

# Логические операторы и цепочки

Пишем логику на все случаи жизни

## Содержание

1. Ключевое слово else
2. Еще больше логики

### 1. Ключевое слово else

Для начала напишем код из задания в конце предыдущей лекции

```
1 public class Main {
2
3     public static void main(String[] args) {
4         showDivision(10,5);
5         showDivision(1, 0);
6     }
7
8     private static void showDivision(int number1, int number2) {
9         if (number2 != 0) {
10             print(number1/number2);
11         }
12     }
13
14     private static void print(int number) {
15         System.out.println(number);
16     }
17 }
18
```

Main

Run: Main x

/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...

2

Process finished with exit code 0

Итак, мы написали простую проверку, чтобы второй аргумент не был нулем. Но думаю каждый согласится, что не так уж и хорошо вызывать функцию и не видеть ничего в консоли. Хотелось бы иметь понимание что произошло не читая код функции. Было бы неплохо выводить в консоль информацию о том, что второй аргумент ноль. В принципе нам ничего не мешает написать еще один if. Посмотрим.

```
1 public class Main {
2
3     public static void main(String[] args) {
4         showDivision(10,5);
5         showDivision(1, 0);
6     }
7
8     private static void showDivision(int number1, int number2) {
9         if (number2 != 0) {
10             print(number1/number2);
11         }
12         if (number2 == 0) {
13             print("second argument is zero!");
14         }
15     }
16
17     private static void print(int number) {
18         System.out.println(number);
19     }
20
21     private static void print(String text) {
22         System.out.println(text);
23     }
24 }
25
```

Main > print()

Run: Main x

/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...

2

second argument is zero!

Process finished with exit code 0

Теперь мы в консоли сразу видим что пошло не так. Хорошо. Но давайте посмотрим на код метода showDivision. Выглядит вполне правильно, но хотелось бы улучшить. Ведь сами посудите. Число или равно нулю или нет. Так? Третьего же не дано. Значит, ровно так же как и с 2 значениями типа boolean должна быть структура чтобы рассмотреть отличный от первого вариант. Я имею ввиду – если число равно нулю, то сделать одно, во всех остальных случаях – поступить по-другому. И конечно же в языке Java есть такое ключевое слово else.

Вы можете поставить брейкпойнты на линии 9 и посмотреть что происходит во время дебагинга. Если второй аргумент равен нулю, то выводим информацию об этом. Иначе (т.е. если не равно нулю) можно найти результат деления и отобразить его. Мне кажется это не так уж и сложно. Главное запомните одно хорошее правило, старайтесь писать именно так как у нас в примере.

```
if (condition) {
    doCode1();
} else {
    doCode2();
}
```

```
1 public class Main {
2
3     public static void main(String[] args) {
4         showDivision(10,5);
5         showDivision(1, 0);
6     }
7
8     private static void showDivision(int number1, int number2) {
9         if (number2 == 0) {
10             print("second argument is zero!");
11         } else {
12             print(number1/number2);
13         }
14     }
15
16     private static void print(int number) {
17         System.out.println(number);
18     }
19
20     private static void print(String text) {
21         System.out.println(text);
22     }
23 }
24
```

Main > showDivision()

Run: Main x /usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...

2  
second argument is zero!

