QA
מסע

אלכס גורבצ'וב

АЛЕКСЕЙ ГОРБАЧЁВ

JAVA

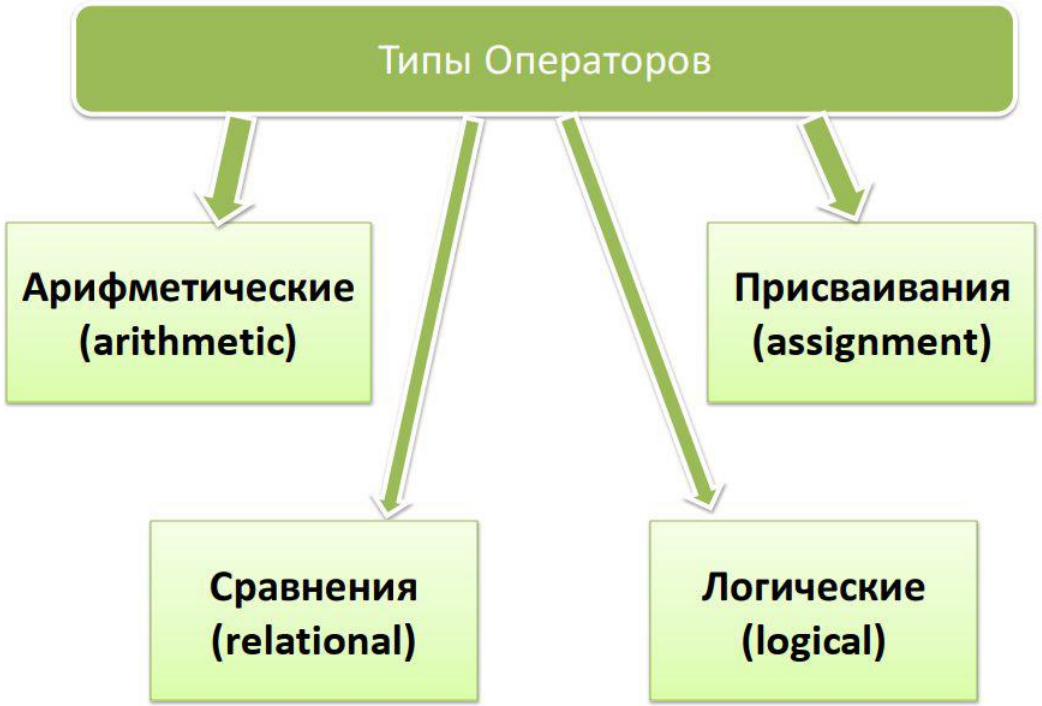
JAVA

Java

Основные операции с переменными в Java

В Java все операторы можно разбить на 4 группы операторов:

- Арифметические
- Присваивания
- Логические
- Сравнения



Java

Основные операции с переменными в Java

Java

Основные операции с переменными в Java

Java

Основные операции с переменными в Java

Операторы сравнения

>	>=
<	<=
==	!=

Основные операции с переменными в Java

Логические операторы

ОПЕРАТОР	ПРИМЕР ИСПОЛЬЗОВАНИЯ	ВОЗВРАЩАЕТ ЗНАЧЕНИЕ "ИСТИННО", ЕСЛИ...
>	a > b	a больше b
>=	a >= b	a больше или равно b
<	a < b	a меньше b
<=	a <= b	a меньше или равно b
==	a == b	a равно b
!=	a != b	a не равно b
&&	a && b	a и b истинны, b оценивается условно (если a ложно, b не вычисляется)
||	a || b	a или b истинно, b оценивается условно (если a истинно, b не вычисляется)
!	!a	a ложно
&	a & b	a и b истинны, b оценивается в любом случае
|	a | b	a или b истинно, b оценивается в любом случае

Java

Арифметические операторы

В Java кроме нам знакомых существуют и дополнительные операторы:

- Сложение (+)
- Вычитание (-)
- Умножение (*)
- Деление (/)
- Остаток (%)

Сокращённые арифметические операторы

Также к арифметическим операторам относят так называемые сокращённые арифметически операторы. Есть 2 формы записи основных арифметических операций:

```
m += 7;  // это всё равно, что m = m+7;
m -= 7 ; // это всё равно, что m = m-7;
m*= 7; // это всё равно, что m = m*7;
m/= 7; // это всё равно, что m = m/7;
```

Зачем придумали сокращённые арифметические операторы? Да потому что лень - двигатель прогресса! Когда пишешь много-много строчек кода, каждый лишний символ отнимает силы и время. Проще использовать сокращённые арифметические операторы.

