Lesson 1

Блокнот:	EpamJava0		
Создана:	11/09/2019 2:58 PM	**Обновлено:**	17/09/2019 3:27 PM
Автор:	n.masuhranov@gmail.com		

1. Объясните, что имеется в виду, когда говорится: Java - язык программирования и Java - платформа.

Java как язык программирования -
это высокоуровневый и объектно-ориентированным языком программирования.
Java как платформа - это программное обеспечение, представляющее собой рабочую среду для работы программ. Она состоит из Java API и JVM.

---

2. Поясните, как связаны имя java-файла и классы, которые в этом файле объявляются.

Имя файла и класса должны иметь одинаковое имя.
Имена классов должны быть существительными. В смешанном регистре первой заглавной буквой.

---

3. Расшифруйте аббревиатуры JVM, JDK и JRE; покажите, где "они находятся" и что собой представляют.

Java Development Kit, сокращенно JDK – это пакет, включающий в себя компилятор Java (javac), стандартные библиотеки классов Java, , документацию, различные утилиты и исполнительную систему Java (JRE).

Java Runtime Environment, сокращенно JRE – это исполнительная среда Java в которой выполняются программы, написанные на этом языке. Среда состоит из виртуальной машины – Java Virtual Machine(JVM) и библиотеки Java классов. JRE является частью JDK.

Java Virtual Machine, сокращенно JVM – это виртуальная машина Java — основная часть исполняющей среды JRE. Виртуальная машина Java посредством компилятора (javac) компилирует написанный код в байт-код и исполняет его.

---

4. Объясните, как скомпилировать и запустить приложение из командной строи, а также зачем в переменных среды окружения прописывать пути к

установленному JDK.

В командной строке прописать название нашего файла для компилятора (javac HelloWorld.java), после будет создан файл HelloWorld.class который является скомпилированным байт-кодом программы. Затем мы запускаем программу с помощью нашей виртуальной машины (java HelloWorld).

Переменная PATH - это системная переменная которую операционная система использует для того чтобы найти нужные исполняемые объекты в командной строке. Если не прописать путь, то наша система не найдёт компилятор.

---

5. Перечислите атрибуты доступа и объясните их действие.

- Модификатор public. К переменной, методу или классу, помеченному модификатором public, можно обращаться из любого места программы, никаких ограничений нет.
- Модификатор private. К переменной, методу или классу, помеченному модификатором private, можно обращаться только из того же класса, где он объявлен. Для всех остальных классов помеченный метод или переменная – невидимы. Только свой класс! Такие методы не наследуются и не переопределяются. Доступ к ним из класса-наследника также невозможен.
- Модификатор по умолчанию (default или package-private).Если переменная или метод не помечены никаким модификатором, то считается, что они помечены «модификатором по умолчанию». Переменные и методы с таким модификатором видны всем классам пакета, в котором они объявлены, и только им.
- Модификатор protected. К переменной, методу или классу, помеченному модификатором protected, можно обращаться из его же пакета (как package), но еще из всех классов, унаследованных от текущего.

---

6. Что такое пакеты в java-программе, что представляют собой пакеты на диске? Каково соглашение по именованию пакетов? Как создать пакет?

Пакеты в Java это механизм позволяющий организовать классы в пространстве имён. Имена и расположение пакетов соответствуют их каталогам в файловом менеджере. Всё это сделано для того, чтобы не возникал конфликт имён в случае если имеются несколько классов с одинаковыми названиями, но в разных пакетах.

Соглашение по пакетам. Имена пакетов пишутся в нижнем регистре (позволяет избежать конфликта имён с классами или интерфейсами). Компании используют инвертированное имя своего доменного адреса чтобы начать их имена пакетов (com.epam.first_lesson).

В некоторых случаях имя домена может являться не допустимым, в таком случае следует использовать подчёркивание(вместо дефиса или спец.символа, если имя пакета начинается с цифры, содержит зарезервированное имя).

