

Лекция 1

Вводная лекция, обучение JAVA.

Логическая задача №1

$$8809 = 6$$

$$7111 = 0$$

$$2172 = 0$$

$$6666 = 4$$

$$1111 = 0$$

$$3213 = 0$$

$$7662 = 2$$

$$9313 = 1$$

$$0000 = 4$$

$$2222 = 0$$

$$3333 = 0$$

$$5555 = 0$$

$$8193 = 3$$

$$8096 = 5$$

$$1012 = 1$$

$$7777 = 0$$

$$9999 = 4$$

$$7756 = 1$$

$$6855 = 3$$

$$9881 = 5$$

$$5531 = 0$$

$$2581 = ???$$

Логическая задача №2

- Пяти машинам текстильной фабрики требуется пять минут, чтобы изготовить пять вещей. За сколько минут 100 машин изготовят 100 вещей?

Логическая задача №3

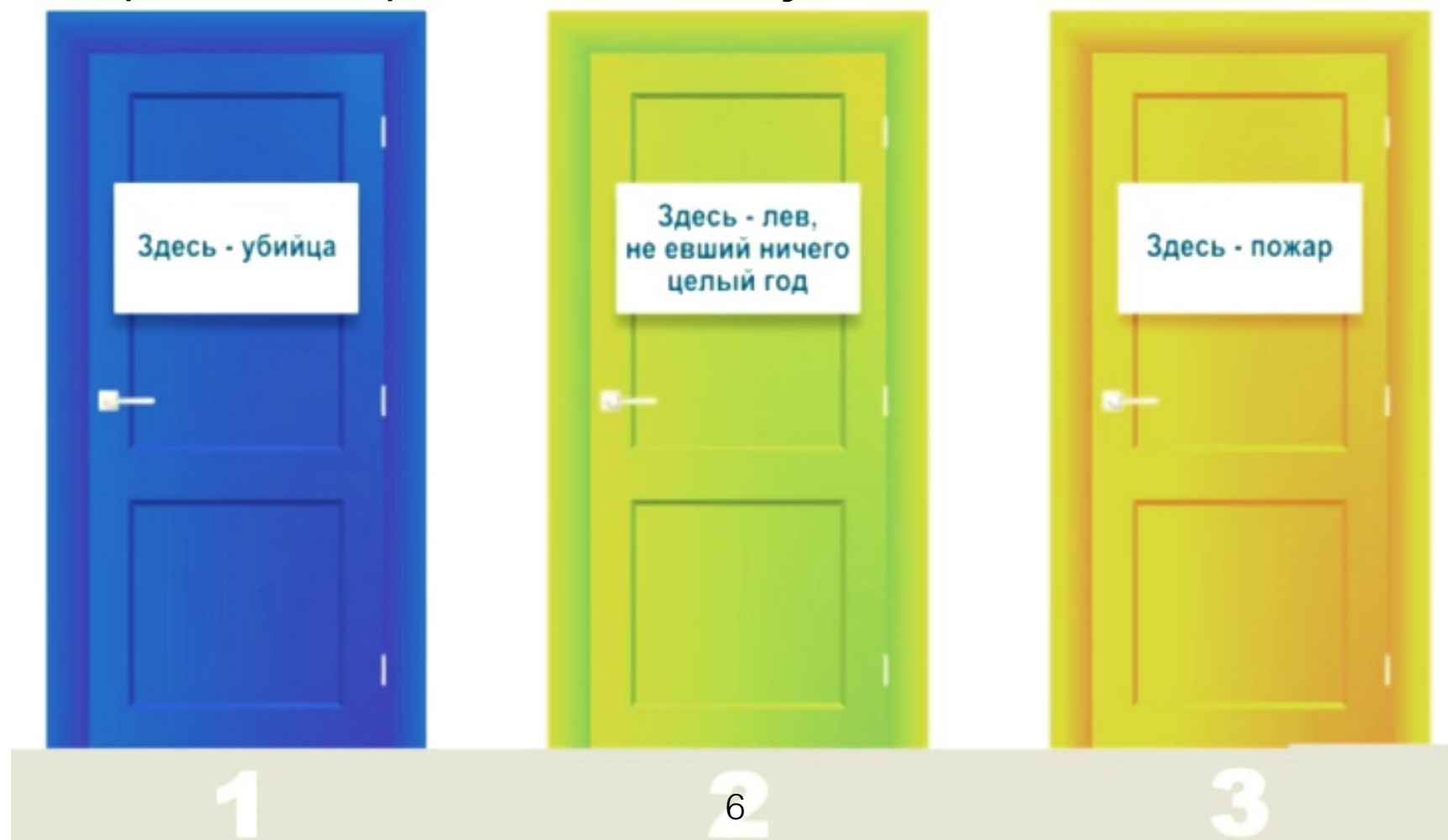
- Новая экспериментальная производственная линия тестируется перед запуском на заводе. Линия выпускает автомобильные двигатели. В ходе тестирования выпуск продукции на линии удваивался ежедневно, и задача по выпуску продукции была выполнена за 18 дней. Сколько дней занял выпуск 25% этой продукции?

