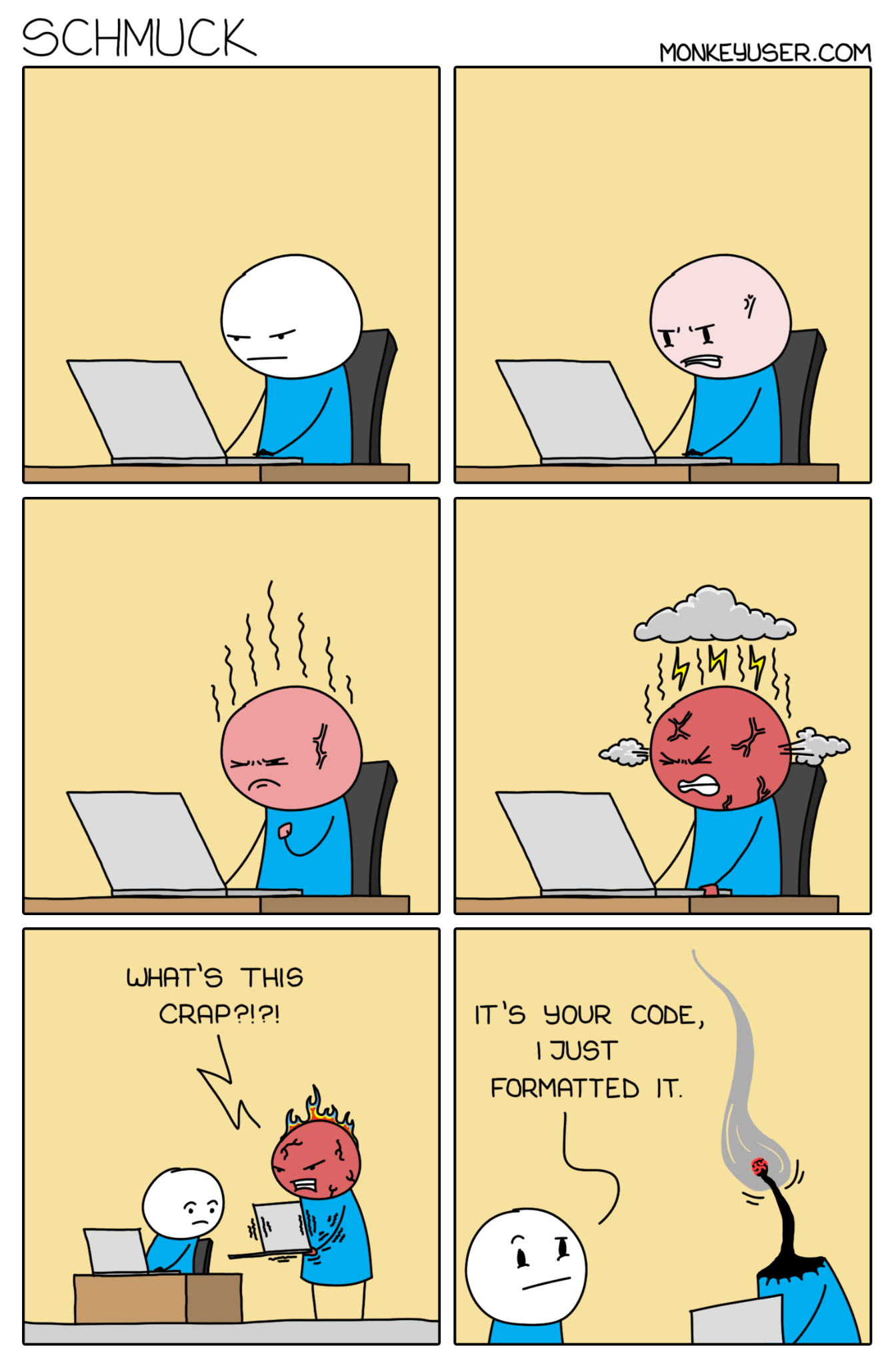
Правила оформления кода на языке Java

**(**[**ссылка**](https://github.com/ichimax/Java-Style-Guide/blob/main/guide.md) **на это же руководство на GitHub)**

****

[***Правила для классов***](#_ieg8mbgihsrd) ***2***

[***Правила для методов***](#_tn53uxeylkbe) ***9***

[***Правила для переменных***](#_s5yo9ea0zsy8) ***14***

[***Общие правила***](#_6atupulybuf9) ***17***

**Перед каждой** сдачей наставнику на проверку выполненное ДЗ, в обязательном порядке сверяйте форматирование своих классов, с указанными ниже правилами (делайте это до тех пор, пока не запомните их наизусть).

