

## Практическая работа №8. Рекурсия Программирование рекурсии в Java. Решение задач на рекурсию

**Цель:** разработка и программирование рекурсивных алгоритмов на языке Java.

### Теоретические сведения

В контексте языка программирования рекурсия — это некий активный метод (или подпрограмма) вызываемый сам по себе непосредственно, или вызываемой другим методом (или подпрограммой) косвенно. В первую очередь надо понимать, что рекурсия — это своего рода перебор. Вообще говоря, всё то, что решается итеративно можно решить рекурсивно, то есть с использованием рекурсивной функции.

Так же, как и у перебора (цикла) у рекурсии должно быть условие остановки — базовый случай (иначе также, как и цикл, рекурсия будет работать вечно — infinite). Это условие и является тем случаем, к которому рекурсия идет (шаг рекурсии). При каждом шаге вызывается рекурсивная функция до тех пор, пока при следующем вызове не сработает базовое условие и не произойдет остановка рекурсии (а точнее возврат к последнему вызову функции). Всё решение сводится к поиску решения для базового случая. В случае, когда рекурсивная функция вызывается для решения сложной задачи (не базового случая) выполняется некоторое количество рекурсивных вызовов или шагов, с целью сведения задачи к более простой. И так до тех пор, пока не получим базовое решение.

Итак, рекурсивная функция состоит из:

- условие остановки или же *базового случая* или условия;
- условие продолжения или *шага рекурсии* — способ сведения сложной задачи к более простым подзадачам.

Рассмотрим это на примере нахождения факториала:

Листинг 8.1 – Пример вычисления факториала

```
public class Solution {  
    public static int recursion(int n) {  
        // условие выхода  
        // Базовый случай  
        // когда остановиться повторять рекурсию ? if (n ==  
1) {  
        return 1;
```

```

    }
    // Шаг рекурсии / рекурсивное условие
    return recursion(n - 1) * n;
}

public static void main(String[] args) {
    System.out.println(recursion(5)); // вызов рекурсивной
    функции
}
}

```

Тут базовым условием является условие, когда  $n=1$ . Так как мы знаем, что

$1!=1$  и для вычисления  $1!$  нам ни чего не нужно. Чтобы вычислить  $2!$  мы можем использовать  $1!$ , т.е.  $2!=1!*2$ . Чтобы вычислить  $3!$  нам нужно  $2!*3...$  Чтобы вычислить  $n!$  нам нужно  $(n-1)!*n$ . Это и является шагом рекурсии.

Иными словами, чтобы получить значение факториала от числа  $n$ , достаточно умножить на  $n$  значение факториала от предыдущего числа.

В сети интернет при объяснении понятия рекурсии часто даются примеры решения задач нахождения чисел Фибоначчи и Ханойская башня

Представлены задачи с различным уровнем сложности. Попробуйте их решить, самостоятельно используя подход, описанный выше.

При решении попробуйте думать рекурсивно и ответить на вопросы:

- какой базовый случай или условие задается в задаче?
- какой шаг рекурсии или рекурсивное условие?

### **Задания на практическую работу №8**

*Замечание. Задания по этой теме выполняются следующим образом: каждый учащийся выполняет от 3 до 5 задач, начиная с номера варианта задания, который соответствует номеру учащегося в журнале группы.*

#### **1. Задание Треугольная последовательность**

Дана монотонная последовательность, в которой каждое натуральное число  $k$  встречается ровно  $k$  раз: 1, 2, 2, 3, 3, 3, 4, 4, 4, 4,...

По данному натуральному  $n$  выведите первые  $n$  членов этой последовательности. Попробуйте обойтись только одним циклом `for`.

#### **2. Задание от 1 до $n$**

Дано натуральное число  $n$ . Выведите все числа от 1 до  $n$ .

#### **3. Задание от $A$ до $B$**

Даны два целых числа  $A$  и  $B$  (каждое в отдельной строке). Выведите все числа от  $A$  до  $B$  включительно, в порядке возрастания, если  $A < B$ , или в порядке убывания в противном случае.

#### 4. Задание Заданная сумма цифр

Даны натуральные числа  $k$  и  $s$ . Определите, сколько существует  $k$ -значных натуральных чисел, сумма цифр которых равна  $d$ . Запись натурального числа не может начинаться с цифры 0.

В этой задаче можно использовать цикл для перебора всех цифр, стоящих на какой-либо позиции.

#### 5. Задание Сумма цифр числа

Дано натуральное число  $N$ . Вычислите сумму его цифр. При решении этой задачи нельзя использовать строки, списки, массивы (ну и циклы, разумеется).

#### 6. Задание Проверка числа на простоту

Дано натуральное число  $n > 1$ . Проверьте, является ли оно простым. Программа должна вывести слово YES, если число простое и NO, если число составное. Алгоритм должен иметь сложность  $O(\log n)$ .