---

7. Объясните, какие классы, интерфейсы, перечисления необходимо импортировать в вашу программу, как это сделать. Влияет ли импорт пакета на импорт классов и др., лежащего в под пакетах? Какой пакет в Java импортируется по умолчанию?

Для импорта классов, перечислений или интерфейсов, необходимо прописать путь через пакеты где лежит данный файл проекта (import com.epam.first_lesson.HelloWorld;) для конкретного класса, или же поставив знак * импортировать все классы данного пакета, а так же (если имеются) и подпакетов. По умолчанию в java импортируется пакет java.lang.

---

1. Объясните различия между терминами "объект" и "ссылка на объект".

Сам объект хранится в памяти и доступ к ним мы имеем только с помощью ссылочных переменных.
Dog dog = new Dog();
Dog dog2 = dog;

Мы имеем один единственный объект в памяти, но получаем к нему доступ с помощью двух ссылочных переменных. Иными словами, если в примитивах int a = 2; int b = a; b = 3;
мы изменим значение одной переменной, то изменив параметр в ссылочной переменной мы изменим сам объект. Ссылка на объект это лишь адрес с доступом к самому объекту.

---

9. Какие примитивные типы Java вы знаете, как создать переменные примитивных типов? Объясните процедуру, по которой переменные примитивных типов передаются в методы как параметры.

byte, short, char, int, long, float, double, boolean.
Для создания переменной требуется проинициализировать переменную, а затем указать её корректное значение.
int a; - инициализация
a = 3; - значение.
int a = 3; более короткая форма.

Переменная - это контейнер со значением. При передаче переменных в метод, переменные передаются по значению (это означает скопировать значение и передать копию не меняя исходного).

---

10. Каков размер примитивных типов, как размер примитивных типов зависит от разрядности платформы, что такое преобразование (приведение) типов и зачем оно необходимо? Какие примитивные типы не приводятся ни к какому другому типу.

byte - 8 бит
short - 16 бит
char - 16 бит
int - 32 бита
long - 64 бита
float - 32 бита
double - 64 бита
boolean (зависит от JVM)

От разрядности платформы зависит только тип boolean, все остальные примитивы размер никак не меняют.
Преобразование типов = изменению типа (из byte в int). Нужно это потому что каждый тип занимает место в памяти и это может накладывать ограничения на операции.
Тип boolean не приводится ни к одному другому.

---

11. Объясните, что такое явное и неявное приведение типов, приведите примеры когда такое преобразование имеет место.

Не явное приведение типов - это когда мы из переменной большего размера присвоить значение меньшей, компилятор это спокойно допускает т.к. из меньшей чашки кофе мы переливаем содержимое в большую.
byte a = 15;
int b = a;

Явное приведение типов - это когда из переменной большего размера присваиваем значение переменной меньшего. Здесь мы явно даём компилятору понять что переливая содержимое из большей чашки в меньшую берём всё на себя.

int a = 4;
byte b = (byte) a;

Это имеет место быть когда в одной операции вовлечены несколько типов переменных.

Допустим воспользовавшись стандартной библиотекой Math мы хотим возвести число 6 (byte) в квадрат. Тут произойдёт неявное преобразование в тип double.

---

12. Что такое литералы в Java- программу, какую классификацию литералов вы знаете, как записываются литералы различных видов и типов в Java-программе.

Литералы - это явно заданные значения в коде программы.
Литералы бывают : числовые (целочисленные, с плавающей точкой), строковые, символьные, логические.

Целочисленные:

- Двоичная int a = 0b1101010110; (854)
- Восьмеричная int b = 012314; (5324)
- Десятичная int c = 456; (456)
- Шестнадцатеричная int d = 0x141D12; (1318162)

С плавающей точкой:

- В качестве десятичной дроби double pi = 3.14;
- В научном виде double b = 4.05E-13 (4.05 * 10^-13)

В случае с типом float необходимо добивать в конце символ f либо F.

