Алгоритмы и структуры данных.

Динамическое программирование и потоки.

Часть первая.

(Динамическое программирование)

Задание №1.

Условие:

1. Кузнечик прыгает из клетки 1 в клетку n, длина прыжка может быть от 1 до k. Найти число различных путей. Время O(n).

Решение:

Для i ячейки находим значение следующим образом, мы храним сумму ячеек, которую высчитываем на каждой итерации следующим образом. Sum = sum – D[I – k] + D[I – 1]

Пример:

при k = 4;

База D[0] = 1;
переход D[i], если i > k: D[i] = sum + D[i - 1] - D[i - k]; sum = sum + D[i - 1] - D[i - k];
итог D[n];

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 4 | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 4 | 7 | 13 | 24 | 44 |

Код:

```
#include <iostream>
#include <vector>

int main()
{
    std::vector<int> f;
    f.push_back(1);
    for (int i = 0; i < 9; i++)
```

```
        f.push_back(0);

    int sum = 0;
    int k = 4;
    for (int i = 1; i < 10; ++i)
    {
        sum += f[i - 1];
        if (i - k >= 0)
        {
            sum -= f[i - k];
        }
        f[i] = sum;
    }

    for (int i = 0; i < 10; i++)
    {
        std::cout << f[i] << ' ';
    }

    return 0;
}
```

Задание №2.

Условие:

2. Кузнечик прыгает из клетки 1 в клетку n, длина прыжка может быть от 1 до k. У каждой клетки есть стоимость. Найти путь с минимальнойть стоимостью. Время O(n).

Решение:

Создадим массив минимальных стоимостей, в каждой ячейке которой будет лежать объект, такой что он будет состоять из текущей минимальной стоимости и указателя на следующий минимальный элемент после текущего. Когда мы будем отрезать текущий минимум от подмассива длины k мы будем переставлять указатель либо на следующий элемент после удаляемого либо на элемент который мы добавляем в подмассив длины k. Так же для каждого объекта мы будем хранить ссылку на объект из которого пришли и после того как мы получим ответ с наименьшей стоимостью мы сможем пройтись назад по ссылкам чтобы определить путь.

Задание №3.

Условие:

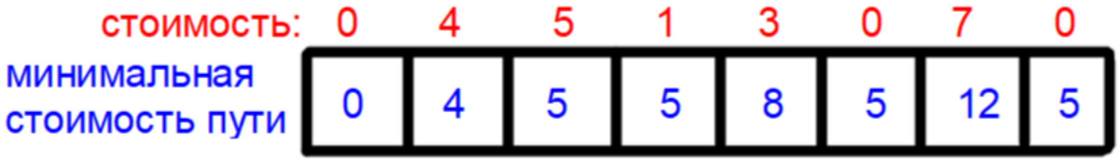
3. Кузнечик прыгает из клетки 1 в клетку n, каждый раз на 1 или 2 клетки вперед. У каждой клетки есть стоимость. Найти число различных путей минимальной стоимости. Время O(n).

Решение:

Идем из ячейки 1 до n при этом рассматриваем для каждой ячейки две предыдущие. То есть

D[i] = min(D[I – 1], D[I – 2]) + стоимость итой ячейки; при наличии одинаковых ячеек просто увеличиваем счетчик у итой ячейки на сумму этих двух ячеек иначе если они отличаются то берем максимум.

Пример:




Задание №4.

Условие:

4. Задана последовательность чисел. Требуется удалить из нее минимальное число элементов, чтобы она стала возрастающей. Время $O(n^2)$.

Решение:

В данном случае мы имеем классическую задачу на нахождение наибольшей возрастающей подпоследовательности, которая еще к тому же работает за квадрат, то есть мы перебираем все значения пока не найдем нужное. Числа которые нужно удалить это те которые не вошли в нашу подпоследовательность.

Задание №5.

Условие:

5. Задана последовательность чисел. Требуется найти число способов удалить из нее некоторые элементы, чтобы она стала возрастающей. Время $O(n^2)$.

Решение:

Для каждого итого элемента будем хранить число равное количеству комбинация для предыдущих чисел + 1 одна комбинация на итой позиции. Еще в конце из числа комбинаций вычитаем N

Пример:

Задание №6.

Условие:

6. У Васи есть калькулятор, который умеет выполнять три операции: прибавить 1, умножить на 2 и умножить на 3. Какое наименьшее число операций необходимо для того, чтобы получить из числа 1 число n.

