, либо “–”. Команда “+ N” означает добавление в стек числа N, по модулю не превышающего 109. Команда “–” означает изъятие элемента из стека. Гарантируется, что не происходит извлечения из пустого стека. Гарантируется, что размер стека в процессе выполнения команд не превысит 106 элементов;

* Формат входного файла (input.txt). В первой строке входного файла содержится M (1 ≤ M ≤ 106) – число команд. Каждая последующая строка исходного файла содержит ровно одну команду;
* Формат выходного файла (output.txt). Выведите числа, которые удаляются из стека с помощью команды “–”, по одному в каждой строке. Числа нужно выводить в том порядке, в котором они были извлечены из стека. Гарантируется, что изъятий из пустого стека не производится;
* Ограничение по времени. 2 сек.;
* Ограничение по памяти. 256 мб.

*Решение:*

Для выполнения задания был создан класс Stack. Исходным объектом является список. Реализованы функции добавления элемента в конец и удаления их оттуда, а также проверки стека на пустоту.

Изображение выглядит как текст

Автоматически созданное описание

В основном коде создаём number\_list – объект класса Stack. Далее построчно считываем команды из файла input.txt. Если на вход подана команда «+», добавляем элемент в стек. В ином случае удаляем элемент, попутно записывая его в файл output.txt.

Изображение выглядит как текст

Автоматически созданное описание

**Задание 3.**

Последовательность A, состоящую из символов из множества «(», «)», «[» и «]», назовем правильной скобочной последовательностью, если выполняется одно из следующих утверждений:

* A – пустая последовательность;
* первый символ последовательности A – это «(», и в этой последовательности существует такой символ «)», что последовательность можно представить как A = (B)C, где B и C – правильные скобочные последовательности;
* первый символ последовательности A – это «[», и в этой последовательности существует такой символ «]», что последовательность можно представить как A = (B)C, где B и C – правильные скобочные последовательности.

Так, например, последовательности «(())» и «()[]» являются правильными скобочными последовательностями, а последовательности «[)» и «((» таковыми не являются. Входной файл содержит несколько строк, каждая из которых содержит последовательность символов «(», «)», «[» и «]». Для каждой из этих строк выясните, является ли она правильной скобочной последовательностью.

* Формат входного файла (input.txt). Первая строка входного файла содержит число N (1 ≤ N ≤ 500) – число скобочных последовательностей, которые необходимо проверить. Каждая из следующих N строк содержит скобочную последовательность длиной от 1 до 104 включительно. В каждой из последовательностей присутствуют только скобки указанных выше видов;
* Формат выходного файла (output.txt). Для каждой строки входного файла (кроме первой, в которой записано число таких строк) выведите в выходной файл «YES», если соответствующая последовательность является правильной скобочной последовательностью, или «NO», если не является;
* Ограничение по времени. 2 сек.;
* Ограничение по памяти. 256 мб.

*Решение:*

Для выполнения задания был создан класс Stack, как и в задании 1.

Изображение выглядит как текст

Автоматически созданное описание

В основном коде считываем скобочную последовательность и работает с ней посимвольно. Если встречена открывающая скобка, то она кладется в стек. Если встречена закрывающая скобка, то из стека вынимается последняя вложенная скобка. Если тип скобок не совпадает или в стеке не осталось элементов, то программа выдаст ответ «NO». Также, если после выполнения программы в стеке остались элементы, то программа также выдаст ответ «NO». В остальных случаях выведется «YES».

Изображение выглядит как текст

Автоматически созданное описание

**Задание 6.**

Реализуйте работу очереди. В дополнение к стандартным операциям очереди, необходимо также отвечать на запрос о минимальном элементе из тех, которые сейчас находится в очереди. Для каждой операции запроса минимального элемента выведите ее результат.

На вход программе подаются строки, содержащие команды. Каждая строка содержит одну команду. Команда – это либо «+ N», либо «–», либо «?». Команда «+ N» означает добавление в очередь числа N, по модулю не превышающего 109. Команда «–» означает изъятие элемента из очереди. Команда «?» означает запрос на поиск минимального элемента в очереди.

* Формат входного файла (input.txt). В первой строке содержится M (1 ≤ M ≤ 106) – число команд. В последующих строках содержатся команды, по одной в каждой строке;
* Формат выходного файла (output.txt). Для каждой операции поиска минимума в очереди выведите её результат. Результаты должны быть выведены в том порядке, в котором эти операции встречаются во входном файле. Гарантируется, что операций извлечения или поиска минимума для пустой очереди не производится;
* Ограничение по времени. 2 сек.;
* Ограничение по памяти. 256 мб.

*Решение:*

Для выполнения задания используется очередь, построенная двух стеках. В первый стек происходит только добавление элементов, из второго – только удаление. Если второй стек пуст, то в него перемещаются все элементы первого стека, таким образом в конце второго стека оказывается начало очереди. В стеках хранятся пары элементов: сам элемент и минимальный элемент на текущий момент. Для того, чтобы найти минимум очереди, достаточно сравнить минимумы двух стеков, которыми являются элементы стеков с индексами -1,1.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

**Задание 7.**

Задан массив из n целых чисел - a1, ..., an и число m < n, нужно найти максимум среди последовательности ("окна") {ai , ..., ai+m−1} для каждого значения 1 ≤ i ≤ n − m + 1. Простой алгоритм решения этой задачи за O(nm) сканирует каждое "окно" отдельно. Ваша цель - алгоритм за O(n).

* Формат входного файла (input.txt). В первой строке содержится целое число n (1 ≤ n ≤ 105) – количество чисел в исходном массиве, вторая строка содержит n целых чисел a1, ..., an этого массива, разделенных пробелом (0 ≤ ai ≤ 105). В третьей строке - целое число m - ширина "окна"(1 ≤ m ≤ n);
* Формат выходного файла (output.txt). Нужно вывести max ai, ..., ai+m−1 для каждого 1 ≤ i ≤ n − m + 1;
* Ограничение по времени. 5 сек.;
* Ограничение по памяти. 512 мб.

*Решение:*

Для выполнения данного задания используется очередь на двух стеках, описанная в задании 6. В основном коде начала считываются и добавляются в очередь m элементов. Далее добавляем по элементу, находим минимум и удаляем первый элемент, пока не дойдем до конца входной последовательности.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание