

## Задача А. Игральные кубики

Подзадачи 1-2. Для проверки условия правильности будем сравнивать сумму каких-нибудь двух чисел набора с двумя суммами пар остальных чисел. Если все три суммы окажутся равными, значит, набор правильный. Таким образом, для реализации этого подхода необходимо устроить перебор сумм из всевозможных пар чисел данного набора. Для небольшого числа наборов такое решение будет проходить первую группу тестов.

Подзадача 3. Проверку правильности набора можно сделать более эффективно. Заметим, что наибольшее и наименьшее числа в наборе обязательно находятся на противоположных гранях правильного кубика. То же будет верно для следующих по величине чисел набора, и так далее. Поэтому сначала отсортируем (например, по возрастанию) текущий набор из 6 чисел, затем подсчитаем суммы первого и шестого, второго и пятого, третьего и четвёртого чисел полученного набора. Если все три суммы совпадают, набор правильный.

## Задача В. Лунная арифметика

Автор задачи — Киндер М.И.

Подзадача 1. Если исходные числа  $a$  и  $b$  не превосходят 9, то их сумма и произведение вычисляются непосредственно из условия задачи:  $a + b = \max\{a, b\}$  и  $a \times b = \min\{a, b\}$ .

Подзадача 2. Если числа  $a$  и  $b$  — оба двузначные, то их сумма и произведение вычисляются отдельно в каждом разряде цифр. Если же одно из чисел однозначное, а другое двузначное, то при сложении получается двузначное число, у которого первая слева цифра совпадает с первой цифрой двузначного числа, а вторая равна наибольшей среди цифр этого разряда. Также легко вычисляется и произведение  $a$  и  $b$ .

Подзадача 3. Как и в подзадаче 2, сначала определяем, какое из чисел  $a$  и  $b$  имеет больше цифр. Для вычисления суммы  $a + b$  выделяем первые (слева) цифры большего числа, которые в итоге будут совпадать с цифрами суммы. Оставшиеся цифры складываем поразрядно по правилу наибольшей цифры. Произведение  $a$  и  $b$  вычисляется «столбиком» по правилу лунного умножения цифр с последующим лунным сложением всех промежуточных результатов.

Приведём примерный код процедур сложения и умножения в лунной арифметике на языке Python:

```
def Lunar_Add(a,b):
    s = min(len(a),len(b))
    t = ''
    if len(a) > s:
        t = a[:len(a)-s]
        a = a[len(a)-s:]
    if len(b) > s:
        t = b[:len(b)-s]
        b = b[len(b)-s:]
    for i in range(s):
        t += str(max(a[i],b[i]))
    return t
```

```
def Lunar_Mult(a,b):
    if len(a) < len(b):
        a,b = b,a
    t = ''
    for i in range(len(a)):
        s = ''
        for j in range(len(b)):
            s += str(min(a[i],b[j]))
        s += (len(a)-i-1) * '0'
        t = Lunar_Add(s,t)
    return t
```

## Задача С. Рациональное дерево

Подзадача 1 [Полный перебор]. Сгенерируем массив строк, каждая содержит  $n$  символов L и R ( $1 \leq n \leq 20$ ). Каждую из них проверим, соответствует ли она заданной дроби  $a/b$ . Будем перебирать их до тех пор, пока не найдём нужную. Такое решение пройдёт лишь для небольших дробей  $a/b$ , у которых запись содержит не более 20 символов L и R.

Подзадачи 2-3. Движение по дереву можно установить бинарным поиском. Алгоритм поиска записи дроби  $a/b$  в виде строки символов  $L$  и  $R$  можно записать в виде следующей процедуры:

```
while (a != b)
{
    if (a < b)
        { cout << "L"; b = b - a; }
    else
        { cout << "R"; a = a - b; }
}
```

Например, если  $a/b = 5/7$ , то с использованием этого алгоритма последовательно получаем:

$a = 5$	5	3	1	1
$b = 7$	2	2	2	1

при этом путь вдоль дерева, ведущий в исходную дробь  $5/7$ , будет записан в виде: LRRLL.

## Задача D. Охрана крепости

Давайте сначала переформулируем условие «охраняемой вершины» более понятным образом:

- Существует по крайней мере один охранник  $i$ , такой что расстояние между вершиной  $v$  и вершиной  $p_i$  (позиция охранника) не превышает  $h_i$  (запас здоровья охранника).

Мы можем перефразировать это следующим образом:

1. Охранник может перемещаться из текущей вершины в соседнюю, затрачивая единицу здоровья.
2. Как только здоровье достигает 0, охранник больше не может двигаться.
3. Вершина считается «охраняемой», если хотя бы один охранник может её достичь.

Такая переформулировка облегчает поиск решения.

## Формализация

Пусть  $d_i$  — максимальное здоровье охранника, который может достичь вершины  $i$  (или  $-1$ , если ни один охранник не может туда добраться). Если мы найдём все значения  $d_1, d_2, \dots, d_N$ , то задача будет решена.

На самом деле, значения  $d_1, d_2, \dots, d_N$  можно вычислить следующим образом:

## Алгоритм

1. Создайте массив  $x_1, x_2, \dots, x_N$ , который будет хранить предварительные значения  $d_i$ :
  - Если в вершине  $p_i$  находится охранник, то  $x_i$  инициализируется начальными значениями здоровья охранника.
  - Если охранника в  $p_i$  нет, то  $x_i = -1$ .
2. Повторяйте следующий процесс  $N$  раз, чтобы определить значения  $d_i$  по одному:
  - (a) Выберите вершину  $v$ , значение  $d_v$  которой ещё не определено, с максимальным  $x_v$ .
  - (b) Установите  $d_v = x_v$ , чтобы определить значение  $d_v$ .
  - (c) Для каждой вершины  $u$ , смежной с  $v$ , выполните следующее:
    - Если значение  $d_u$  ещё не определено, замените  $x_u$  на  $\max(x_u, x_v - 1)$ .

Этот процесс обоснован методом математической индукции (доказательство аналогично обоснованию алгоритма Дейкстры).

## Временная сложность

- Наивная реализация алгоритма потребует  $O(N^2)$  времени, так как поиск вершины с максимальным  $x_i$  занимает  $O(N)$ .
- С использованием структур данных, таких как куча или дерево отрезков, для управления вершинами с максимальным  $x_i$ , сложность можно сократить до  $O((N + M) \log(N + M))$ , что достаточно быстро для успешного решения задачи.
- Поскольку веса рёбер графа равны 1, сложность может быть дополнительно уменьшена до  $O(N + M)$ .

## Связь с задачами кратчайшего пути

Этот алгоритм эквивалентен решению задачи кратчайшего пути, если перевернуть знаки стоимостей. Таким образом, задачу можно свести к задаче кратчайшего пути, построив соответствующий граф. Алгоритм похож на алгоритм Дейкстры, но со знаками, изменёнными на противоположные.