Java

Инкрементация и декрементация

В программировании очень часто приходится выполнять операции, когда:

- переменная должна увеличиться на единицу
- переменная должна уменьшиться на единицу

Поэтому придумали отдельные операции с переменными, которые называются инкрементация и декрементация.

- Инкрементация - отвечает за увеличение переменной на единицу. Обозначается как ++. Например, если у нас есть переменная i и мы к ней применим инкремент, тогда это будет записано как i++. А это значит, что значение переменной i должно быть увеличено на 1.
- Декрементация - отвечает за уменьшение переменной на единицу. Обозначается как --. Например, если у нас есть переменная n и мы к ней применим декремент, тогда это будет записано как n --. А это значит, что значение переменной n должно быть уменьшено на 1.

Две формы инкрементации и декрементации

Существует 2 формы:

- постфиксная (n++, n--)
- префиксная (++n, --n)

В чём отличие между постфиксной и префиксной формами?

В постфиксной форме:

- сначала используется старое значение в вычислениях
- далее в последующих вычислениях используется уже новое значение

В префиксной форме:

- сразу используется новое значение в вычислениях

Java

Задание.

- Вычислите следующую часть кода:

```
int i1=5;
int i2=11;
double d1 = 5.5;
double d2 = 1.3;
long l = 20l;
double result=0;
result = i2 / d1 + d2 % i1 - l;
```

- Чему равны выражения:

a-- - --a + ++a + a++ + a; где a=5

++b - b++ + ++b - --b; где b=8

Java

Задание.

Имеется 3 переменные типа int x = 10, y = 12, и z = 3;

- Выполните и рассчитайте результат следующих операций для этих переменных:

```
x += y - x++ * z;
z = --x – y * 5;
y /= x + 5 % z;
z = x++ + y * 5;
x = y - x++ * z;
```

Java

Конкатенация строк в Java

Конкатенация - это синоним слова "объединение". Истоки слова "конкатенация" от латыни: "con" означает "вместе", "catena" в переводе с латыни это "цепь". По-русски будет «сцеплять».

Где используется конкатенация в Java?

В основном - при работе со строками. Конкатенация строк в Java означает, в целом, слияние, "склеивание" двух строк в одну:

```
class Test {
    public static void main(String[] args) {
                    String morning1 ="Morning";
                    String morning2 ="Evening";
                    System.out.println(" Good " + morning1);
                    System.out.println(" Good " + morning2);
        }}
```

Java

Что такое библиотеки классов Java?

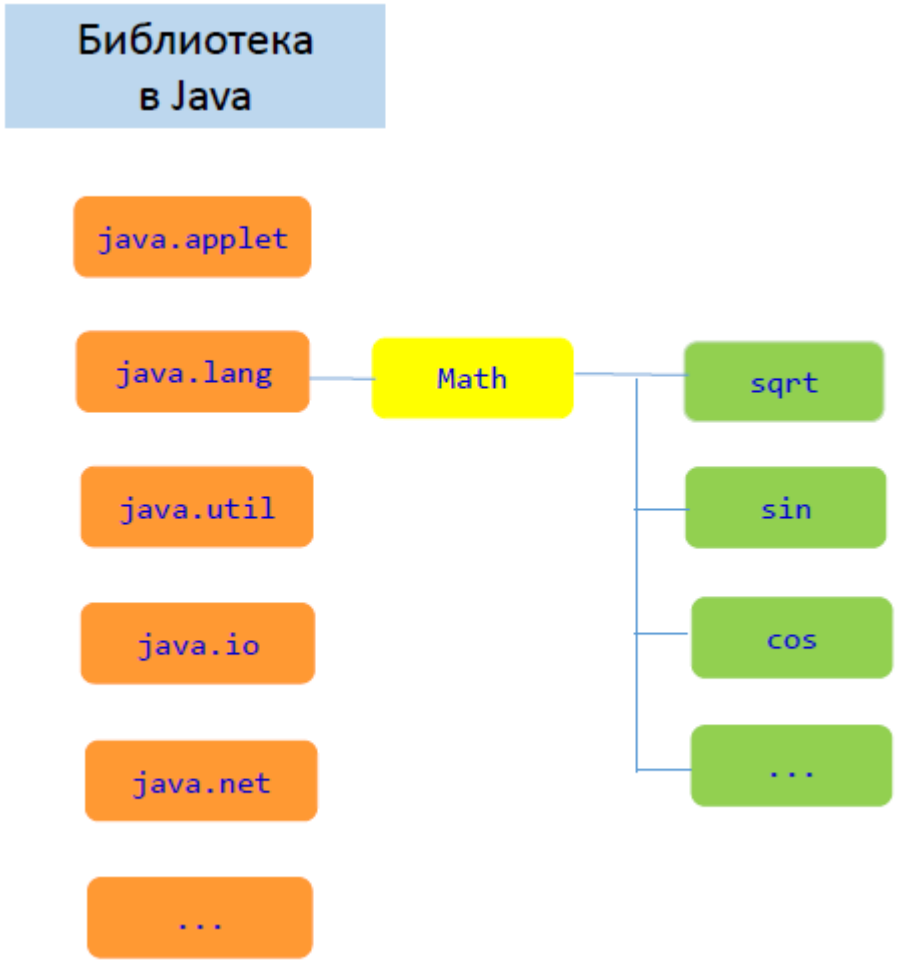
В Java есть виртуальная библиотека протестированного кода - это уже готовые решения ко многим задачам, которые стоят перед программистами в их ежедневной работе. То есть бери код из библиотеки и используй. А это очень и очень экономит время программиста, потому что не надо писать абсолютно весь код с нуля.