Применение данных техник сделает ваш код чистым, читаемым, понятным. **Не пренебрегайте ими!** Будьте предельно внимательными.

## Правила для классов

1. Имя класса должно быть существительным и состоять из одного или более слов. Оно не должно содержать глагол

неправильно:

| **class** **Sit** **class** **SaveInteger** **class** **CalculateSum** |
| --- |

правильно:

| **class** **SortedArrayStorage** **class** **CalculatorMain** **class** **Resume** |
| --- |

1. Не оставляйте пустые строки после объявления класса и перед последней }

неправильно:

| **public** **class** **Player** {   **private** String name;  **public** String **getName**() {  **return** name;  }  // лишняя пустая строка }  // лишняя пустая строка |
| --- |

правильно:

| **public** **class** **Player** {   **private** String name;   **public** String **getName**() {  **return** name;  } } |
| --- |

1. После имени класса и перед { необходим пробел

неправильно:

| **public** **class** **Calculator**{ |
| --- |

правильно:

| **public** **class** **Calculator** { |
| --- |

1. Поля необходимо размещать в начале класса. При этом все они должны быть сгруппированы в одном месте

неправильно:

| **public** **class** **Wolf** {   **private** String sex;  **public** String **getSex**() {  **return** sex;  }   **private** String name;  **public** String **getName**() {  **return** name;  } } |
| --- |

правильно:

| **public** **class** **Wolf** {   **private** String sex;  **private** String name;   **public** String **getSex**() {  **return** sex;  }   **public** String **getName**() {  **return** name;  } } |
| --- |

1. Между блоком полей и конструктором необходима пустая строка

неправильно:

| **public** **class** **Resume** {   **private** String uuid;  **private** String name  **public** **Resume**(String uuid) {  **this**.uuid = uuid;  } } |
| --- |

правильно:

| **public** **class** **Resume** {   **private** String uuid;  **private** String name  **public** **Resume**(String uuid) {  **this**.uuid = uuid;  } } |
| --- |

1. Группируйте геттеры и сеттеры по именам переменных, для которых они созданы. При этом не “приклеивайте” их друг к другу

неправильно:

| **public** **class** **Jaeger** {   **private** String mark;  **private** String origin;   **public** String **getMark**() {  **return** mark;  }  **public** String **getOrigin**() {  **return** origin;  }   **public** **void** **setMark**(String mark) {  **this**.mark = mark;  }  **public** **void** **setOrigin**(String origin) {  **this**.origin = origin;  } } |
| --- |

правильно:

| **public** **class** **Jaeger** {   **private** String mark;  **private** String origin;   **public** String **getMark**() {  **return** mark;  }   **public** **void** **setMark**(String mark) {  **this**.mark = mark;  }   **public** String **getOrigin**() {  **return** origin;  }    **public** **void** **setOrigin**(String origin) {  **this**.origin = origin;  } } |
| --- |

1. Размещайте конструктор(ы) после всех полей

неправильно:

| **class BankAccount {  private String name;   String getName() {  return name;  }   public BankAccount(String name) {  this.name = name;  } }** |
| --- |

правильно:

| **class** **BankAccount** {  **private** String name;   **public** **BankAccount**(String name) {  **this**.name = name;  }   String **getName**() {  **return** name;  } } |
| --- |

1. Размещайте геттеры и сеттеры в порядке следования полей, к которым они относятся

неправильно:

| **class** **Player** {   **private** String name;  **private** **int** number;   **int** **getNumber**() {  **return** number;  }   **void** **setNumber**(**int** number) {  **this**.number = number;  }   String **getName**() {  **return** name;  }   **void** **setName**(String name) {  **this**.name = name;  } } |
| --- |

правильно:

| **class** **Player** {   **private** String name;  **private** **int** number;   String **getName**() {  **return** name;  }   **void** **setName**(String name) {  **this**.name = name;  }    **int** **getNumber**() {  **return** number;  }   **void** **setNumber**(**int** number) {  **this**.number = number;  } } |
| --- |

1. Размещайте конструкторы в классе в порядке, зависящем от принимаемых ими числа аргументов: от меньшего к большему

неправильно:

| **class** **Player** {   Resume(String name, **int** number) {}   Resume(String name) {}   Resume(String name, **int** number, String uuid) {} } |
| --- |

правильно:

| **class** **Player** {   Resume(String name) {}   Resume(String name, **int** number) {}   Resume(String name, **int** number, String uuid) {} } |
| --- |

1. В одной строке объявляйте за раз только одну переменную

неправильно:

| **class** **Person** {  String uuid, name;  **double** height, weight; } |
| --- |

правильно:

| **class** **Person** {  String uuid;  String name;  **double** height;  **double** weight; } |
| --- |

1. Между именем конструктора и открывающейся круглой скобкой пробел не требуется

неправильно:

| Player (String name) Resume (String uuid) |
| --- |

правильно:

| Player(String name) Resume(String uuid) |
| --- |

1. Если геттеры не используются, то в классе их можно не писать

## Правила для методов

1. Имя метода должно быть глаголом или содержать глагол и состоять из одного или более слов. Первым в имени должен идти глагол

неправильно:

| **void** **endGame**() **int** **calculation**() String **playerArray**() |
| --- |

правильно:

| **int** **calculate**() **void** **clearNumbers**() **void** **start**() |
| --- |

1. Имена методов пишите с маленькой буквы

неправильно:

| **char** **GetLetter**() **void** **ClearEnteredNumbers**() Player **CreatePlayer**() |
| --- |

правильно:

| **char** **getLetter**() **void** **clearEnteredNumbers**() Player **createPlayer**() |
| --- |

1. Между именем метода и открывающейся круглой скобкой пробел не требуется

неправильно:

| **public** **static** **void** **main** (String[] args) **public** **int** **getIndex** (String uuid) System.out.**println** ("Math operator is wrong!") |
| --- |

правильно:

| **public** **static** **void** **main**(String[] args) **public** **int** **getIndex**(String uuid) System.out.**println**("Math operator is wrong!") |
| --- |

1. После сигнатуры любого метода пустая строка не ставится

неправильно:

| **public** **static** **void** **main**(String[] args) {   **int** number = 5; } |
| --- |

правильно:

| **public** **static** **void** **main**(String[] args) {  **int** number = 5; } |
| --- |

\*я это называю программерской клаустрофобией. Ей страдают по началу все новички. В интернете полно примеров, где пустая строка ставится. Но не делайте так. Нет никаких правил или обоснований это делать

1. Между любыми методами всегда оставляйте одну пустую строку

неправильно:

| **class** **Jaeger** {  **private** String name;    **public** **void** **setName**(String name) {  **this**.name = name;  }  **public** String **getName**() {  **return** name;  }   **public** **void** **move**() {  // some code  }  **public** **void** **fire**() {  // some code  } } |
| --- |

правильно:

| **class** **Jaeger** {  **private** String name;    **public** **void** **setName**(String name) {  **this**.name = name;  }   **public** String **getName**() {  **return** name;  }   **public** **void** **move**() {  // some code  }   **public** **void** **fire**() {  // some code  } } |
| --- |

1. Геттеры и сеттеры для boolean-поля именуются по специальному правилу: для геттера вместо get пишется префикс is, а имя boolean-переменной должно быть прилагательным

правильно:

| **class** **Car** {   **private** **boolean** active;   **boolean** **isActive**() {  **return** active;  }   **void** **setActive**(**boolean** active) {  **this**.active = active;  } } |
| --- |

1. Используйте this только при конфликте имен, например в сеттерах (и то не всегда). Если конфликта нет, писать this не нужно

будет работать, но использование this избыточно:

| **public** **int** **getIndex**() {  **return** **this**.index; }  **public** **void** **setEnteredNumber**(**int** number) {  **this**.enteredNumbers[index] = number;  } |
| --- |

правильно:

| **public** **void** **setIndex**(**int** index) {  **this**.index = index; } |
| --- |

1. Размещайте методы в классе в порядке их вызовов, а не хаотично, как получится. Код должен читаться сверху вниз, как текст в книге

неправильно:

| **private** **boolean** **makeMove**() {  inputNumber();  **return** compareNumbers(); }  **private** **void** **inputNumber**() {  // some code }  **public** **void** **start**() {  makeMove(); }  **private** **boolean** **compareNumbers**() {  **return** **true**; } |
| --- |

правильно:

| **public** **void** **start**() {  makeMove(); }  **private** **boolean** **makeMove**() {  inputNumber();  **return** compareNumbers(); }  **private** **void** **inputNumber**() {  // some code }  **private** **boolean** **compareNumbers**() {  **return** **true**; } |
| --- |

1. Если поле является массивом и называется, например enteredNums, то как правильно назвать его сеттер, если он принимает за раз только одно число?

