Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский Авиационный Институт»

Национальный Исследовательский Университет

Институт №8 «Информационные технологии и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №8

по курсу «Дискретный анализ»

Студент:	Хренникова А. С.
Группа:	М8О-208Б-19
Преподаватель:	Капралов Н. С.
Подпись:	
Оценка:	
Дата:	

Москва, 2021

Лабораторная работа №8

Задача:

Разработать жадный алгоритм решения задачи, определяемой своим вариантом. Доказать его корректность, оценить скорость и объём затрачиваемой оперативной памяти.

Реализовать программу на языке C или C++, соответсвующую построенному алгоритму. Формат входных и выходных данных описан в варианте задания.

Вариант 1:

На первой строке заданы два числа, N и p > 1, определяющие набор монет некоторой страны с номиналами p0, p1, ..., pN−1. Нужно определить наименьшее количество монет, которое можно использовать для того, чтобы разменять заданную на второй строчке сумму денег M ≤ 2^32 - 1 и распечатать для каждого i-го номинала на i-ой строчке количество участвующих в размене монет. Кроме того, нужно обосновать почему жадный выбор неприменим в общем случае (когда номиналы могут быть любыми) и предложить алгоритм, работающий при любых входных данных.

Формат входных данных:

На первой строке заданы два числа N и p > 1, определяющие набор монет некоторой страны с номиналами p0, p1, ..., pN−1. На второй сторке находится сумма денег которую необходимо разменять M ≤ 2^32 - 1.

Формат результата:

Для каждого i-го номинала на i-ой строчке вывести количество участвующих в размене монет.

Москва, 2021

1 Описание

Основная идея жадного алгоритма в том, чтобы на каждом этапе решения подзадачи выбирать локально оптимальное решение и в итоге получить оптимальное решение для всей задачи.

До тех пор, пока заданная сумма больше нуля, из нее вычитается максимально возможное число монет так: сначала это монеты наибольшего номинала, потом наминал меньше и т. д. То есть, если монеты определенного номинала можно вычесть, то вычитаем максимальное возможное количество, иначе переходим к следующему по убыванию номиналу.

Такое решение является оптимальным, так как номиналы представляют собой степени какого-либо числа, следовательно монету одного номинала можно заменить несколькими монетами большего номинала, если это не 1.

Если же номиналы не будут представлять собой степени какого-либо числа, тогда данный алгоритм не будет оптимальным. В таком случае необходимо воспользоваться динамическим программированием. Суть алгоритма в том, чтобы в массиве запоминать минимальное количество монет, которое нужно для размена от 1 до необходимой суммы монет.

Сложность используемого жадного алгоритма: линейная.

2 Исходный код:

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

int main()
{
    int N, p;
    unsigned int M;
    std::cin >> N >> p >> M;
    int k = N - 1;
    std::vector<int> p_arr(N);
    std::vector<int> res_arr(N);
    int current = 1;
    for (int i = 0; i < N; ++i) {
        p_arr[i] = current;
        current = current * p;
    }
    while (M > 0) {
        int count = M / p_arr[k];
        if (count == 0) {
            k--;
        }
        else {
            M = M - count * p_arr[k];
            res_arr[k] = count;
            k--;
        }
    }
    for (int i = 0; i < N; ++i) {
        std::cout << res_arr[i] << '\n';
    }
    return 0;
}
```

3 Консоль:

```
lina_tucha@LAPTOP-44CRFC1U:~/labs/da/lab8$ ./123
3 5 71
1
4
2
lina_tucha@LAPTOP-44CRFC1U:~/labs/da/lab8$ ./123
4 4 108
0
3
2
1
lina_tucha@LAPTOP-44CRFC1U:~/labs/da/lab8$ ./123
2 17 105
3
6
lina_tucha@LAPTOP-44CRFC1U:~/labs/da/lab8$ ./123
1 1 15
15
```

4 Тест производительности:

N = номиналы

p = 2

M = 5.000.000

Номиналы	Время(sec.)
100	0,000014
1000	0,000136
10000	0,001302
100000	0,01245
1000000	0,1388

Из графика видно, что временная сложность данного алгоритма линейная.

Москва, 2021

5 Выводы:

Выполняя данную лабораторную работу, я познакомилась с жадными алгоритмами. Основное отличие жадных алгоритмов от динамического программирования в том, что жадные алгоритмы не перебирают все возможные варианты решения подзадач, в поисках оптимального, а сразу берут наилучшее решение, которое заранее определено оптимальным.

Такой подход сокращает время работы программы и ее расход по памяти, но при этом применение жадного алгоритма должно учитываться еще на этапе планирования решения, так как подходит к более узкому множеству задач.

Москва, 2021