В этой виртуальной библиотеке Java информация разбита по пакетам ("packages") - это своеобразный аналог полочек в книжном магазине. В каждом пакете протестированный код по какому-то отдельно взятому направлению. Например, есть такие пакеты:

- java.applet
- java.lang - это основной пакет языка Java
- java.util
- java.io
- java.net

Java

Что такое библиотеки классов Java?



Java

Java Scanner Class

При написании программы часто необходимо выполнить такие задачи:

- Пользователь ввёл в консоли какое-то число. А программа должна считать с консоли, какое же число ввёл пользователь.
- Пользователь ввёл в консоли какое-то слово. А программа должна считать с консоли, какое же слово ввёл пользователь.

Для решения таких задач в Java используется сканер (от англ. Scanner). Если что-то ввели в консоли, а нам надо считать что же именно ввели - используем сканер.

4 основных метода сканера:

- next ();
- nextLine ();
- nextInt ();
- nextDouble ();

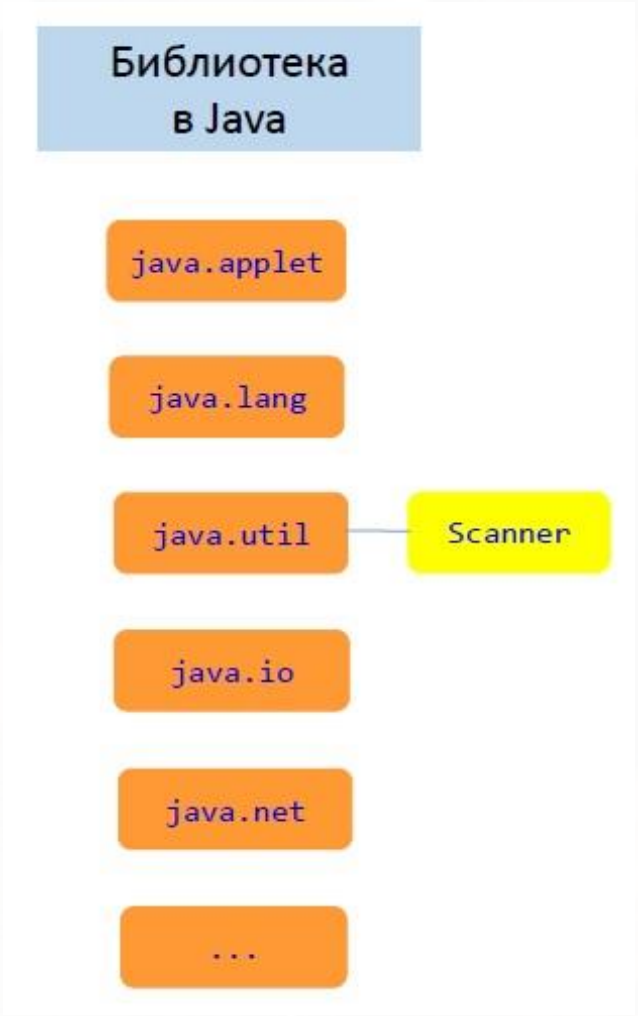
Java

Java Scanner Class

- **next()** может читать ввод только до пробела. Он не может прочитать два слова, разделенных пробелом. Кроме того, next() после прочтения ввода курсор помещается в ту же строку.
- **nextLine()** читает ввод, включая пробелы между словами (то есть читает до конца строки \n). После считывания ввода помещает nextLine() курсор в следующую строку. Для чтения всей строки можно также использовать nextLine() .
- **nextInt()** - метод считывает и возвращает введенное число.
- **nextDouble()** - метод отвечает за то, чтобы считать введенное пользователем дробное число с консоли.