неправильно:

| **private** **int**[] enteredNums; **private** **int** index;  **void** **setEnteredNums**(**int** enteredNums) {  this.enteredNums[index] = enteredNums; } |
| --- |

правильно:

| **private** **int**[] enteredNums; **private** **int** index;  **void** **setEnteredNum**(**int** enteredNum) {  enteredNums[index] = enteredNum; } |
| --- |

## Правила для переменных

1. Давайте переменным осмысленные и понятные имена, глядя на которые, любому читающему код, было бы понятно, какие данные они хранят

неправильно:

| **char** a = '+'; **int** b = 10; String с = "yes"; |
| --- |

правильно:

| **char** sign = '+'; **int** size = 10; String answer = "yes"; |
| --- |

1. Объявляйте переменные максимально близко к месту их первого использования. Не группируйте их в начале метода

1. Имя переменной не должно быть глаголом или содержать глагол

неправильно:

| Resume[] getAll = storage.getAll(); |
| --- |

правильно:

| Resume[] resumes = storage.getAll(); или Resume[] AllResume = storage.getAll(); |
| --- |

1. Перед переменной, при приведении типа, требуется пробел

неправильно:

| (**char**)i |
| --- |

правильно:

| (**char**) i |
| --- |

1. Имена переменных пишите с маленькой буквы

неправильно:

| **int** Age; **int** Title; **int** FullName |
| --- |

правильно:

| **int** age; **int** title; **int** fullName |
| --- |

1. Аббревиатуры в Java записываются, как имена обычных переменных

неправильно:

| **int** RAM; **long** freqCPU; String OS; |
| --- |

правильно:

| **int** ram; **long** freqCpu; String os; |
| --- |

1. При использовании сокращенной формы инкремента и декремента, после имени переменной пробел не ставится

неправильно:

| number ++ **for** (**int** i = 10; i > 0; i --) |
| --- |

правильно:

| number++ **for** (**int** i = 10; i > 0; i--) |
| --- |

1. При проверке boolean-значений явно указывать true или false не нужно

будет работать, но использование true/false избыточно:

| **if** (male == **false**) **while** (isNext() == **true**) **if** (isExist(uuid) != **true**) |
| --- |

правильно:

| **if** (!male) **while** (isNext()) **if** (!isExist(uuid)) |
| --- |

1. При именовании переменных, для разделения слов, нижнее подчеркивание \_ ставьте только у констант. Все остальные переменные должны именоваться с использованием camelCase

неправильно:

| **int** hidden\_number; String player\_answer; |
| --- |

правильно:

| **int** hiddenNumber; String playerAnswer; |
| --- |

1. Инициализируйте переменные (если это возможно) в строке их объявления, а не где-то ниже в коде

неправильно:

| **double** num1; **char** sign; num1 = 4; sign = '\*'; |
| --- |

правильно:

| **double** num1 = 4; **char** sign = '\*'; |
| --- |

## Общие правила

1. Перед и после операторов =, +, -, /, \*, ==, !=, <, <=, >, >= требуется пробел

неправильно:

| System.out.println("дата="+m+"."+d+"."+y); **int** length=5; **for** (**int** i=0;i<length;i++) |
| --- |

правильно:

| System.out.println("дата = " + m + "." + d + "." + y); **int** length = 5; **for** (**int** i = 0; i < length; i++) |
| --- |

1. Между ){ скобками требуется пробел

неправильно:

| **public** **static** **void** **main**(String[] args){ **for** (**int** i = 10; i > 0; i--){ **if** (a > 10){ |
| --- |

правильно:

| **public** **static** **void** **main**(String[] args) { **for** (**int** i = 10; i > 0; i--) { **if** (a > 10) { |
| --- |

1. Форматирование фигурных скобок

неправильно (так форматируют в C# и C++):

| foo() { } |
| --- |

правильно:

| foo() { } |
| --- |

1. Форматирование else и else if

неправильно:

| } **else** {  System.out.println("Exit"); } |
| --- |

правильно:

| } **else** {  System.out.println("Exit"); } |
| --- |

неправильно:

| } **else** **if** (size < storage.length) { |
| --- |

правильно:

| } **else** **if** (size < storage.length) { |
| --- |

плохо (зачем тратить лишнюю строку под if):

| } **else** {  **if** (number > randomNumber) { |
| --- |

хорошо:

| } **else** **if** (number > randomNumber) { |
| --- |

1. Форматирование цикла do-while

неправильно:

| **do** {  // some code } **while** (answer.equals("yes")); |
| --- |

правильно:

| **do** {  // some code } **while** (answer.equals("yes")); |
| --- |

1. Если ветка if-else или тело цикла состоит из одного выражения, всегда заключайте их в {}. Это поможет избежать трудно уловимых багов

плохо:

| **if** (doc.isImportant)  send(doc); **else**  System.out.println("Throw out the trash");  delete(doc); |
| --- |

плохо:

| **while** (number < 5)  System.out.println("number less");  number++; |
| --- |

