Замятин Евгений Игоревич,
Филиппов Дмитрий Сергеевич,
Ведерников Николай Викторович,
Ульянцев Владимир Игоревич

ЗАДАЧА «БУТЕРБРОДЫ ИЗ ЖУКОВ»

Этой статьей мы продолжаем цикл публикаций олимпиадных задач по информатике для школьников. Решение таких задач и изучение разборов поможет Вам повысить уровень практических навыков программирования и подготовиться к олимпиадам по информатике.

В этой статье рассматривается задача «Бутерброды из жуков», которая предлагалась на Второй командной интернет-олимпиаде по программированию в 2013–2014 учебном году. Материалы этой олимпиады можно найти на сайте http://neerc.ifmo.ru/school/io/.

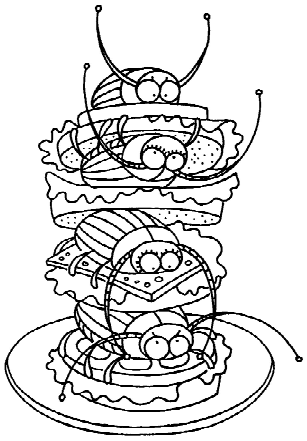
УСЛОВИЕ ЗАДАЧИ

Тимон и Пумба очень любят есть жуков, особенно бутерброды с ними. А их лучший друг Симба относится к этому лакомству равнодушно. Тимон и Пумба хотят доказать Симбе, что вкуснее бутербродов с жуками ничего нет. В дупле одного из деревьев они нашли n жуков и k кусков хлеба. Теперь они хотят сделать несколько бутербродов для Симбы.

Бутерброд делается следующим образом: кладется один кусок хлеба, сверху на него кладется один жук, на жука кладется еще один кусок хлеба и т. д. В итоге получится конструкция, в которой снизу лежит один кусок хлеба, дальше жуки и куски хлеба чередуются, причем наверху всей конструкции может лежать как жук, так и кусок хлеба. Для того чтобы бутерброд считался съедобным, он должен содержать хотя бы одного жука. Тимон и Пумба считают, что если Симба съест бутерброд, в котором будет t жуков, то уровень его удовольствия увеличится на a_t. Они хотят, чтобы Симба получил от их бутербродов как можно больше удовольствия, причем задействовать нужно всех жуков и все куски хлеба. Помогите им в этом нелегком деле!

Формат входного файла

В первой строке входного файла через пробел записаны два целых числа n и k, где n – количество жуков, k – количество кусков хлеба ($1 \le n \le 500$, $1 \le k \le 10^9$). Во



второй строке содержится n чисел a_i ($1 \le a_i \le 10^7$).

Формат выходного файла

В единственной строке выходного файла выведите максимальное суммарное удовольствие, которое Симба может получить, съев бутерброды. Если собрать бутерброды с использованием всех жуков и кусков хлеба невозможно, выведите «`Impossible`».

Примеры входных и выходных данных

sandwiches.in	sandwiches.out
3 3 1 3 2	4
3 2 1 2 3	Impossible

РАЗБОР ЗАДАЧИ

Для начала определим, когда ответ на задачу «`Impossible`». По условию задачи, в каждом бутерброде должен быть кусок хлеба и жук, значит, кусков хлеба должно быть не меньше, чем жуков. Кроме того, на одного жука максимум можно использовать два куска хлеба (в бутерброде из двух кусков хлеба, где посередине находится жук). Следовательно, если кусков хлеба больше, чем в два раза, чем жуков, то ответа не существует. То есть, если $k < n$ или $2n < k$, то следует вывести «`Impossible`» и завершить работу программы.

Для решения этой задачи будем использовать метод *динамического программирования* [1]. Суть этого метода – разбиение сложной задачи на более простые. Для решения задач этим методом необходимо определить три сущности: *состояние*, *переход* и *базу*. Метод динамического программирования основан на методе *математической индукции*.