Логическая задача №4

- Есть два города А и Б, расстояние между ними по железной дороге 100 км. Одновременно из каждого из них навстречу друг другу выехало два поезда (Па и Пб), один со скоростью 40 км в час, другой 60. Одновременно с этим из пункта А в навстречу поезду Пб вылетела муха, достигнув поезда, она резко развернулась и полетела обратно до поезда Па, достигнул его, снова развернулась и так до тех пор пока муху не раздавило столкновением поездов. Вопрос, какое расстояние пролетела муха, если ее скорость была постоянной и равнялась 80 км в час?

Логическая задача №5

- Перед вами имеется три двери. За каждой из них вас ждёт определённая опасность. За первой дверью – убийца, который так и ждёт, когда придёт его жертва. За второй дверью – свирепый лев, которого не кормили целый год. За третьей – полыхает пожар. Вам обязательно нужно войти в одну из дверей. Какую дверь вы откроете и почему?



Первые строки кода

- Программа — это набор (список) команд. Команды выполняются по очереди: слева-направо, сверху-вниз. Сначала исполняется первая команда, затем вторая, третья, и так далее. Когда все команды исполнены, программа завершается.
- В языке программирования Java каждую команду принято писать с новой строки. В конце команды ставится точка с запятой.

```
System.out.println(" Наша первая лекция ");
```

- Программы на языке Java состоят из классов. Классов может быть десятки тысяч. Но минимальная программа — это один класс. Для каждого класса заводится отдельный файл, имя которого совпадает с именем класса.
- Программа не может состоять только из команд.
- Например, представьте себе комнату. Комната не может быть сама по себе, потому что комната — часть какой-то квартиры. Квартира тоже не может существовать сама по себе, она находится в каком-то доме. С другой стороны можно сказать, что дом делится на квартиры, а квартиры делятся на комнаты.
- Так вот, команда — это как комната. В языке программирования Java команда не может быть сама по себе, она — часть функции (в Java функции еще называют методами). А метод — это часть класса. Или, другими словами, класс делится на методы, а методы на команды.
- Т.е. класс — это многоквартирный дом, функция/метод — это квартира, а команда — это комната.

- Допустим, вы решили создать класс, который будет описывать дом (дом по-английски – house/home). Тогда вам нужно создать класс Home, который будет содержаться в файле Home.java.