Process finished with exit code 0

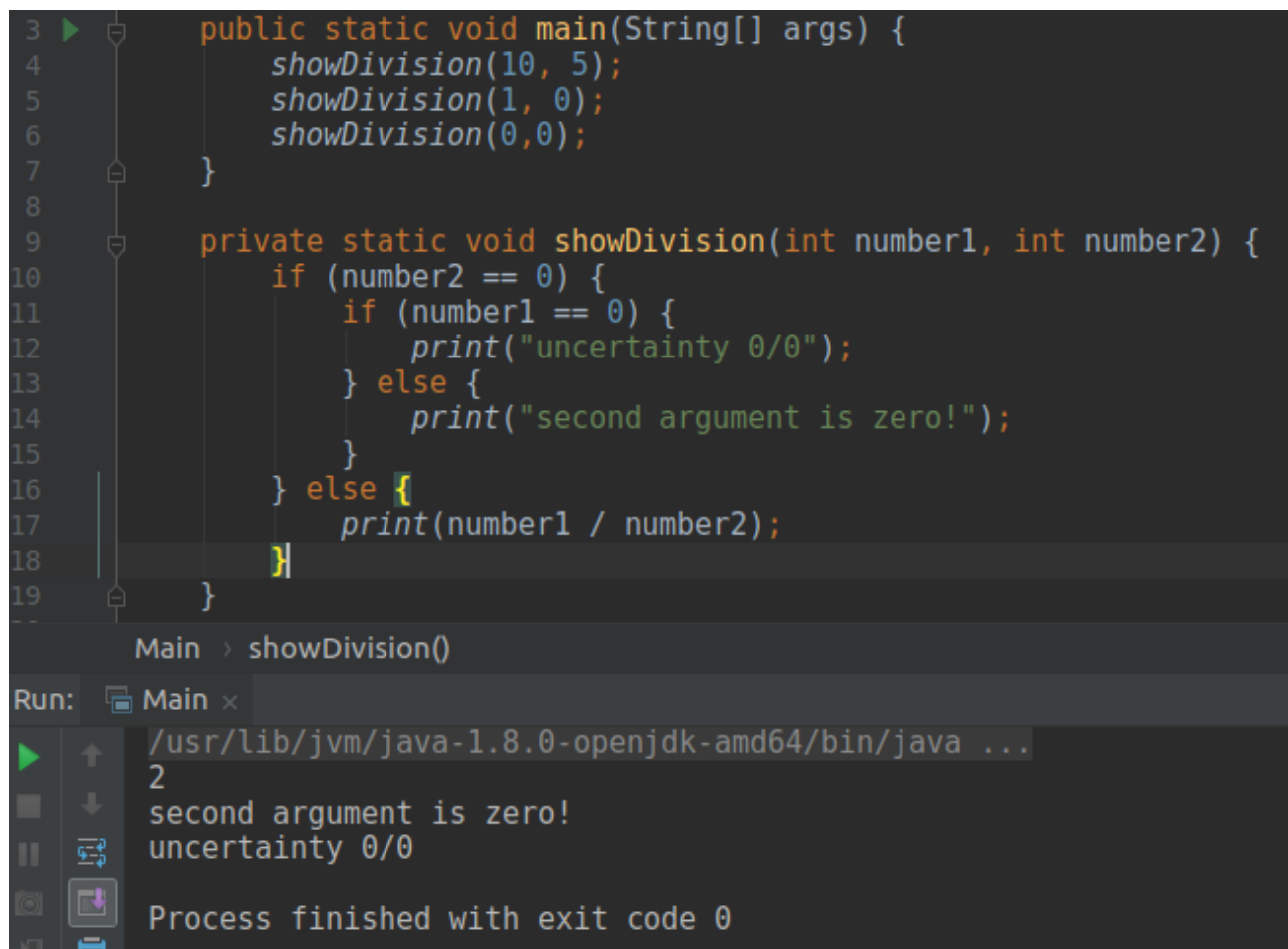
Если же вы видите что написано обратное, то просто ставим курсор на слове if и нажимаем Alt+Enter и выбираем Invert if condition. Согласитесь, лишний знак ! Зачем менять на обратное условие, делать код 2 и потом во всех остальных случаях делать код1. Если можно не писать “!” и выполнить код1 если выполняется условие. Без лишних действий.

```
if (!condition) {
    doCode2();
} else {
    doCode1();
}
```

## 2. Еще больше логики

Итак, у нас простой код для деления. Но истинные математики негодуют. Подождите, могут сказать они, ведь нужно проверять не только второй аргумент на ноль, но и первый. Чего? Да, друзья мои. Если вы истинный ценитель математики, то вы так же можете хотеть избежать ситуации 0/0. Ведь это... НЕОПРЕДЕЛЕННОСТЬ!

