**Быстрое возведение в степень**

Рассмотрим задачу о возведении числа в целую неотрицательную степень. Напишем рекурсивную функцию, которая решает эту задачу:

**def** power(a, n):

**if** n == 0:

**return** 1

**else**:

**return** power(a, n - 1) \* a

Такая функция будет работать очень долго, если степень n является большим числом. Например, для n=100 такая функция сделает 100 вызовов самой себя. Чтобы вычислить степень числа быстрее, можно использовать следующее наблюдение. Для того чтобы вычислить a100, можно сначала вычислить a50, а затем возвести результат в квадрат. Аналогично, для вычисления a50 нам потребуется a25. Вычислить a25 таким же способом не получится, так как число 25 — нечётное. Поэтому для вычисления a25 используем значение a24, как мы это делали в предыдущей реализации. В итоге получим вот такую рекурсивную функцию:

**def** power(a, n):

**if** n == 0:

**return** 1

**elif** n % 2 == 1:

**return** power(a, n - 1) \* a

**else**:

a2 = power(a, n // 2)

**return** a2 \* a2

Можно заметить, что такая реализация будет работать значительно быстрее. Так, для n=100 наша функция уже не 100 раз, а только 9 раз вызовет саму себя: a50,a25,a24,a12,a6,a3,a2,a1,a0.