Строковые литералы :
Набор символов заключённый в двойные кавычки. Так же могут находиться служебные символы.
String str = "Hello, World!";

Символьные литералы :
Представлены кодовой таблицей Unicode. В коде его записывается в одинарных кавычках.
Может быть записан просто как символ
char a = 'a';
А может указываться в 16-битовом виде
char b = '\u00F7';
А так же в восьмеричном
chat c = '\122';
+ служебные символы (escape последовательности : \t, \n и т.д.)

Логические :
Имеет всего два значения true и false, указываются явно, без символов.

---

13. Как осуществляется работа с типами при вычислении арифметических выражений Java.

При арифметических выражение меньший тип всегда приводится к большему, а в случае целочисленного с дробным, целочисленный приводится к дробному. При делении целочисленных типов, ответом будет целое число, без дробной части.

---

14. Что такое классы-оболочки, для чего они предназначены? Объясните, что значит: объект класса оболочки - константный объект.

Классы-оболочки являются объектным представлением примитива. Они предназначены для расширения возможностей, например перевести строку в число
String a = "11";
int b = a;
Выдаст ошибку
int b = Integer.parseInt(a);
Приведёт строку "11" к примитиву типа int со значением 11.

Это значит что этот объект является неизменяемым. К примеру :
Integer a = 11; (= new Integer(11))
a = 12;
Если вывести на консоль, то выйдет число 12, но объект не был изменён, а был создан новый, со значением .

---

15. Объясните разницу между примитивными и ссылочными типами данных. Поясните существующие различия, при передаче параметров примитивных и ссылочных типов в методы. Объясните, как константные объекты ведут себя при передаче в метод.

Примитивный тип данных хранит значение, ссылочный хранит адрес к объекту.
int a = 1;
int b = a;
b = 2;
Т.к. примитивы хранят значения, то b скопировало значение из a, а затем изменило своё значение никак не влияя на a.

Dog dog = new Dog(12);
Dog dog2 = dog;
dog2.setLength(15);
В данном примере обе переменные являются сслыками на один и тот же объект, присвоив адрес объекта второй переменной, она получила доступ к объекту и изменив длину собаки, она изменила параметр этого объекта, а следовательно dog.getLength(), выдаст 15.

При передаче параметра в метод, независимо от того примитив это или ссылка, создаётся копия, в новое значение. Иными словами передав переменную int a = 3 в метод
static int fact(int c){}, переменная с скопирует значение а не меняя её параметры. В случае с объектами передавая ссылку на объект в метод, он так же копирует значение, но значение данной переменной является сслыка на объект. Иными словами появляется новая ссылка на объект, а следовательно и свойства объекта могут быть изменены в методе.

Если создать объекты
```
final Dog dog = new Dog(12);
Dog dog2 = dog;
dog = length(dog);
```

а затем поместить в метод
```
static Dog length(Dog ex){
    ex.setLength(15);
    return ex;
}
```

то будет ошибка т.к. мы пытаемся работать с final переменной.

однако если мы используем не final переменную ссылающуюся на тот же объект
dog2 = length(dog);
то объект будет изменён

---

16. Поясните, что такое автоупаковка и автораспаковка.

Автоупаковка - это функция преобразования примитивных типов в эквивалентные объекты.
Intrger integer = 9;

Автораспаковка - обратный процесс упаковки, т.е. преобразование объектов в соответствующие им примитивные типы.
int in = 0;
in = new Integer(9);

---

17. Перечислите известные вам арифметические, логические и битовые операторы, определите случаи их употребления. Что такое приоритет оператора, как определить в какой последовательности будут выполняться операции в выражении, если несколько из них имеют одинаковый приоритет.

Арифметические:
+; -; *; /; %; ++; --;

Логические :
&&; ||; !

Побитовые :

&; |; ^; ~; <<; >>; >>>;

Приоритеты

1. (), [], .
2. ++, --, ~, !
3. *, /, %
4. +, -
5. >>, >>>, <<
6. >, >=, <, <=
7. ==, !=
8. &
9. ^
10. |
11. &&
12. ||
13. ?:
14. =, +=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=
15. ,

При одинаковом приоритете операции выполняются в порядке записи.