Хорошо, значит нам нужно сделать следующее – проверить что оба аргумента ноль. Как это сделать? На самом деле легко. Ведь у нас уже есть одна проверка на ноль на линии 9. Значит на линии 10 мы точно знаем, что аргумент2 ноль. Значит нам осталось проверить аргумент1 на ноль. Давайте сделаем это.



```
3 public static void main(String[] args) {
4     showDivision(10, 5);
5     showDivision(1, 0);
6     showDivision(0,0);
7 }
8
9 private static void showDivision(int number1, int number2) {
10     if (number2 == 0) {
11         if (number1 == 0) {
12             print("uncertainty 0/0");
13         } else {
14             print("second argument is zero!");
15         }
16     } else {
17         print(number1 / number2);
18     }
19 }
20
21 Main > showDivision()
```

Run: Main x

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
2
second argument is zero!
uncertainty 0/0
Process finished with exit code 0
```

Итак, давайте поработаем вместо компилятора и подумаем, как он пойдет когда дойдет до линии 6. Он перейдет на линию 10. Там он проверит, что второй аргумент ноль, после чего пойдет проверять на ноль первый аргумент на линии 11. И выведет строку на линии 12. А какой же порядок будет для линии 5? Будет 10, 11, 14. Логично? Вроде несложно, правда?

Это называется вложенные условия, вложенные if-ы. И если вы подумали о том, что это смотрится немного страшно, то вы совершенно правы. Но знаете, иногда вложенность нужна и мы позднее увидим в каких случаях (забегая вперед – когда вы не знаете ничего о переменной и пытаетесь ее использовать раньше чем нужно).

Теперь, как же мы можем избавиться от вложенности? До сих пор мы проверяли лишь одно условие за раз. `Number2==0`, `text.isEmpty()`. А мы не могли бы проверить 2 условия сразу? Ну в нашем случае мы хотим знать, не равны ли нулям оба аргумента одновременно. И да, самое время познакомиться с логическим оператором `&&`. Из английского мы знаем что этот символ (ampersand) заменяет слово AND. И довольно логично использовать его и в языках программирования. Но заметьте что их 2 так же как и в сравнении чисел через двойное `==`.

Давайте попробуем переписать код метода `showDivision`.

```
9 private static void showDivision(int number1, int number2) {
10     if (number2 == 0 && number1 == 0) {
11         print("uncertainty 0/0");
12     } else {
13         if (number2 == 0) {
14             print("second argument is zero!");
15         } else {
16             print(number1 / number2);
17         }
18     }
19 }
```

Main > showDivision()

Run: Main x

/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...  
2  
second argument is zero!  
uncertainty 0/0

Так, стоп. У нас опять вложенный if. Теперь он переключался в else. Давайте разберемся почему. Итак линия 10. Если и первый аргумент и второй одновременно равны 0, то вывести неопределенность. А иначе... что же значит иначе? У нас есть первое условие и оно требует чтобы оба аргумента были ноль. Что в этом случае значит else иначе? А значит следующее: что оба аргумента не равны нулю одновременно. т.е. они могут быть нулями по отдельности? Да, именно так. Вы можете это проверить дебагером. После линии 5 мы перейдем на 13.

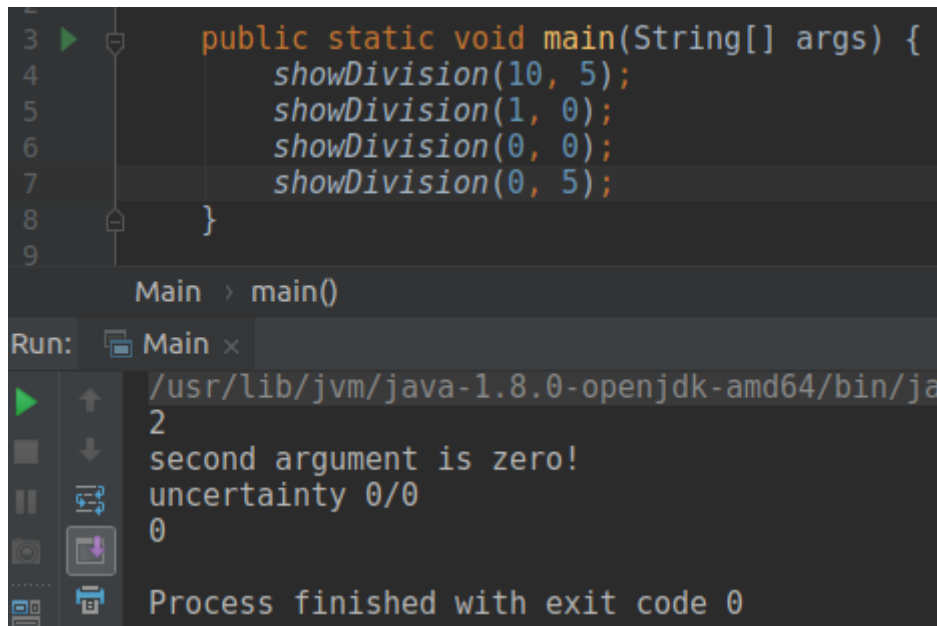
Еще раз. Если оба аргумента не равны нулю одновременно означает что они могут быть нулями по отдельности. Т.е. или первый ноль или второй, но не оба сразу. Надеюсь понятно. Так, разобрались с этим. Но почему у нас опять осталась вложенность? А потому что у нас 3 случая.

1. Оба аргумента ноли
2. Второй аргумент ноль
3. Ноль только в первом аргументе может быть

И чтобы избежать вложенности нам нужна новая конструкция, ведь простой if else не справляется. Т.е. нам нужно объединить эти 2 ключевых слова else if. Давайте посмотрим

