

Разбор задач

Задача 1. Игра роботов

- Для решения данной задачи будем перебирать сколько идентификаторов назовут роботы в порядке слева направо.
- Будем решать эту задачу в 0-индексации. Пусть текущий робот назовёт i идентификаторов, тогда если $k - i > 0$ выполним $k = k - i$ и перейдем к следующему роботу, в противном случае, выведем $a[k-1]$, где a — это массив с идентификаторами роботов

Задача 2. А и В и командная тренировка

Будем образовывать команды жадно:

- Пока мы можем образовать хотя бы одну любую команду:
 - Выбираем команду – каких больше участников тех и берем в 2-ом количестве, а с других в одном. И одновременно считать (+1 к переменной)

Задача 3. Команды cd и pwd

- Текущую директорию будем хранить в стеке. В начале он пуст- мы в корневой директории
 - Если строка начинается с «/», то это означает абсолютный путь, то есть вся предыдущая директория перезаписывается, следовательно стек надо очищать.
 - А остальную строку обрабатывать в цикле
 - Если встречается «..», то выходим с последней директории – удаляем вершину стека
 - Иначе эта новая директория и мы переходим – добавляем в стек
- При команде «pwd» выводим содержимое стека, после каждого элемента «/»

Задача 4. Скобочная последовательность

- Для каждой открывающейся скобки попытаемся определить ей соответствующую закрывающуюся. Более формально, пусть открывающаяся скобка находится на позиции i , тогда скобка на позиции j называется ей соответствующей, если подстрока $s_i \dots s_j$ является кратчайшей правильной скобочной последовательностью, начинающейся с индекса i .
- Если s не является правильной скобочной последовательностью, то не для каждой скобки найдется ей соответствующая.

Задача 4. Скобочная последовательность

- Будем идти по строке s и складывать позиции, на которых находятся открывающиеся скобки, в стек.
- Пусть мы находимся на позиции i , если s_i — открывающаяся скобка, то просто положим ее номер на вершину стека. Если нет, то посмотрим на вершину стека:
 - если стек пустой или последняя открывающаяся скобка не соответствует текущей, то очистим стек.
 - В противном случае запомним, что для скобки, лежащей на вершине, соответствующая есть скобка на позиции j и снимем вершину со стека.

Задача 4. Скобочная последовательность

Мы можем склеивать подряд идущие блоки в правильные скобочные последовательности.

При этом **выгодно** склеить как можно больше блоков, чтобы набрать как можно больше скобок нужного типа. Склеим все подряд идущие блоки, получим несколько подстрок, являющихся правильными скобочными последовательностями. Выберем из получившихся подстрок ту, в которой наибольшее количество скобок «[».

Задача 5. Поход в кино

- Воспользуемся «Dictionary<int, int> cnt» и насчитаем, сколько учёных говорит на каждом языке (то есть $cnt[i]$ должно быть равно количеству учёных, которые говорят на языке номер i).
- Заведем переменные $max=0$, $max1=0$ – для нахождения фильма с максимальным количеством «очень довольных» человек и «довольных».
- Переберем все фильмы, начиная с первого. Пусть текущий фильм имеет номер i . Мы для него можем, быстро посчитать, сколько будет очень довольных и довольных(по cnt).
- Тогда, сравниваем количество очень довольных с max , если он больше, то обновим ответ номером текущего фильма.
- Если очень довольных людей одинаковое количество, то сравниваем довольных и обновляем $max1$, если новое значение больше $max1$

Задача 6. Волшебный порошок -1

- Будем печь по одной печенье до тех пор пока это возможно.
- Для каждой новой печенки насчитаем val — сколько нужно волшебного порошка для её приготовления.
- Для этого переберём все ингредиенты, и для ингредиента номер i :
 - если $a[i] \leq b[i]$ выполним присвоение $b[i] = b[i] - a[i]$
 - в противном случае, выполним присвоение $b[i] = 0$ и $val = val + a[i] - b[i]$.
- После того, как мы перебрали все ингредиенты, если $val > k$, то больше печенек испечь мы не сможем. В противном случае, выполним присвоение $k = k - val$ и перейдем к приготовлению следующей печенки.

Задача 7. Психи - в шеренгу!

Задача 8. Редактор правильных скобочных последовательностей

- Будем решать данную задачу следующим образом. Сначала с помощью *stack* насчитаем массив *pos*, где *pos[i]* будет означать позицию скобки, парной для скобки в позиции *i*. Затем заведём два массива *left* и *right*. Тогда *left[i]* будет равно позиции ближайшей слева относительно позиции *i* неудалённой скобки, а *right[i]* будет равно позиции ближайшей справа относительно позиции *i* неудалённой скобки. Если таковых скобок нет, будет хранить в соответствующей позиции в массиве число «-1».

Задача 8. Редактор правильных скобочных последовательностей

- Пусть текущая позиция курсора равна p . Тогда при операции «L» выполним присвоение $p = left[p]$, а при операции «R» выполним присвоение $p = right[p]$. Осталось научиться обрабатывать операцию «D».
- Пусть lf равно p , а rg равно $pos[p]$. Если $lf > rg$ сделаем $swap(lf, rg)$. То есть теперь мы знаем границы подстроки, которую нужно удалить. Пересчитаем сначала позицию p . Если $right[rg] = -1$ (то есть после удаления текущей подстроки не останется скобок справа), нужно сдвинуть p влево, то есть выполнить присвоение $p = left[lf]$, иначе нужно выполнить присвоение $p = right[rg]$. Осталось только пересчитать ссылки для концов удаляемой подстроки. Здесь нужно быть аккуратным, и проверять есть ли скобки слева и справа относительно концов удаляемой подстроки.

Задача 8. Редактор правильных скобочных последовательностей

Для вывода ответа нужно определить номер первой слева неудалённой скобки, с помощью массива *right* пройти по всем неудалённым скобкам и вывести их в ответ. Для определения номера первой неудалённой скобки можно сложить все пары концов удаляемых подстрок в массив, затем отсортировать его и, проитерировавшись по полученному массиву, определить искомую позицию.