Первый класс

- Внутри файлов содержится код (текст) на языке программирования Java. Обычно код класса состоит из «имени класса» и «тела класса». Тело класса помещается в фигурные скобки. Вот как может выглядеть класс Home (файл Home.java):

```
public class Home  
{
```

ТЕЛО КЛАССА

```
}
```

- Тело класса может содержать переменные (их еще называют данные класса) и методы (функции класса).

```
public class Home
```

```
{
```

ПЕРЕМЕННАЯ A

ПЕРЕМЕННАЯ Z

МЕТОД 1

МЕТОД N

```
}
```

- Пример:


```
public class Home
{
    int a;
    int b;

    public static void main(String[] args)
    {
        System.out.print("1");
    }

    public static double pi()
    {
        return 3.14;
    }
}
```

«int a» и «int b» — это переменные, а «main» и «pi» — это методы

- Минимальная программа должна состоять минимум из одного класса, который должен содержать минимум один метод/функцию, с которого начинается выполнение программы. Такой метод должен иметь имя `main`.
- Класс, с которого начинается программа, может иметь любое имя, но метод `main`, с которого начинает выполняться программа, всегда имеет один и тот же вид:

```
public class Home
{
    //неизменяемая часть
    public static void main (String[] args)
    {
        
    }
}
```

- Тело метода состоит из команд. Можно даже сказать, что метод — это команды, объединенные в группу, которой дали имя (имя метода). И так и так будет верно.
- В языке Java есть команды на все случаи жизни. Каждая команда описывает какое-то определенное действие. В конце каждой команды ставится **точка с запятой**.

System.out.println

	Команда	Описание, что делает
1	<code>System.out.println(1);</code>	Выводит на экран число 1
2	<code>System.out.println(" Сириус.ИС ");</code>	Выводит на экран надпись Сириус.ИС
3	<code>System.out.println(" Первая лекция ");</code>	Выводит на экран надпись Первая лекция

- Если вы хотите вывести на экран текст, его нужно с двух сторон обозначить двойными кавычками «"». Одинарная кавычка выглядит вот так «'», а двойная — вот так «"». Двойная кавычка — это не две одинарных, просьба не путать.
- Есть две вариации команды: `System.out.println()` и `System.out.print()`.
- Команда `println` не выводит текст с новой строки — она выводит текст на текущей строке, но делает так, чтобы следующий текст выводился на новой строке.

Готовая программа

```
public class Home
{
    public static void main(String[] args)
    {
        System.out.print("Имя ");
        System.out.print("Фамилия ");
        System.out.print("Группа");
    }
}
```

Изучение командной строки

- Основные команды для работы в терминале:

В какой папке находитесь: `pwd`

Содержимое текущего каталога: `ls`

Подробное содержание текущего каталога: `ls -l`

Копировать файлы: `scp "что" "куда"`

Создание каталога: `mkdir "название"`

Переход в каталог: `cd "куда"`

Создать новый файл: `touch "имя файла с расширением"` (`touch test.txt`)

Изменение прав: `chmod XXX name`

Удалить папку со всем содержимым: `rm -rf "Название папки"`

Исполняемые процессы: `ps`

Прерывание: `ctrl+C`

Скомпилировать: `javac name.java`

Запуск: `java name`

Итоговые задания:

1.
 - а. Узнать, в какой папке находитесь.
 - б. Добавить новую папку с вашим именем.
 - в. Создать в ней несколько файлов с разным расширением.
 - г. Вывести подробное содержание папки.
 - д. Поменять права всех файлов в вашей папке на 664.
 - е. Вернуться на одну папку выше.
 - ж. Удалить папку с вашим именем.
2. Напишите в блокноте программу, которая выводит на экран ваше Имя, Фамилию и номер группы. Скомпилируйте и выполните вашу программу.

Основы JAVA. Вывод на экран, типы String и int.

Переменные

- Переменные – это специальные штуки для хранения данных.
- Значение – это некий объект, данные или информация, которая хранится в переменной.
- У переменной тоже есть свой тип. Переменная может хранить значения только того же типа, что и она сама.
- Чтобы создать переменную используется команда вида: «тип имя».

- Два самых часто используемых типа — это целые числа (обозначается словом `int`) и текст (обозначается словом `String`).

	Создание переменной: сначала тип, затем имя.	Описание
1	<code>int a;</code>	Создаем переменную по имени <code>a</code> типа <code>int</code> .
2	<code>String s;</code>	Создаем переменную по имени <code>s</code> типа <code>String</code> .
3	<code>double c;</code>	Создаем переменную по имени <code>c</code> типа <code>double</code> .

