



CodingBootcamp

Week 0 Day 3

Аннотация: Этот документ является практическим руководством к Week 0 Day 3 в CodingBootcamp.

Академия Ковалевского



Содержание

1. Рекурсия	2
2. Практическая работа	3





1. Рекурсия

Справка

Определение

Рекурсия — вычислительный процесс, направленный на решение определенной задачи таким образом, что само решение использует этот же процесс, решающий аналогичную подзадачу. В программировании под рекурсией понимают такую реализацию, в которой подпрограмма использует в своем теле вызов самой себя. Такие вызовы называют рекурсивными. Когда функция в своем теле вызывает только одну рекурсивную функцию (саму себя), то говорят о простой рекурсии. Под косвенной рекурсией понимают явление, когда рекурсивные функции вызывают друг друга.

Рекомендации

- Найти информацию о рекурсии в Java
- Посмотреть видео "[Рекурсия](#)"

2. Практическая работа

Класс NumberUtils

factorialNormal

Написать итеративную функцию `factorialNormal(int number)`, которая принимает на вход число `number` типа `int` и возвращает его факториал. Если `number` меньше нуля, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static int factorialNormal(int number);
```

factorialRecursive

Написать рекурсивную функцию `factorialRecursive(int number)`, которая принимает на вход число `number` типа `int` и возвращает его факториал. Если `number` меньше нуля, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static int factorialRecursive(int number);
```

powerNormal

Написать итеративную функцию `powerNormal(int base, int power)` — возведения числа (`base`) в степень (`power`). Если `power` меньше нуля, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static int powerNormal(int base, int power);
```

powerRecursive

Написать рекурсивную функцию `powerRecursive(int base, int power)` — возведения числа (`base`) в степень (`power`). Если `power` меньше нуля, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static int powerRecursive(int base, int power);
```



fibRecursive

Написать рекурсивную функцию `fibRecursive(int index)`, которая возвращает число по заданному индексу. Если `index` меньше нуля, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static int fibRecursive(int index);
```

fibSequence

Написать рекурсивную функцию `fibSequence(int length)`, которая возвращает массив с рядом Фибоначчи заданной длины. Если `length` меньше нуля, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static int[] fibSequence(int length);
```

sqrt

Написать функцию `sqrt(int target)`, которая принимает на вход число `target` типа `int` и возвращает его корень (если такой существует). Если целого корня нет — верните `(-1)`. Если число отрицательное, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static int sqrt(int target);
```

isPrime

Написать функцию `isPrime(int target)`, которая принимает на вход число `target` типа `int` и возвращает `true`, если число простое и `false` в другом случае. Если `target` меньше нуля, ожидается `IllegalArgumentException`.

Прототип функции:

```
1. public static boolean isPrime(int target);
```



nextPrime

Написать функцию `nextPrime(int target)`, которая принимает на вход число `target` типа `int` и возвращает следующее простое число. Если `target` меньше нуля, ожидается `IllegalArgumentException`. Если `target` простое число — вернуть его.

Прототип функции:

```
1. public static int nextPrime(int target);
```