хорошо:

| **if** (doc.isImportant) {  send(doc); } **else** {  log.info("Throw out the trash");  delete(doc); } |
| --- |

хорошо:

| **while** (number < 5) {  System.out.println("number less");  number++; } |
| --- |

1. После , и ; требуется пробел

неправильно:

| **public** **void** **addAttempt**(**int** number,**int** i) **for** (**int** i = 0;i < storage.length;i++) |
| --- |

правильно:

| **public** **void** **addAttempt**(**int** number, **int** i) **for** (**int** i = 0; i < storage.length; i++) |
| --- |

1. После ( и перед ) скобками пробел не требуется

неправильно:

| **for** ( **int** i = 0; i < 21; i++ ) compareNumber( String number ) |
| --- |

правильно:

| **for** (**int** i = 0; i < 21; i++) compareNumber(String number) |
| --- |

1. Если класс называется Player, то у его полей или методов слово Player уже использовать не нужно, т.к. и так понятно, что все, что там находится относится к Player, а не к Cat или к колбасе :)

неправильно:

| **public** **class** **Player** {    **private** **int**[] playerNumbers = **new** **int**[10];   **public** **int** **getPlayerNumber**(**int** index) {  **return** playerNumbers[index];  } } |
| --- |

правильно:

| **public** **class** **Player** {    **private** **int**[] numbers = **new** **int**[10];   **public** **int** **getNumber**(**int** index) {  **return** numbers[index];  } } |
| --- |

1. После package и import требуется пустая строка

неправильно:

| **package** com.webapp.storage; **import** java.util.Scanner; **public** **class** **ArrayStorage** { |
| --- |

правильно:

| **package** com.webapp.storage;  **import** java.util.Scanner;  **public** **class** **ArrayStorage** { |
| --- |

1. Размещайте комментарии над комментируемым кодом, а не справа от него. Комментарий должен начинаться с той же позиции, что и код

неправильно:

| **int** sumOdd = 10; // сумма нечетных чисел |
| --- |

правильно:

| // сумма нечетных чисел **int** sumOdd = 10; |
| --- |

1. При создании переменной для хранения ссылки на массив и самого массива, размещайте [] в Java-стиле, а не в Си или С++, а также не ставьте лишних пробелов

неправильно:

| **int** numbers[] **int** [] numbers **int** numbers [] **new** **int** [10] |
| --- |

правильно:

| **int[]** numbers = **new** **int**[10] |
| --- |

1. Имя boolean-переменной должно быть прилагательным

неправильно:

| **boolean** security **boolean** answer **boolean** continue |
| --- |

правильно:

| **boolean** active **boolean** current **boolean** alive |
| --- |

1. Не забывайте код, вложенный в класс, метод, цикл и другие конструкции, которые имеют тело в виде { } каждый раз сдвигать вправо на 4 пробела (или один tab) относительно начала этой конструкции

неправильно:

| **public** **class** **Person** {   **int** age = 25;   **public** **static** **void** **main**(String[] args) {  **if** (age > 20) {  System.out.println("Человек старше 20 лет");  } } } |
| --- |

правильно:

| **public** **class** **Person** {   **int** age = 25;   **public** **static** **void** **main**(String[] args) {  **if** (age > 20) {  System.out.println("Человек старше 20 лет");  }  } } |
| --- |

1. Если имя переменной/метода/класса состоит из более, чем одного слова, то для именование используется camelCase

неправильно:

| **int** playeranswer **class** **Guessnumbermain** **void** **addnumber()** |
| --- |

правильно:

| **int** playerAnswer **class** **GuessNumberMain** **void** **addNumber()** |
| --- |

1. Стандартное именование для счетчиков в цикле for - это i, j, k. Последние два используются для вложенных циклов

плохо:

| **for** (a = 0; a < 3; a++) {  } |
| --- |

плохо:

| **for** (d = 0; d < 10; d++) {  **for** (c = 0; c < 4; c++) {   } } |
| --- |

хорошо:

| **for** (i = 0; i < 3; i++) {  } |
| --- |

хорошо:

| **for** (i = 0; i < 10; i++) {  **for** (j = 0; j < 4; j++) {   } } |
| --- |

1. При переносе строки каждую следующую ее подстроку необходимо смещать вправо на 8 пробелов относительно первой строки

неправильно:

| System.out.println("Characteristics:\n" + "age = " + age + "\n" + "name = " + name + "\n" + "color = " + color); |
| --- |

правильно:

| System.out.println("Characteristics:\n" +  "age = " + age + "\n" +  "name = " + name + "\n" +  "color = " + color); |
| --- |