- При объявлении переменных в них сразу можно заносить их значения.
- Для того, чтобы занести новое значение в переменную, используется знак “=”. Его ещё называют **«оператором присваивания»**. Присваивание – это занесение в переменную значения, взятого из другой переменной или вычисленного на основе нескольких переменных.
- При вычислении нового значения переменной может использоваться её старое значение.

	Код	Описание
1	<code>i = 3;</code>	В переменную <code>i</code> заносится значение 3.
2	<code>a = 1;</code> <code>b = a + 1;</code>	В переменную <code>a</code> заносится значение 1. В переменную <code>b</code> заносится значение 2.
3	<code>x = 3;</code> <code>x = x + 1;</code>	В переменную <code>x</code> заносится значение 3. В следующей строчке значение <code>x</code> увеличивается на 1, <code>x</code> теперь равен 4

- Для занесения значения в переменную существует специальная операция – **операция присваивания**. Она копирует значение из одной переменной в другую. Не переносит, а именно копирует.
- Для операции присваивания используется **символ равно «=»**.
- **Это не сравнение**. Это именно копирование значения справа от знака равно в переменную, которая слева. Для сравнения в языке Java используется двойное равно «==».

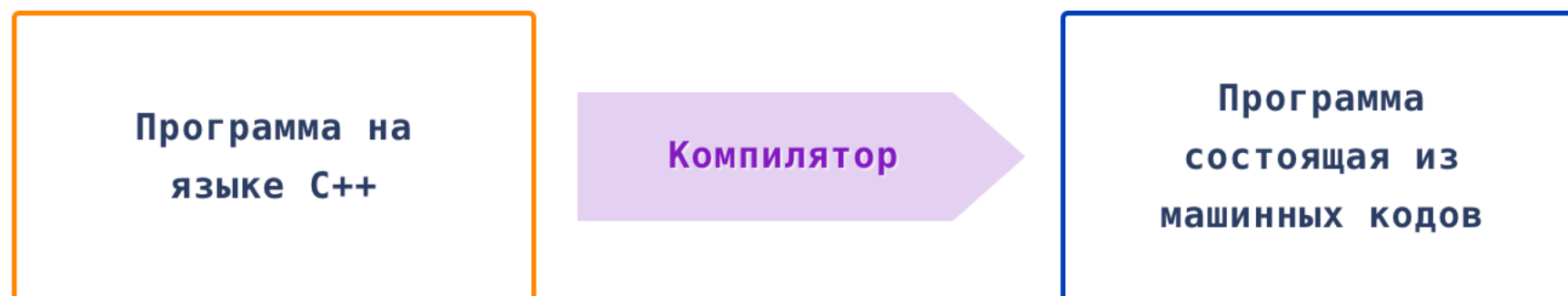
Задачи

3. Написать программу, выводящую на экран надпись «**Изучаем java**» **5 раз**. Каждый раз - с новой строки.
4. Напишите программу, которая в методе `main` объявляет такие переменные: `name` типа `String`, `age` типа `int` и `city` типа `String`. Примечание: "объявить переменную" - значит то же, что и "создать переменную".