Java

Java Scanner Class



Java

Java Scanner Class

Чтобы мы смогли использовать в коде класс Scanner, нам необходимо сделать следующие 4 шага:

1. Обязательно прописать вот такую строчку в коде:
   ```
   import java.util.Scanner; // импорт сканнера
   ```
2. Объявить сканера:
   ```
   Scanner scan = new Scanner(System.in);
   ```
3. Вызвать соответствующий метод для считывания с консоли:
   ```
   int number = scan.nextInt();
   ```
4. Закрыть сканер:
   ```
   scan.close();
   ```

Java

Задание.

- Запросите у пользователя ввести имя и, отдельно год рождения
- Посчитайте возраст и выведите сообщение на консоль, в котором будет его имя и подсчитанный возраст

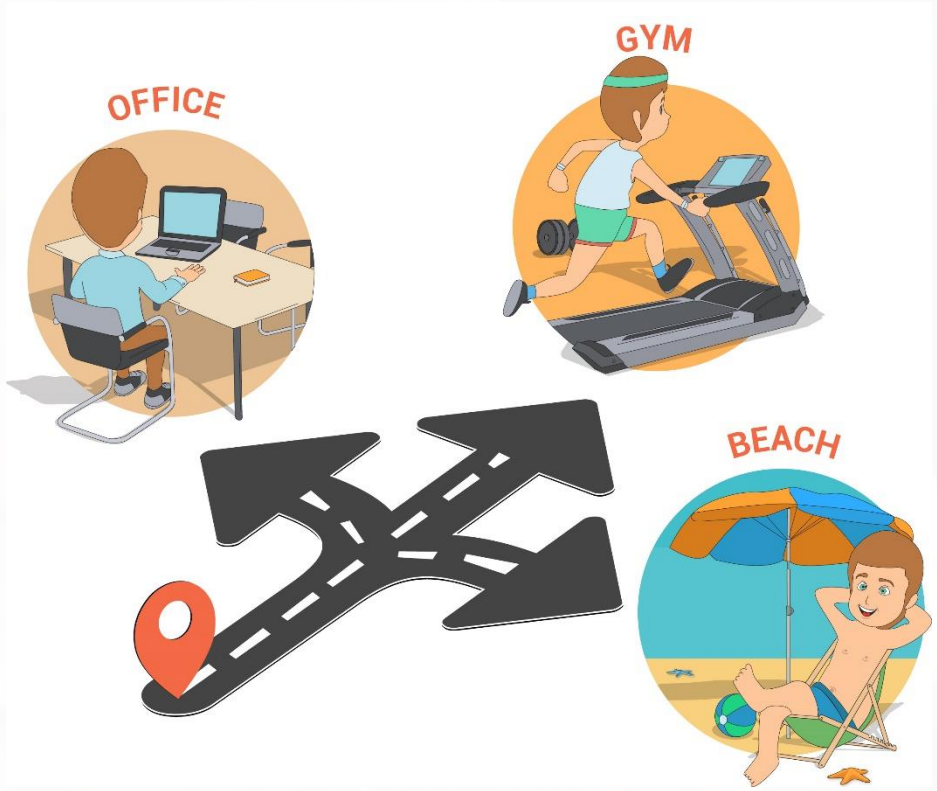
Правильно ли подсчитан возраст?

