



CodingBootcamp

Week 0 Day 1

Аннотация: Этот документ является практическим руководством к Week 0 Day 1 в CodingBootcamp.

Академия Ковалевского



Содержание

1. Классы	2
2. Методы	3
3. Типы данных	4
4. Условный оператор if	5
5. Массивы	6
6. Цикл for	7
7. Локальные переменные.....	8
8. Оператор return	9
9. Исключения	10
10. Максимальное и минимальное значения Integer	11
11. Практическая работа	12



1. Классы

Справка

Определение

Классы — это, можно сказать, основа основ программирования на Java. Класс — это, по сути, шаблон для объекта. Он определяет, как объект будет выглядеть и какими функциями обладать. Каждый объект является объектом какого-то класса.

Рекомендации

- Найти информацию о классах в Java
- Посмотреть видеоурок "[Введение в ООП](#)"
- Посмотреть видеоурок "[Типы данных в Java](#)"
- Посмотреть видеоурок "[Работа с экземплярами в Java](#)"
- Посмотреть видеоурок "[Пакеты в Java](#)"



2. Методы

Справка

Определение

Метод — это блок кода, который объявляется внутри класса и может быть использован многократно. Можно сказать, что методы — это строительные "блоки", из которых состоит программа. Например, когда мы вызываем метод `System.out.println()`, начинается выполнение целого ряда операций для того, чтобы в итоге отобразить вывести сообщение в консоль.

Статические методы

Статические методы классов — это такие методы, к которым можно обратиться, не создавая объект данного класса.

Рекомендации

- Найти информацию о методах в Java
- Посмотреть видеоурок "[ОСНОВЫ ИСПОЛЬЗОВАНИЯ МЕТОДОВ В КЛАССАХ](#)"
- Посмотреть видеоурок "[УРОВНИ МЕТОДОВ В JAVA](#)"





3. Типы данных

Справка

Определение

Одной из основных особенностей Java является то, что данный язык является строго типизированным. А это значит, что каждая переменная и константа представляет определенный тип и данный тип строго определен. Тип данных определяет диапазон значений, которые может хранить переменная или константа.

Тип `int`

Тип `int` служит для представления 32-битных целых чисел со знаком. Диапазон допустимых для этого типа значений — от -2147483648 до 2147483647. Чаще всего этот тип данных используется для хранения обычных целых чисел со значениями, достигающими двух миллиардов.

Тип `char`

Символьный тип `char` предназначен для представления элементов из таблицы символов, например, букв или цифр. Поскольку в Java для представления символов в строках используется кодировка Unicode, разрядность типа `char` в этом языке — 16 бит. Диапазон

Рекомендации

- Найти информацию о типах данных в Java
- Посмотреть видеоурок "[Типы данных в Java](#)"



4. Условный оператор IF

Справка

Определение

Иногда нам нужно выполнить различные действия в зависимости от условий. Оператор `if` является основным оператором выбора в Java и позволяет выборочно изменять ход выполнения программы и это одно из основных отличий между программированием и простым вычислением.

Рекомендации

- Найти информацию об условном операторе `if` в Java
- Посмотреть видеоурок "[Условный оператор IF в Java](#)"



5. Массивы

Справка

Определение

Массив — это структура данных, в которой хранятся элементы одного типа. Его можно представить, как набор пронумерованных ячеек, в каждую из которых можно поместить какие-то данные. Доступ к конкретной ячейке осуществляется через её номер. Номер элемента в массиве также называют индексом. В Java массивы однородны, то есть во всех ячейках хранятся элементы одного типа. Так, массив целых чисел содержит только целые числа, массив строк — только строки и т.д.

Рекомендации

- Найти информацию о массивах в Java
- Посмотреть видеоурок “[Массивы в Java](#)”



6. Цикл `for`

Справка

Определение

Циклы являются такой же важной частью программирования, как условные операторы. С помощью циклов можно организовать повторение выполнения участков кода. Циклы, как правило, используются для прохода по одномерным и многомерным массивам и структурам данных для нахождения определенных элементов и дальнейших операций с ними.

Рекомендации

- Найти информацию о цикле `for` в Java
- Посмотреть видеоурок "[Циклы в Java](#)"



7. Локальные переменные

Справка

Определение

Переменная — это область памяти, к которой мы обращаемся за находящимися там данными, используя имя переменной. При этом у этой помеченной именем области есть еще и адрес, выраженный числом и понятный компьютерной системе. Сам этот адрес, в свою очередь, также можно записать в особую переменную. Переменную, содержащую адрес области памяти, называют указателем.

Область видимости

Область видимости переменной — это часть программы, в которой на переменную можно сослаться. Переменные, объявленные внутри блока кода, имеют область видимости блок и называются локальными переменными. Локальные переменные, объявленные в начале метода, имеют область действия блок так же, как и параметры функции, являющиеся локальными переменными.

Рекомендации

- Найти информацию о локальных переменных в Java



8. Оператор `return`

Справка

Определение

Методы могут возвращать некоторое значение. В языке Java для выполнения явного возврата из метода используется оператор `return`. То есть он снова передает управление объекту, который вызвал данный метод. Если метод возвращает значение `void`, то оператор `return` не является необходимым. Если метод возвращает значение, оператор `return` сопровождается некоторым выражением. Значение этого выражения становится возвращаемым значением метода.