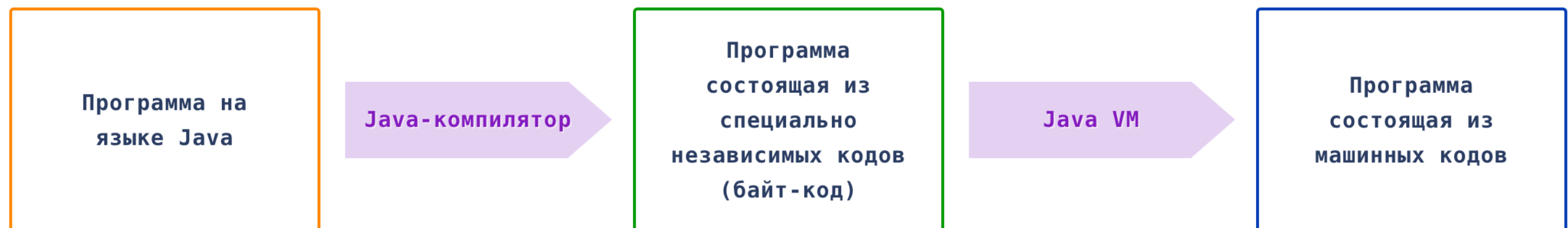
Компиляторы

- Дело в том, что компьютер умеет исполнять только простейшие числовые команды.
- Писать программу в виде чисел очень сложно, поэтому люди придумали языки программирования и компиляторы. Такой язык с одной стороны понятен человеку, с другой — компилятору. Компилятор — это специальная программа, которая переводит текст программы, написанный на языке программирования, в набор машинных кодов.
- Обычно программист пишет программу на языке программирования, а затем запускает компилятор, который на основе написанных программистом файлов с кодом программы делает один файл с машинным кодом — окончательную (скомпилированную) программу.

- Получившаяся в итоге программа сразу может выполняться на компьютере. Минусом такого подхода есть то, что код полученной программы сильно зависит от процессора и операционной системы. Программа, скомпилированная под Windows, не будет работать на телефоне с Android.



- Но у Java гораздо более инновационный подход.
- Компилятор Java не компилирует все классы в одну программу из машинных кодов. Вместо этого он компилирует каждый класс по отдельности и не в машинные коды, а в специальный промежуточный код (байт-код).
Компиляция в машинный код выполняется при запуске программы.
- Есть специальная программа под названием JVM (Java Virtual Machine) – Виртуальная Машина Java. Именно ее запускают первой, когда нужно запустить программу, состоящую из байт-кода. А уже JVM перед выполнением нужной программы компилирует ту в машинный код.
- Благодаря такому подходу программы, написанные на Java, могут выполняться практически на любом устройстве.



Комментарии

- В языке Java вы можете писать не только команды, но и комментарии к ним прямо в коде. Такие комментарии игнорируются компилятором, будто и нет их вовсе. При исполнении программы все комментарии пропускаются!
- В коде класса мы написали комментарий «Сейчас мы выведем на экран фразу ...». Начало комментария обозначается парой символов «/*», а конец — «*/». Когда программа будет компилироваться, компилятор пропустит все, что находится между символами /* и */

```
public class Home
{
    public static void main (String[] args)
    {
        /*
        Сейчас мы выведем на экран фразу 'SIRIUS IS'
        */
        System.out.print("SIRIUS IS");
    }
}
```

- Есть еще один способ задать комментарий в коде — с помощью символов «//».
- При этом **комментарием считается часть кода, начиная с пары символов // и до конца строки**, где они расположены. Т.е. второй пары символов, которые «закрывают комментарий» нет.

```
public class Home
{
    public static void main (String[] args)
    {
        System.out.print("SIRIUS IS"); //вот тут еще один комментарий
    }
}
```

Повторение

- Программа на Java состоит из классов. Каждый класс хранится в отдельном файле. Имя файла совпадает с именем класса, расширение файла – java.
- Программа состоит из набора файлов с расширением java, и в каждом файле написан код одного класса.
- Если файл называется MyCat.java, то он содержит класс MyCat.

- Когда у нас много файлов с классами, мы группируем их в папки и подпапки. Классы при этом группируются по пакетам и подпакетам. Имена пакетов и подпакетов нужно указывать в коде класса, и они должны совпадать с именами папок и подпапок на диске. Т.е. с одной стороны у нас есть файлы, разложенные по папкам, а с другой — классы, разложенные по пакетам. При этом имя класса обязано совпадать с именем файла, в котором этот класс описан. А имя пакета совпадает с именем папки, в которой хранится класс.

Основы JAVA. Примитивные переменные.