Java

Условные операторы в Java

В Java есть так называемые конструкции ветвления:

- Условный оператор if
- Оператор switch



Если налево – попадёшь в офис

Если прямо – попадёшь в спортзал

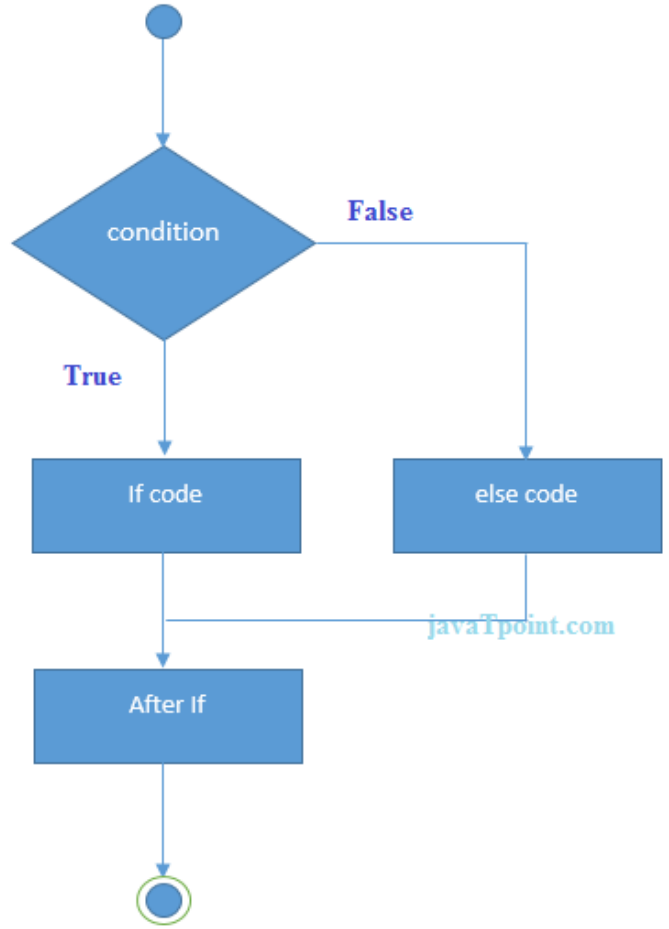
Если направо – попадёшь на пляж

Java

Условные операторы в Java

Общая форма условного оператора if в Java такая:

```
if (условие) {
//действие(-я), которые выполняются, если условие истинно;
}
else if (условие) {
//действие(-я), которые выполняются, если условие истинно;
}
else if (условие) {
//действие(-я), которые выполняются, если условие истинно;
}
else {
//действие(-я), которые выполняются, если условие истинно;
}
```



Java

Условные операторы в Java

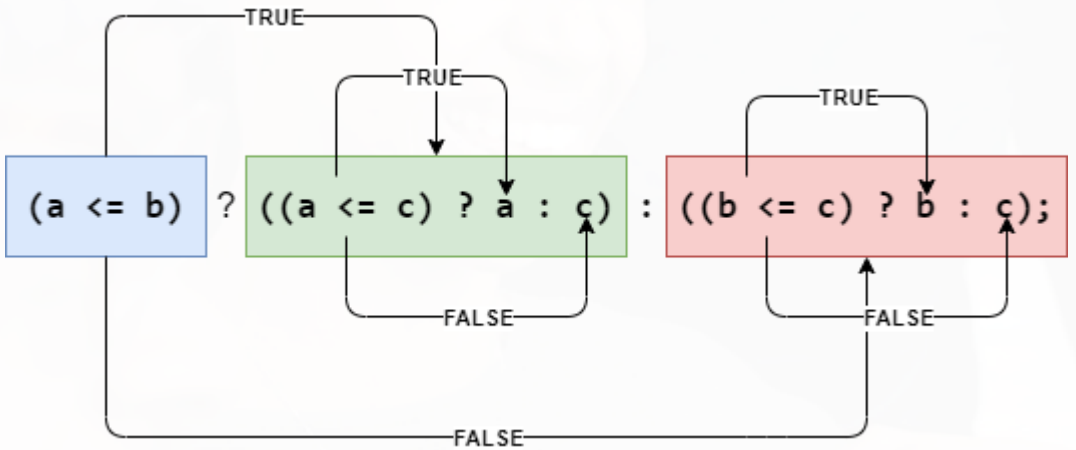
Обратите внимание, что:

- Оператор всегда начинается со слова **if**, за которым всегда идут круглые скобки с условием.
- После круглых скобок никогда не ставится точка с запятой
- Для того, чтобы указать альтернативный вариант (если не выполняется, тогда...) используется слово **else**.
- Если условий несколько, то каждое из них будет записываться через комбинацию **else if**, после которых в круглых скобках записывается альтернативное условие. Последний вариант (если не то, не то и не то, то...) записывается через **else** без условия.

Условные операторы в Java

Существует и укороченная форма (тернарный оператор):

условие …….. ? ……. : ……



Java

Задание.