Рекомендации

- Найти информацию об операторе `return`



9. Исключения

Справка

Определение

Исключение — это нештатная ситуация, ошибка во время выполнения программы. В методе, в котором происходит ошибка, создаётся и передаётся специальный объект. Метод может либо обработать исключение самостоятельно, либо пропустить его. В любом случае исключение ловится и обрабатывается.

RuntimeException

Эти исключения обычно возникают в результате ошибок программирования, таких как ошибки разработчика или неверное использование интерфейса приложения. Такие ошибки могут быть в любом месте программы, поэтому компилятор не требует указывать runtime исключения в объявлении метода. Теоретически приложение может поймать это исключение, но разумнее исправить ошибку.

IllegalArgumentException

Это исключение самое простое, его легко понять, найти его причину и исправить. Оно случается, когда JVM пытается передать методу неподходящий аргумент (например, отрицательный, когда метод предполагает задание положительных значений) или аргумент неправильного типа.

Рекомендации

- Найти информацию об исключениях в Java
- Посмотреть видеоурок "[Исключения в Java](#)"
- Посмотреть видеоурок "[Иерархия исключений](#)"

10. Максимальное и минимальное значения Integer

Справка

Определение

Класс `Integer` является классом-оберткой примитивного типа данных `int`. Будучи классом-оберткой, `Integer` предоставляет различные константы и методы для работы с целыми числами. Чтобы найти максимальное и минимальное значения `Integer`, можно обратиться к двум его полям. Максимальное значение объекта `Integer` (2147483647) может быть найдено с помощью вызова `Integer.MAX_VALUE`, а минимальное значение `Integer` (-2147483648) — через обращение к `Integer.MIN_VALUE`.

Рекомендации

- Найти информацию об классах-обертках примитивных типов



11. Практическая работа



Класс Alphabet

generateAlphabet

Написать функцию `generateAlphabet()`, которая возвращает массив символов — алфавит (нижний регистр), начиная с 'a' заканчивая 'z' (ascending order).
(Внимание! Реализация должна быть без хардкода, только при помощи циклов).

Прототип функции:

```
1. public static char[] generateAlphabet();
```

generateReverseAlphabet

Написать функцию `generateReverseAlphabet()`, которая возвращает массив символов — алфавит (нижний регистр), начиная с 'z' заканчивая 'a' (descending order).
(Внимание! Реализация должна быть без хардкода, только при помощи циклов).

Прототип функции:

```
1. public static char[] generateReversedAlphabet();
```



Класс Numbers

generateNumbers

Написать функцию `generateNumbers()`, которая возвращает массив символов — цифры, начиная с '0' заканчивая '9' (ascending order).
(Внимание! Реализация должна быть без хардкода, только при помощи циклов).

Прототип функции:

```
1. public static int[] generateNumbers();
```

isNegative

Написать функцию `isNegative(int number)`, которая принимает число и возвращает `true` или `false`, в зависимости от знака принятого числа. Если число отрицательное — `true`. Если положительное или '0' — `false`.

Прототип функции:

```
1. public static boolean isNegative(int number);
```



generateTriplets

Написать функцию `generateTriplets()`, которая возвращает массив всех уникальных комбинаций трех цифр в порядке возрастания, в массиве числа тоже в порядке возрастания: `012, 013, 014, ... 019, 023, ... 789`. Да, `789` последнее число — `987` не подходит, эта комбинация уже есть.

Прототип функции:

```
1. public static String[] generateTriplets();
```

generateTuples1

Написать функцию `generateTuples()`, которая возвращает массив всех уникальных комбинаций двух цифр от `00` до `99`, записанных в порядке возрастания: `00 01, 00 02, 00 03 ... 00 99, 01 02, ... 98 99`.

Прототип функции:

```
1. public static String[] generateTuples();
```

generateTuples2

Написать функцию `generateTuples(int amount)`, которая возвращает массив всех уникальных комбинаций n цифр (n от `1` до `9` включительно), если $n = 2$: `01, 02, 03, 04, ... 09, 12, 13, 14 ... 79, 89`.

Прототип функции:

```
1. public static String[] generateTuples(int amount);
```

Если функция принимает отрицательное число — `throw IllegalArgumentException`.

convertToString

Написать функцию `convertToString(int number)`, которая принимает на вход число, преобразовывает его и возвращает в формате `char[]` — `int` в `char`.

Прототип функции:

```
1. public static char[] convertToString(int number);
```

Не забудьте протестировать с `Integer.MAX_VALUE` и `Integer.MIN_VALUE`.





Требования

Можно использовать

Классы и методы: `StringBuilder`, `String.format`, `String.toCharArray`, `IntStream` (опционально, только если Вы знаете, как использовать; в идеале реализация должна быть без `IntStream`).

Структуры: `for`, `while`, `var`, `arrays`.

Запрещено использовать

Всё, что не разрешено выше — запрещено.

Например, множество `Set`, метод `String.valueOf` или его неявное использование:

```
1. (" " + i);
```