```
9 private static void showDivision(int number1, int number2) {
10     if (number2 == 0 && number1 == 0) {
11         print("uncertainty 0/0");
12     } else if (number2 == 0) {
13         print("second argument is zero!");
14     } else {
15         print(number1 / number2);
16     }
17 }
```

Мы получили следующее: сначала проверяем что оба аргумента ноль. Иначе (т.е. не оба сразу ноль, но один из них ноль, неважно какой) если второй аргумент ноль. Иначе же (т.е. когда не выполнилось ни первое ни второе условие, т.е. когда оба аргументы не ноль одновременно и когда не ноль второй аргумент, значит из первого условия мы поняли что один из аргументов не ноль, а из второго что именно второй аргумент не ноль) просто делим одно число на второе понимая что первый аргумент все же может быть нулем.



The screenshot shows an IDE with a Java file named `Main.java`. The code defines a `main` method that calls `showDivision` with four different pairs of arguments: `(10, 5)`, `(1, 0)`, `(0, 0)`, and `(0, 5)`. Below the code editor, the 'Run' tab is active, showing the execution output. The output starts with the command `/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java` followed by the arguments `2` and `second argument is zero!`. The output then shows `uncertainty 0/0` and `0`. Finally, it states `Process finished with exit code 0`.

```
public static void main(String[] args) {  
    showDivision(10, 5);  
    showDivision(1, 0);  
    showDivision(0, 0);  
    showDivision(0, 5);  
}
```

Run: Main x

/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java  
2  
second argument is zero!  
uncertainty 0/0  
0  
Process finished with exit code 0

Можете сами разобраться с помощью дебагера. На самом деле код метода `showDivision` можно написать иначе. И я вам предлагаю сделать это самостоятельно.

Если у вас это получилось, то попробуйте написать метод, который принимает 2 логических переменных `isTodayFriday`, `aLotOfMoney`, т.е. сегодня пятница и у вас много денег. Выведите в консоль следующее

если пятница и у вас много денег, то пойти в бар выпить пивка

если пятница, но денег немного, то попросить в долг у друга и пойти с ним попить пивка

во всех остальных случаях (если не пятница и денег нет) пойти домой и поесть гречи

Текстовки можете придумать сами.

И вот вам второе задание для закрепления темы

метод принимает 3 числа на вход : стоимость пива, стоимость виски и деньги в кармане

если денег в кармане хватает на виски, то берем виски

если денег в кармане хватает только на пиво, то берем пиво

если денег в кармане хватает и на пиво и на виски, то берем и то и другое

но если у вас ветер в кармане – покидаем бар