Примитивные типы данных

- У каждой переменной есть область памяти, привязанная к ней, где эта переменная хранит своё значение.
- Все сложные типы состоят из более простых. Те, в свою очередь, из ещё более простых. Пока, наконец, дело не доходит до самых **примитивных**, неделимых типов. Их так и называют — **примитивные типы**. Например, **int** — это один из примитивных типов, а **String** — это уже сложный тип, хранящий свои данные в виде таблицы символов (где каждый символ — это примитивный тип — **char**).
- Сложные типы образуются из простых путём группировки. Такие типы мы называем **классами**. Когда мы описываем в программе новый класс — это значит, что мы объявляем новый **сложный составной тип**, данные которого будут или другими сложными типами, или примитивными типами.

	Код на Java	Описание
1	<pre> public class Person { String name; int age; } </pre>	<p>Объявили новый сложный тип — Person.</p> <p>Его данные — это переменная name типа String (сложный тип) и переменная age типа int (примитивный тип)</p>
2	<pre> public class Rectangle { int x, y, width, height; } </pre>	<p>Объявили новый сложный тип — Rectangle.</p> <p>Он состоит из четырёх переменных примитивного типа — int.</p>
3	<pre> public class Cat { Person owner; Rectangle territory; int age; String name; } </pre>	<p>Объявили новый сложный тип — Cat.</p> <p>У него есть переменные:</p> <ul style="list-style-type: none"> — owner, сложный тип Person — territory, сложный тип Rectangle — age, примитивный тип int — name, сложный тип String

- Т.к. большие (сложные) типы содержат в себе много маленьких (примитивных), то их объекты занимают много памяти. Больше, чем обычные переменные примитивных типов. Иногда намного больше. Присваивание таких переменных выполнялось очень долго и требовало копирования больших объёмов памяти. Поэтому **переменные сложных типов хранят в себе не сам объект, а всего лишь ссылку на него!** Т.е. четырёхбайтовый адрес. Этого хватает, чтобы можно было обращаться к данным этих объектов. Всю сложность, связанную с этим, берет на себя Java-машина.

Пример

- Мы уже говорили, что переменная — это как коробка. Если вы хотите сохранить в ней число 13, то вы можете написать его на листе и положить в коробку.
- Но представьте, что вам надо сохранить в коробку (переменную) что-нибудь побольше. Например, собаку, машину или твоего соседа Васю. Чтобы не пихать в коробку невпихиваемое, можно поступить проще: вместо собаки взять ее фото, вместо машины — ее номер, вместо Васи — его номер телефона.
- Вот мы берем лист бумаги и пишем на нем телефонный номер Васи. Это и будет аналогом ссылки на объект. Если мы достанем из коробки лист с номером Васи, отксерим его и положим в несколько коробок, то количество ссылок на Васю увеличится, но Вася как был один, так и остался.
- Особенность такого хранения данных в том, что **ссылок может быть много, а объект — один.**

- Если в переменной ссылочного (сложного) типа ещё нет ссылки на какой-то объект, то она хранит null – специальную «пустую ссылку». На самом деле, она просто хранит адрес объекта равный 0. Но Java-машина никогда не создаёт объекты с таким адресом, и поэтому всегда знает, что если переменная-ссылка содержит 0, то никакого объекта там нет.
- Переменные делятся на два типа: примитивные и ссылочные. Примитивные типы у себя внутри хранят значение, а ссылочные – ссылку на объект. Примитивные типы – это int, char, boolean и ещё немного, а ссылочные типы – это все остальные, и образуются они с помощью классов.

	Код на Java	Описание
1	String s; String s = null ;	Эквивалентные записи.
2	Person person; person = new Person(); person = null ;	Создали переменную person, её значение null. Занесли в неё адрес новосозданного объекта. Присвоили переменной ссылку null.
3	Cat cat = new Cat(); cat.owner = new Person(); cat.owner.name = "God";	Создали объект Cat, занесли его ссылку в переменную cat. cat.owner равен null. Занесли в cat.owner ссылку на новосозданный объект Person. cat.owner.name пока ещё null. cat.owner.name присвоили имя – God.