Указание. Понятно, что задача сама по себе не рекурсивна, т.к. проверка числа  $n$  на простоту никак не сводится к проверке на простоту меньших чисел. Поэтому нужно сделать еще один параметр рекурсии: делитель числа, и именно по этому параметру и делать рекурсию.

#### 7. Задание Разложение на множители

Дано натуральное число  $n > 1$ . Выведите все простые множители этого числа в порядке не убывания с учетом кратности. Алгоритм должен иметь сложность  $O(\log n)$

#### 8. Задание Палиндром

Дано слово, состоящее только из строчных латинских букв. Проверьте, является ли это слово палиндромом. Выведите YES или NO.

При решении этой задачи нельзя пользоваться циклами, в решениях на питоне нельзя использовать срезы с шагом, отличным от 1.

#### 9. Задание Без двух нулей

Даны числа  $a$  и  $b$ . Определите, сколько существует последовательностей из  $a$  нулей и  $b$  единиц, в которых никакие два нуля не стоят рядом.

#### 10. Задание Разворот числа

Дано число  $n$ , десятичная запись которого не содержит нулей.

Получите число, записанное теми же цифрами, но в противоположном порядке.

При решении этой задачи нельзя использовать циклы, строки, списки, массивы, разрешается только рекурсия и целочисленная арифметика.

Функция должна возвращать целое число, являющееся результатом работы программы, выводить число по одной цифре нельзя.

#### 11. Задание Количество единиц

Дана последовательность натуральных чисел (одно число в строке), завершающаяся двумя числами 0 подряд. Определите, сколько раз в этой последовательности встречается число 1. Числа, идущие после двух нулей, необходимо игнорировать.

В этой задаче нельзя использовать глобальные переменные и параметры, передаваемые в функцию. Функция получает данные, считывая их с клавиатуры, а не получая их в виде параметров.

#### 12. Задание Вывести нечетные числа последовательности

Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Выведите все нечетные числа из этой последовательности, сохраняя их порядок.

В этой задаче нельзя использовать глобальные переменные и передавать какие-либо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция не возвращает значение, а сразу же выводит результат на экран. Основная программа должна состоять только из вызова этой функции.

#### 13. Задание Вывести члены последовательности с нечетными номерами

Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Выведите первое, третье, пятое и т.д. из введенных чисел. Завершающий ноль выводить не надо.

В этой задаче нельзя использовать глобальные переменные и передавать какие-либо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция не возвращает значение, а сразу же выводит результат на экран. Основная программа должна состоять только из вызова этой функции.

#### 14. Задание Цифры числа слева направо

Дано натуральное число N. Выведите все его цифры по одной, в

обычном порядке, разделяя их пробелами или новыми строками.

При решении этой задачи нельзя использовать строки, списки, массивы (ну и циклы, разумеется). Разрешена только рекурсия и целочисленная арифметика

#### 15. Задание Цифры числа справа налево

Дано натуральное число  $N$ . Выведите все его цифры по одной, в обратном порядке, разделяя их пробелами или новыми строками.

При решении этой задачи нельзя использовать строки, списки, массивы (ну и циклы, разумеется). Разрешена только рекурсия и целочисленная арифметика.

#### 16. Задание Количество элементов, равных максимуму

Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом ноль. Определите, какое количество элементов этой последовательности, равны ее наибольшему элементу.

В этой задаче нельзя использовать глобальные переменные. Функция получает данные, считывая их с клавиатуры, а не получая их в виде параметра. В программе на языке Python функция возвращает результат в виде кортежа из нескольких чисел, и функция вообще не получает никаких параметров. В программе на языке C++ результат записывается в переменные, которые передаются в функцию по ссылке. Других параметров, кроме как используемых для возврата значения, функция не получает. Гарантируется, что последовательность содержит хотя бы одно число (кроме нуля)

#### 17. Задание Максимум последовательности

Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Определите наибольшее значение числа в этой последовательности.

В этой задаче нельзя использовать глобальные переменные и передавать какие-либо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция возвращает единственное значение: максимум считанной последовательности. Гарантируется, что последовательность содержит хотя бы одно число (кроме нуля).

### **Пример решения задачи**

Задание: найти точную степень двойки. Дано натуральное число  $N$ .

Выведите слово YES, если число N является точной степенью двойки, или слово NO в противном случае.

Решение:

```
public class Rec1 {
    public static int recursion(double n) { if (n == 1)
{
    return 1;
}
else if (n > 1 && n < 2) { return 0;
}
else {
return recursion(n / 2);
}
}
    public static void main(String[] args) { double n
= 64;

        if      (recursion(n)      ==      1)      {
System.out.println("Yes");
        } else {
System.out.println("No");
        }
        }
    }
```