Решение:

Мы идем с первого числа проверяя следующие условия: сначала к итому элементу прибавляем один и записываем в массив ходов количество ходов из итого минус один элемента + 1. Далее смотрим делится ли нацело число на два, если да то проверяем больше ли значение по индексу делить на два чем текущее если нет то мы записываем в итую ячейку ходов индекс итого делить на два + 1. И точно такая же ситуация с итым делить на три элементом.

Задание №7.

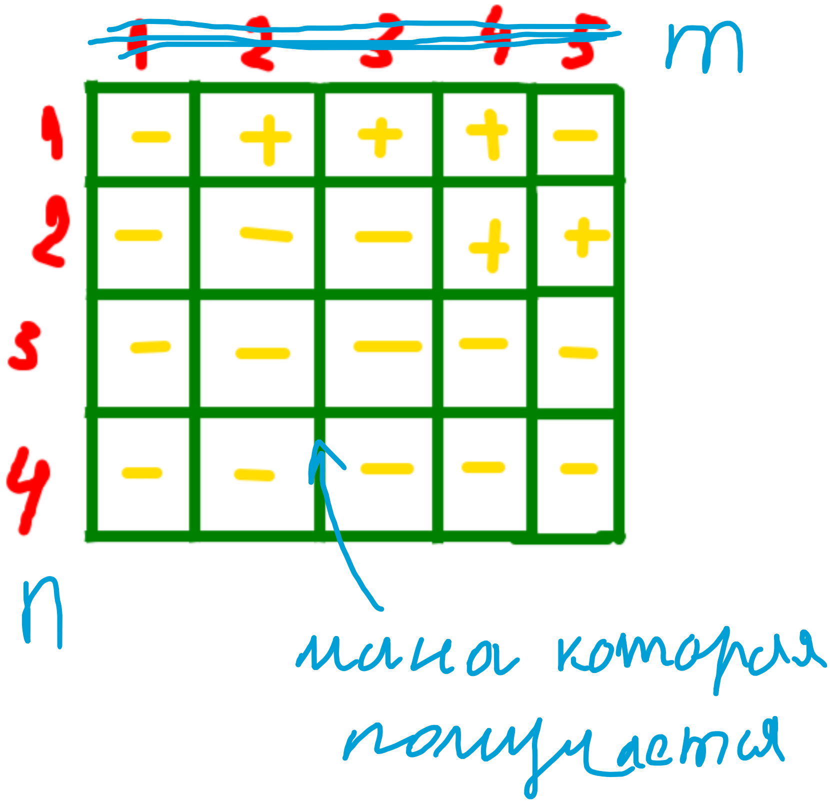
Условие:

7. У мага есть n заклинаний и m единиц маны. Заклининие i тратит $c_i$ маны и наносит врагу урон $d_i$. Убейте монстра с h здоровьем, потратив как можно меньше маны. Время $O(nm)$.

Решение:

Решение заключается в том, что мы просто составляем таблицу зависимости количества маны от числа заклинаний(как в задаче о рюкзаке). То есть наша задача в том чтобы определить когда мы достигли той точки когда сумма была минимальна.

~~m = 2, 3, 4, 6~~
~~n = 1, 2, 3, 4~~
~~h = 5~~



Задание №8.

Условие:

8. Петя и Вася выиграли на олимпиаде n призов, суммарной стоимостью r. Они хотят разделить их между собой так, чтобы разница в суммарной стоимости была минимальна. Время $O(nr)$.

Решение:

Найдем максимально возможное число и поделим все числа это число. Потом будем брать два максимальных числа и класть их в разные множества, а потом дополнять множества, которое меньше до равенства с другим множеством максимально возможным образом.