Определим *состояние* как пару чисел (i, j), где i – количество уже съеденных жуков, j – количество съеденных бутербродов, наверху у которых был кусок хлеба. Чтобы решить задачу, посчитаем для каждого состояния значение $d_{i,j}$ – максимальное суммарное удовольствие, которое может получить Симба, съев сколько-то бутербродов так, что при этом суммарно было потрачено i жуков, и j из этих бутербродов «заканчивались» куском хлеба (то есть наверху лежал хлеб, а не жук).

Очевидно, Симба не может съесть больше бутербродов, чем количество имеющихся у него жуков (в каждом бутерброде должен быть хотя бы один жук). Следовательно, Симба не может съесть больше, чем n бутербродов. Эта оценка понадобится для определения времени работы программы.

Определим базу для дальнейших вычислений: $d_{0,0} = 0$. То есть, имея 0 жуков и 0 кусков хлеба, мы не можем ничего съесть, следовательно, не получим удовольствия вообще. Для удобства будем изначально считать, что для остальных значений $d_{i,j} = -\infty$. С точки зрения программы, возьмем достаточно большое число, такое, что любое суммарное удовольствие, доступное при любых входных данных, удовлетворяющих ограничениям, будет больше этого числа.

В листинге 1 приведена реализация инициализации массива d, в котором будут храниться все значения $d_{i,j}$.

Теперь определим *переход* динамического программирования. Пускай мы хотим узнать значение для состояния, в котором у нас съедено i жуков, и j съеденных бутербродов заканчиваются куском хлеба. Рассмотрим последний съеденный бутерброд. Пускай он состоял из w жуков и заканчивался хлебом. Тогда мы можем утверждать, что $d_{i,j} = d_{i-w,j-1} + a_w$. Или он состоял из w

Листинг 1. Инициализация массива d

```
for i := 0 to MAXN do begin
  for j := 0 to MAXN do begin
    d[i][j]:= -INF;
  end;
end;
d[0][0] := 0;
```

жуков и заканчивался жуком, тогда $d_{i,j} = d_{i-w,j} + a_w$. Таким образом, мы научились находить значение $d_{i,j}$ для заданного w через предыдущие, для которых ответ уже посчитан ранее. Нам осталось выбрать максимальное значение, перебрав допустимые значения w:

$$d_{i,j} = \max_{1 \le w \le i}(\max(d_{i-w,j}, d_{i-w,j-1}) + a_w).$$

Реализация подсчета значений $d_{i,j}$ на языке Pascal приведена в листинге 2.

Ответ на задачу содержится в ячейке массива `d[n][k - n]`. Осознание того, почему ответ содержится именно в этой ячейке, оставим читателю в качестве несложного упражнения. Итоговое время работы программы составляет $O(n^3)$, что удовлетворяет ограничениям на входные данные задачи.

Листинг 2. Подсчет значений d

```
for i := 0 to n do begin
  for j := 0 to i do begin
    for w := 1 to i do begin
      d[i][j] := max(d[i][j], d[i - w][j] + a[w]);
      if (j > 0) then begin
        d[i][j] := max(d[i][j], d[i - w][j - 1] + a[w]);
      end;
    end;
  end;
end;
```

Литература

1. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К.* Алгоритмы. Построение и анализ. М.: Вильямс, 2009.

Замятин Евгений Игоревич,
студент первого курса кафедры «Компьютерные технологии» НИУ ИТМО, член жюри Интернет-олимпиад по информатике,

Филиппов Дмитрий Сергеевич,
студент первого курса кафедры «Компьютерные технологии» НИУ ИТМО, член жюри Интернет-олимпиад по информатике,

Ведерников Николай Викторович,
студент четвертого курса кафедры «Компьютерные технологии» НИУ ИТМО, член жюри Интернет-олимпиад по информатике,

Ульянцев Владимир Игоревич,
аспирант кафедры «Компьютерные технологии» НИУ ИТМО, член жюри Интернет-олимпиад по информатике.