Дмитрий Баранов

Алгоритм

Даны города, расположенные на прямой, их координаты находятся в массиве *locations*. Нужно посчитать количество маршрутов из города *start* в город *finish* при начальном топливе *fuel*.

Из города можно поехать в любой другой город, потратив |*x*-*y*| (модуль разности) топлива. Города можно посещать множество раз. Каждый заезд в город *finish* считается отдельным маршрутом, даже если потом уехать и вернуться.

Решаем через динамическое программирование. Создаём двумерный массив *ways* размером *(fuel+1)* на *n*, где *n* — количество городов. Элемент *ways[f][v]* хранит количество способов оказаться в городе *v* с остатком топлива *f*. Изначально *ways[fuel][start] = 1* — мы начинаем в старте с полным баком, это уже один маршрут.

Перебираем все остатки топлива *f* от *fuel* до 0. Для каждого города *v* смотрим значение *ways[f][v]*. Если оно ноль — пропускаем (сюда не доехать). Иначе пробуем поехать во все другие города *u*.

Считаем стоимость поездки *cost* = |*locations[v] - locations[u]*|.

Если *f*>= *cost*, то обновляем *ways[f-cost][u] += ways[f][v]* по модулю 10^9+7 (1000000007).

После обработки всех состояний суммируем *ways[f][finish]* по всем *f* от 0 до *fuel*. Эта сумма и есть ответ — все возможные заезды в город *finish*.

Дмитрий Баранов

Временная сложность — асимптотика

Три вложенных цикла: по топливу (*fuel+1* итераций), по городам (*n*), и ещё раз по городам (*n*). Внутри — константные операции (*модуль, сравнение, сложение*). Итого O(*fuel* * n^2). При ограничениях n <= 100, fuel <= 200 получается не больше 2 млн операций — укладывается в время.

Затраты памяти — асимптотика

Массив *ways* размером *(fuel+1)* * *n*. При максимальных значениях —201 * 100 = 20100 целых чисел. Каждое по 4 байта — около 80 КБ. Остальные переменные занимают константное количество памяти. Итого O(*fuel* * *n*).