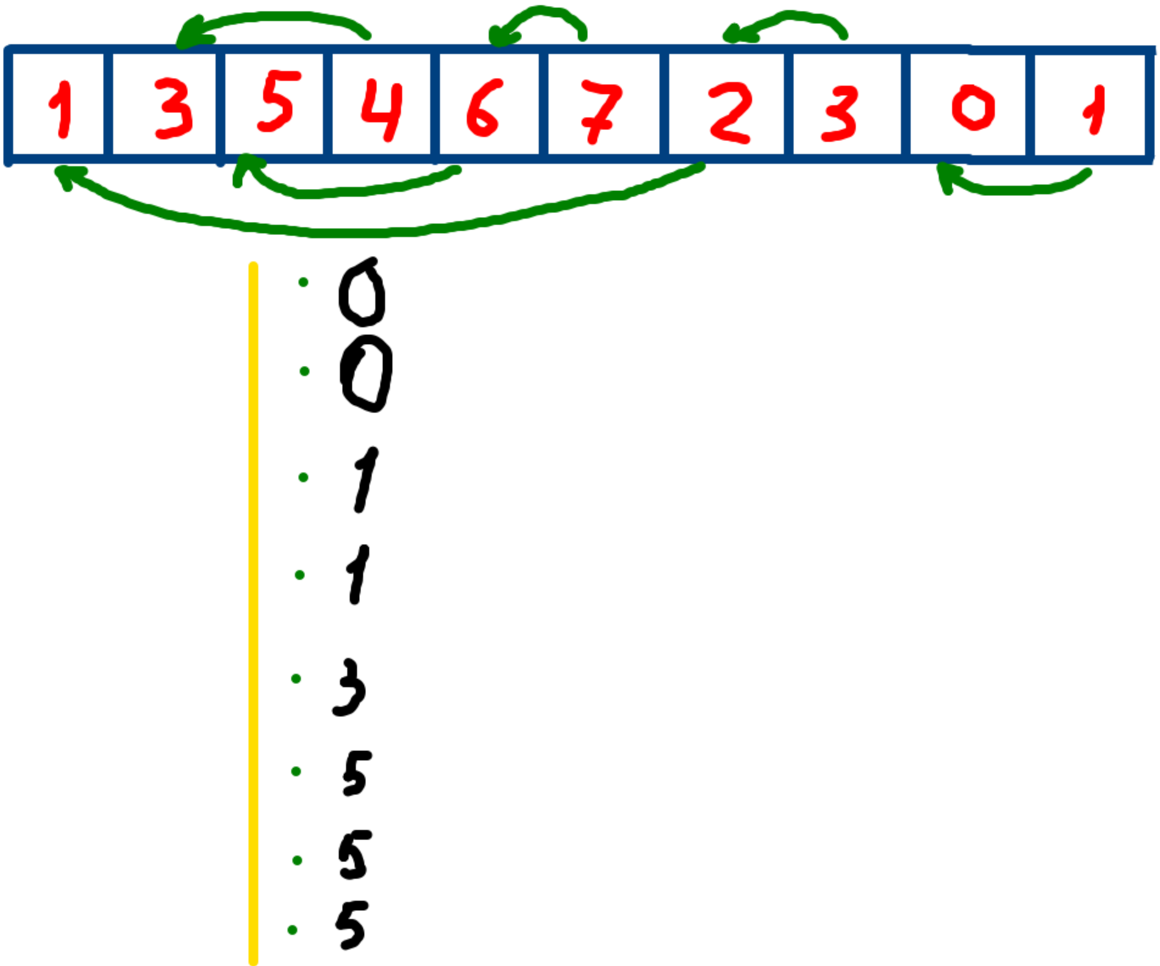
Задание №9.

Условие:

9. Дана последовательность, найдите максимальную по длине подпоследовательность из последовательных чисел (например, 5, 6, 7, 8). Время O(n).

Решение:

Смысл заключается что мы создаем связи и находим наибольшую последовательность.



Часть вторая.

(Потоки)

Задание №1.

Условие:

1. Докажите, что если в G существует поток величины x, то для любого $0 \le y < x$, существует поток величины y.

Решение:

Так как x это максимальный поток то по нему всегда можно пропустить поток y так как он меньше.

Задание №2.

Условие:

2. Докажите, что если все пропускные способности целочисленные, то существует максимальный поток, в котором поток по каждому ребру тоже целочисленный.

Решение:

Если все пропускные способности целочисленные, то и все максимальные потоки целочисленные, следовательно в максимальном потоке поток по каждому ребру тоже целочисленный.

Задание №3.

Условие:

3. Проверить, что в графе существует единственный минимальный разрез.

Решение:

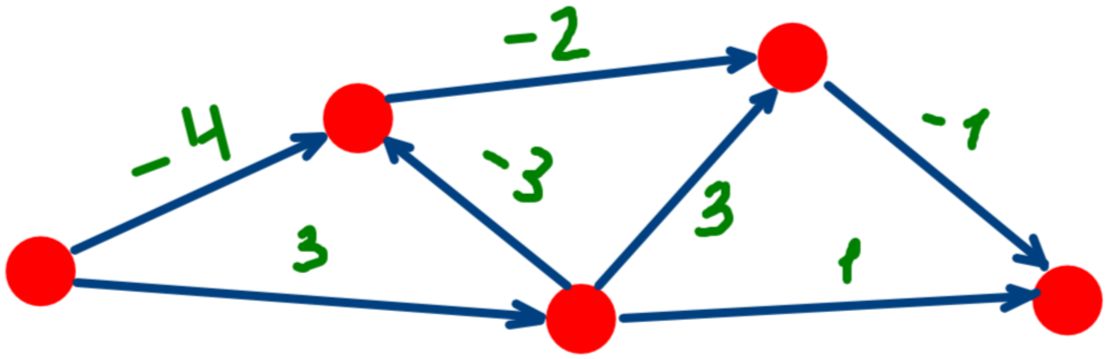
Для существования двух минимальных разрезов должно выполняться следующее условия ребра из этого разреза должны иметь такой же вес, как и ребра из другого разреза. Найдем первый разрез, и увеличим веса ребер в этом разрезе на единицу, после еще раз запустим поиск разреза. Если ребра из нового разреза будут сопоставимы с ребрами из старого разреза значит мы нашли второй минимальный разрез, следовательно в графе несколько минимальных разрезов, а это значит, что минимальный разрез в этом графе не единственный.

