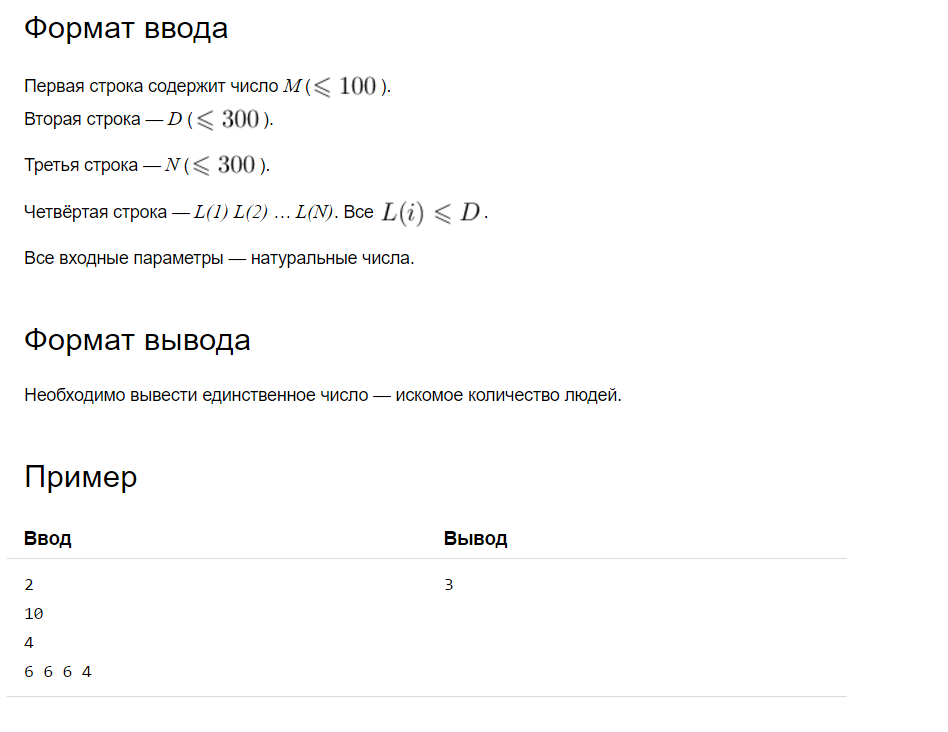
Задача 1.3 «Автобусы»

<https://github.com/olderor/advanced1>





Решение.

1. Создадим и считаем переменные – количество автобусов ( по условию),  – вместимость автобусов ( по условию) и – количество людей ( – по условию).

1. Создадим массив размером элементов. Тут будут храниться веса каждого пассажира ( по условию). Заполним массив значениями из условия.

2. Создадим массив размером элементов. Тут будут храниться максимальное количество пассажиров из диапазона от до , , которые могут поместиться в один автобус.

2.1. Для каждой строки посчитаем это максимальное количество. Для этого создадим набор , в котором будем хранить набор весов пассажиров, а также две переменные – сумма элементов набора, то есть сумма весов всех пассажиров, которых мы можем посадить в автобус, и – количество элементов в наборе, то есть количество пассажиров, которые мы можем посадить в автобус.

2.2 Переберем всех пассажиров с индексами от до . Далее рассмотрим следующие случаи:

2.2.1 В автобусе есть достаточно свободного места, и мы можем спокойно посадить пассажира  в автобус, то есть . Добавляем вес пассажира в набор , увеличиваем сумму на вес пассажира и инкрементируем переменную .

2.2.2 Если пассажир не влезает в автобус, необходимо взять максимальный вес пассажира, который находится в наборе . Если вес больше, чем вес пассажира , заменим этот вес на вес пассажира , а также изменим вместимость . Таким образом мы не меняем максимальное количество пассажиров, однако увеличиваем доступную вместимость автобуса, что дает нам больше свободы в будущем при добавлении новых пассажиров. Иначе следует пропустить пассажира, поскольку его добавление не делает вместимость оптимальным.

2.3 Запишем полученное максимальное количество в массив .

Исходя из этого, можно сделать вывод, что для хранения набора следует использовать структуру данных для того, чтобы мы могли быстро находить максимальное значение в этом наборе. А значит, что значение (максимальный вес пассажира в набое) мы можем получить используя за . Для того, чтобы заменить максимальный вес пассажира на вес пассажира следует достать максимальный вес с помощью , который работает за , где  – количество элементов в наборе (, и добавить новый вес , который работает за , где  – количество элементов в наборе (. Поскольку может принимать максимальное значение , то время работы изменения очереди в целом будет работать за .

3. Создадим массив размером . Заполним его с помощью динамического программирования, разбив задачу на подзадачи «максимальное количество пассажиров , которое можно поместить в количество автобусов , взяв пассажиров от до . Поскольку у нас уже имеются такие данные для одного автобуса, то база динамики – . Для заполнения будем брать максимальное , то есть ), где . То есть либо для пассажиров от до для количества автобусов уже есть оптимальным, либо мы можем взять больше пассажиров, если возьмем оптимальное количество пассажиров для пассажиров от до для количества автобусов , а также добавим новый автобус для остальных пассажиров – максимальное количество пассажиров на отрезке от до .

4. Ответ будет храниться в – максимальное количество пассажиров, которые можно посадить в автобусов, используя пассажиров.

Итоги

Оценка памяти. Для решения задачи потребовалось: массив размером ; массив размером ; очередь , максимальный размер которой мог достичь ; а также массив размером . Следовательно, полная сложность алгоритма по памяти равна .

Оценка времени работы. Для решения задачи потребовалось: считать данные в массив за ; заполнить массив с помощью двух вложенных циклов по и , за каждый, а также изменять очередь за , что в целом составляет ; а также заполнить массив с помощью трех вложенных циклов по, который работал за и , которые работали за каждый, что в целом составляет . Следовательно, полная сложность алгоритма по времени работы равна.

Итог:

* затраты по памяти
* затраты по времени