- Необходимо написать программу, где бы пользователю предлагалось ввести число 1. Если пользователь ввёл число 1, программа должна вывести сообщение: "Вы ввели число 1". Если пользователь ввёл другое число, программа должна вывести такое сообщение: "Вы ввели число не равное 1".
- Необходимо написать программу, где бы пользователю предлагалось ввести число 1. Если пользователь ввёл число 1, программа должна вывести сообщение: "Вы ввели число 1". Если пользователь ввёл другое число, программа **вообще ничего не должна делать.**
- Необходимо написать программу, где бы пользователю предлагалось ввести любое число. Программа выполнит проверку на чётность.

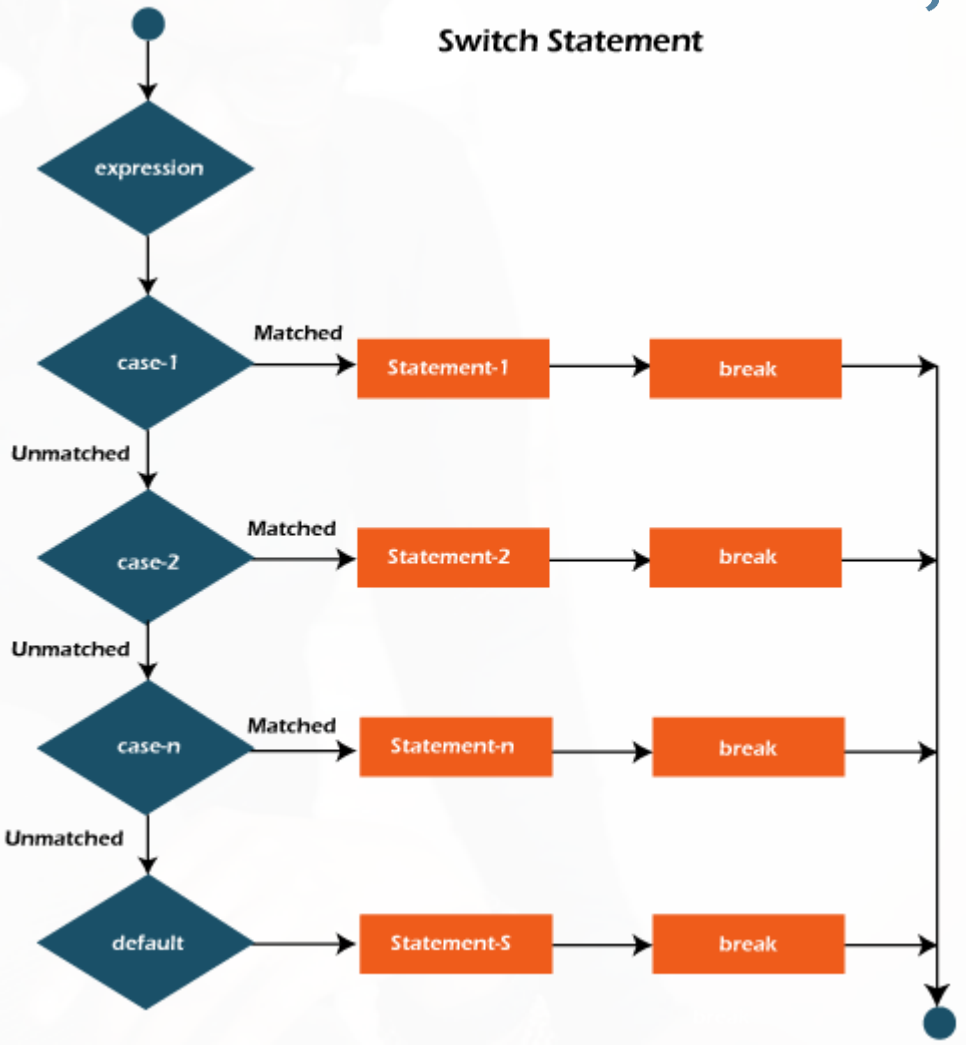
  Если чётное выведите: “Число " + number + " чётное."

  Если нет то выведите: “Число " + number + " нечётное."

Java

Условные операторы в Java

Конструкции с операторами **if else**, предлагающими большое количество условных ветвлений, могут выглядеть очень громоздкими. Поэтому в тех случаях, когда необходимо повторять проверку одного и того же значения переменной, есть более элегантное решение с помощью оператора switch.



Thanks for your time ☺