Задание №4.

Условие:

4. Приведите пример сети с вещественными пропускными способностями, в котором алгоритм Форда-Фалкерсона может никогда не завершиться.

Решение:

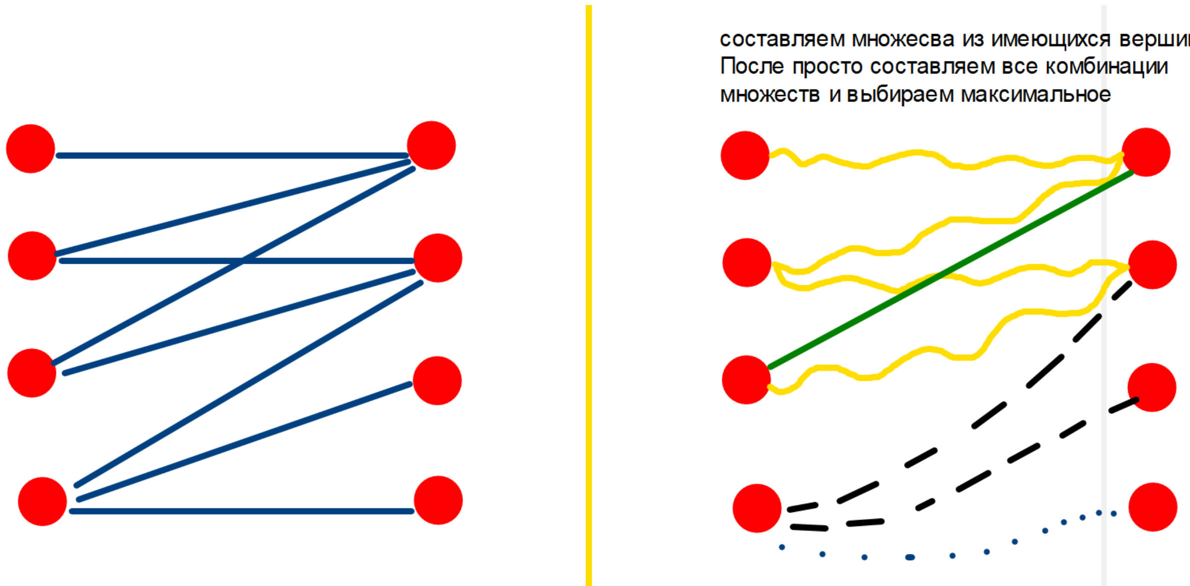



Задание №5.

Условие:

5. Задача о максимальном независимом множестве. Дан двудольный граф. Выберите максимальное число вершин, так чтобы не было ребра, соединяющего выбранные вершины.

Решение:




Задание №6.

Условие:

6. Задача о минимальном реберном покрытии. Дан двудольный граф. Выберите минимальное число ребер так, чтобы любая вершина была концом ребра из выбранного множества.

Решение:

Можно использовать для этого максимальное паросочетание. Так как при работе этого алгоритма как раз будет найдено минимальное число ребер соединяющее максимальное число вершин.

Задание №7.

Условие:

7. Дан двудольный граф и максимальное паросочетание в нем. Проверить, единственное ли это максимальное паросочетание, за $O(n + m)$.

Решение:

Предположение: Запустим дфс по максимальному паросочетанию так чтобы найти цикл. Цикл точно будет четным, так как граф двудольный. После нахождения цикла нам останется только поменять местами ребра.

Задание №8.

Условие:

8. Решите задачу о максимальном паросочетании в двудольном графе с помощью максимального потока. Как связаны разрез и минимальное вершинное покрытие?

Решение:

Минимальное вершинное покрытие связано с разрезом тем что любое ребро из разреза было инцидентно хотя бы одной вершине.