---

18. Укажите правила выполнения операций с плавающей точкой в Java (согласно стандарту IEEE754). Как определить, что результатом вычисления стала бесконечность или не число.

Стандарт IEEE 754 определяет пять правил округления.

Округление к ближайшему :

- Округление к ближайшему (привязка к чётному). Если два ближайших числа с плавающей точкой одинаково близки, то должно быть получено число с чётной самой младшей цифрой.
- Округление к ближайшему (привязка к бесконечности). Если два ближайших числа с плавающей точкой одинаково близки, то должно быть получено число с большим модулем.

Направленные округления :

- Округление к 0 - направленное округление к нулю (также известное как усечение).

- Округление к + ∞ - направленное округление к положительной бесконечности (также известное как округление вверх или потолок).
- Округление к - ∞ - направленное округление к отрицательной бесконечности (также известное как округление вниз или пол).

Положительное число делённое на ноль даёт плюс бесконечность, отрициательное делённое на ноль даёт минус бесконечность.

К операциям, приводящим к появлению NaN в качестве ответа, относятся:

- все математические операции, содержащие NaN в качестве одного из операндов;
- деление нуля на нуль;
- деление бесконечности на бесконечность;
- умножение нуля на бесконечность;
- сложение бесконечности с бесконечностью противоположного знака;
- вычисление квадратного корня отрицательного числа;
- логарифмирование отрицательного числа.

---

19. Что такое статический импорт, какие элементы можно импортировать при статическом импорте.

Для того чтобы получить доступ к статическим членам классов, требуется указать ссылку на класс. Благодаря статическому импорту появляется возможность ссылаться на статические члены непосредственно по именам, не уточняя класса.

---

20. Объясните работу операторов if, switch, while, do-while, for, for-each. Напишите корректные примеры работы этих операторов.

if - условный оператор, если значение true, то будет выполнено условие
```
if (x < 0){
return -x;
}
```

switch - условный оператор, очень хорошо подходит для консольных приложений, в отличии от if его удобство заключается в том что тебе не нужно прописывать полное условие в каждом блоке
```
if (n == 1){
   System.out.print("january);
} и т.д.

switch(n){
```

```
case 1 :
   System.out.print("january);
   break;
и т.д.
}
```
если после case не прописать break то после выполнения условия этого case будут выполнены и все остальные до конца лесенки или до ближайшего break. Если совпадений не будет, то исполняется команда после ключевого слова default. Однако оператор default не является обязательным

while - цикл, будет выполняться до тех пор пока условие true.

```
int x = 5;
while (x>0){
   System.out.print("Have a nice day");
x--;
}
```

do-while - аналогичен while, но если в первом случае при начальном условии false программа может даже не зайти в цикл, то в случае с do-while хотя бы один раз, да программы выполнит прописанное действие

```
i = 0;
do{
   System.out.print("Have a nice day");
} while (i > 0){
   System.out.print("Have a nice day");
}
```

for - цикл, практически аналог while, однако главное удобство что начальную, конечную итерацию и её шаг, можно прописать сразу в условии. Идеален для массивов.

```
for (int i = 0; i < arr.length; i++){
   System.out.print(arr[i]);
}
```

for - each - работает как и for однако если в for приходилось прописывать начальные, конечные значения и шаг, то в for-each итератор сам занимается перебором.

```
for (int i : arr){
   System.out.print(i);
}
```

---

21. Объясните работу оператора instanceof. Что будет результатом работы оператора, если слева от него будет стоять ссылка, равная null?

Оператор instanceof нужен чтобы проверить был ли объект на который ссылается переменная x, создан на основе какого-либо класса Y.

x instanceof Y

Что касается null, то
Dog d;
System.out.print(d instanceof Dog);
Выдаст ошибку.

Но, если написать
System.out.print(null instanceof Dog);
то результатом работу будет false.
