Задача A. Ремонт от BB-8

Автор задачи: Григорий Шовкопляс
Автор разбора: Григорий Шовкопляс

Дан прямоугольник $h \times w$, который нужно замостить минимальным количеством прямоугольников $a \times b$, при этом разрешается, чтобы прямоугольники накладывались и выходили за пределы наружного прямоугольника.

Решение:

Будем решать задачу по высоте и ширине отдельно. Очевидно, что чтобы покрыть отрезок длины h отрезками длины a понадобится минимум $\lceil \frac{h}{a} \rceil$ отрезков. Аналогично для ширины. Осталось только перемножить.

Таким образом ответ на задачу: $\lceil \frac{h}{a} \rceil \times \lceil \frac{w}{b} \rceil$.
Примечание: У некоторых участников был Presentation error на тесте «1 1 1000 1000» из-за вещественного типа данных, потому что в языке C++ по умолчанию $1\,000\,000$ *типа double выводится как «1e+006».*

Задача B. Дроиды и ангар

Автор задачи: Илья Пересадин
Автор разбора: Илья Пересадин

Дана полоска, на которой пустые ячейки, каменные блоки и дроиды. Дроиды движутся по заданному алгоритму, одновременно перемещаясь. Нужно выяснить, какие дроиды не врежутся в стену после выполнения алгоритма.

Решение:

Определим для каждого дроида независимо, врежется ли он в стену в результате выполнения алгоритма. Посчитаем, на сколько ячеек максимально влево и вправо должны будут ходить дроиды во время выполнения алгоритма. Это можно посчитать, прогнав алгоритм на фиктивном дроиде, на которого нет ограничений в движении.

Также предподсчитаем для каждого дроида ближайшие слева и справа стены. Затем проверим, что каждый дроид, сместившись максимально влево, не врежется в ближайшую левую для него стену. Аналогично проверим для правой стены.

Подсчет максимальных сдвигов влево и вправо выполняется за $O(m)$, подсчет ближайших стен выполняется за $O(n)$, проверка каждого дроида происходит за $O(1)$, всего дроидов n штук. Итого, сложность решения $O(n + m)$.

Задача C. Игра в перерыве

Автор задачи: Дмитрий Филиппов
Автор разбора: Дмитрий Филиппов

Дан массив чисел, разрешается делать две операции:

- Взять любое четное число и заменить его на два числа, в два раза меньших выбранного;
- Два одинаковых числа заменить на одно, равное их сумме

Требовалось сказать, какое максимальное число можно получить такими действиями.

Решение:

Будем разбивать каждое число на два, пока можно — хуже от этого не будет, потом можно все собрать обратно. Теперь у нас есть большой массив, но хранить его полностью необязательно — в нем много чисел повторяются, поэтому будет хранить пары (x, cnt), где x — число, а cnt — сколько раз оно встречается. Например, если изначально у нас был массив $[2, 2, 4, 6, 12]$, то хранить мы будем массив $[(1, 8), (3, 5)]$. Делать это можно например с помощью ассоциативных массивов, которые в языке C++ реализованы в контейнере std::map, а в Java — HashMap.

Теперь осталось заметить, что две различных пары нельзя соединять друг с другом, потому что числа в них не равны, а следовательно производить бо́льшие числа можно только внутри одной пары. Несложно заметить, что максимальное число, которое можно получить из пары (x, cnt) равно $x \cdot 2^k$, где k — максимальное число, удовлетворяющее неравенству $2^k \le cnt$.

Осталось по всем парам взять максимум, это и будет ответом на задачу.

Задача D. Джедайские запросы

Автор задачи: Илья Пересадин
Автор разбора: Илья Пересадин

Вам даны n чисел a_i и m запросов q_j. Запрос представляет из себя следующее: это строка, состоящая из цифр и знаков вопроса. Ответ на запрос получается следующим образом:

- Подставим вместо знаков вопросов всевозможные комбинации цифр, получим числа-подзапросы.
- Для каждого числа-подзапроса найдем, сколько чисел из заданных n больше либо равны, чем этот подзапрос.
- Просуммируем все ответы на подзапросы. Данная сумма и будет являться ответом на запрос.

Решение:

Дополним все запросы ведущими нулями до длины 9. Добавим все запросы в бор, так, чтобы в листах бора оказались последние цифра запроса. То есть каждому запросу соответствует лист в вершине бора.

Теперь для каждого числа a_i посчитаем, в каких запросах он учтется. Будем делать это следующим образом. Дополним a_i ведущими нулями до длины 9. Спускаемся по бору, переходя по цифрам.

Пусть сейчас переход по цифре d, тогда во все запросы, в которые переход идет по цифрам от 0 до $d-1$ текущее число a_i учтется 10^k раз, где k — количество знаков вопроса, от текущей вершины до запроса-листа. Прибавим на префиксе в этой вершине от 0 до $d-1$.

Если в запросе на текущей позиции стоит знак вопроса, то a_i учтется в нем d раз (знак вопроса заменяется на каждую цифру от 0 до $d-1$).

Теперь случай, когда текущая цифра запроса равна d, перейдем в эту вершину. Еще один случай равенства цифр, когда на текущем месте в запросе знак вопроса: он заменится на d, перейдем в эту вершину. Итого, из каждой вершины мы переходим максимум в две вершины. Поддерживаем вершины на текущем уровне, в которые мы перешли, и из каждой из них переходим по очередной цифре и знаку вопроса, при этом прибавляя на префиксе 0 до $d-1$. Итого, обработка каждого a_i происходит за 2^9.

После этого нужно спуститься по бору и просуммировать ответы для каждого листа. Эта операция происходит за $10 \times 10 \times m$.

Итоговая сложность: $2^9 \times n + 10 \times 10 \times m$