**Java Core. Java Fundamental.**

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

Java – как язык программирования – набор инструментов, с помощью которых можно создать приложение/программу.

Со временем Java легла в основу других языков (типа Kotlin, Groovi), т.о. область ее применения расширилась и Java рассматривается как платформа.

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

Имя класса пишется с большой буквы и всегда совпадает с названием java-файла. Есть правило, по которому в одном файле должен храниться один класс. Теоретически можно создать и второй класс внутри файла, но только у одного будет атрибут доступа *public*.

Также нельзя создать интерфейс и класс с одинаковым именем, т.к. оба они скомпилируются в файлы с расширением .class

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

JVM – Java Virtual Machine

JDK – Java Development Kit

JRE – Java Runtime Environment

JDK (Java Development Kit) - комплект разработчика приложений на языке Java, включающий в себя JRE, компилятор, стандартные библиотеки классов Java, примеры, документацию, различные утилиты.

JRE (Java Runtime Environment) - среда выполнения Java приложений. Включает в себя JVM (виртуальная машина Java) и минимальный набор библиотек Java классов.

JVM (Java Virtual Machine) - виртуальная машина Java, исполняет байт-код Java, предварительно созданный из исходного текста Java программы компилятором Java. javac (Java Compiler) - компилятор java проверяет код на синтаксис, лексику, семантику, оптимизирует его, и в конечном итоге, генерирует файл с расширением .class содержащий байт-код.

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

Вызвать командную строку (cmd). При необходимости сменить директорию (cd/) прописать

javac *имяфайла.java* -- Появится новый класс с расширением .class

java *имяфайла* -- для запуска приложения

Путь переменных среды окружения надо прописывать для компилятора??????? Этот путь указывает на место расположение файлов **javac.exe** и **java.exe**.

**5. Перечислите атрибуты доступа, объясните их действие.**

Атрибуты доступа дают возможность управлять инкапсуляцией на уровне класса

- public – классы, методы и переменные с этим атрибутом видны всем классам и методам и переменным (интерфейсный класс, виден всем).

- private – использования метода/поля в точке его написания. Private могут быть классы, методы, поля. Для доступа к полям используются специальные методы (геттеры и сеттеры).

- protected – доступен в производном классе, *класс не может быть protected.*

- friendly (по умолчанию) – инкапсуляция на уровне пакета. Чуть больше, чем private. Виден внутри класса и внутри пакета. Friendly могут быть объявлены классы, поля и методы.

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

Пакеты в java = каталоги в файловой системе. Они нужны для структурирования, а также создания уникальности имен классов (исключение конфликтов имен). Т.е. может быть создано несколько классов с одинаковыми именами различными пользователями, они могут использоваться в одном приложении, но из-за именования пакетов и уникальной иерархии их можно различать.

Java.util.Date и java.sql.Date

Правила именования пакетов:

-маленькая буква;

-два_слова;

- 4-8 пакетов;

- единственное число.

Пакеты позволяют сделать другой вид инкапсуляции (архитектурная), где можно нарушать область видимости.

**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 подпакетах? Какой пакет в Java импортируется по умолчанию?**

java.lang импортируется по умолчанию, содержит класс System. При импорте пакета нужно отдельно подключать его подпакеты.

Есть возможность импортировать все классы пакета сразу java.util.* но этого лучше избегать, т.к. это усложняет восприятие кода другими разработчиками.

Java.util.Date и java.sql.Date в этом случае импортировать можно только 1 класс, к другому придется обращаться по полному имени, но рекомендуется обращаться к каждому по полному имени, чтобы не возникало путаницы.

**8. Объясните различия между терминами “объект” и “ссылка на объект”.**

Типы данных бывают примитивные и ссылочные. Примитивные переменные содержат значение, а переменные ссылочного типа не содержат сами данные, а содержат в себе ссылку на объект.

Объекты создаются в heap (new выделяет память под объект), а локальные переменные в stack, *в том числе и ссылки на объект????????*

Ссылка имеет тип и объект имеет тип, в большинстве случаев они будут различные.

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

*Примитивные типы описываются одним значением! Ссылочные типы так описать невозможно (комплексное значение).*

*Примитивные типы нельзя использовать всегда, например, в Коллекциях, их нужно обернуть в класс-оболочку.*

*Вычисления с примитивными типами выполняются быстрее в несколько десятков раз, т.к. выполняются процессором!*

Примитивных типов в Java 8 шутк:

byte, short, int, long, float, double, char, boolean,. При создании переменной нужно определиться с ее типом (тип определяет, какие значения может хранить переменная) и именем переменной (в соответствии с Code convention).

int i; double z; char = ‘a’;

int char хранятся в одном месте, символы закодированы в таблице Юникод в 16-ричной системе счисления.

В Java все типы данных знаковые, т.о. 1-ый бит хранит в себе информацию о знаке (+/-).

*Значения переменных примитивных типов копируются в метод*!!! Параметры в метод передаются по значению!

**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 (1 байт = 8 бит), short (16 бит = 2 байта), int (32 бит = 4 байта), long (64 бита = 8 байт), double (8 байт), float(32 бита = 4 байта), boolean(точно не определен 1-2 бита, ответ на вопрос да/нет), char(2 байта).

Размер числа, которое может хранить переменная определенного типа определяется как $2^n$, n – размер переменной в битах).

*Boolean не приводятся ни к какому другому примитивному типу.*

Если в задаче использование примитивных типов приводит к переполнению, то это основание использовать оболочку.

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

**Неявное приведение типов** – в переменную с большим диапазоном значений записать переменную с меньшим диапазоном значений.

byte->short ->int (char ->int) → long.

При переходе с int long во float, double теряется точность вычислений!!!

При неявном приведении типов java все сделает автоматически.

**Явное преобразование** – в переменную с меньшим диапазоном записывается переменная с большим диапазоном. *Неизбежная потеря точности, при этом вся ответственность с компилятора снимается!*

Для контроля точности вычислений пишутся специальные численные методы или используются классы-оболочки (как в java).

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

Литералы – значения переменных, которые можно изменить в программе.

X = 2 – (2 – литерал)

True/false

10тичнная система 0-9

16ричная: 0x…- часто используется в кодах символов

8ричная: 0… - чтобы отличить от других систем

long …L/l

double 3.14

float 3.14f

‘1 символ’ ‘/n----’

Управляющие символы: /n, /t, //, /r, /f

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

Если при выполнении арифметических операций результат выходит за рамки диапазона примитивного типа, то **Java это никак не контролирует, в итоге результат будет неожиданным** (Java не контролирует переполнение). При выходе за

рамки диапазона счет начинается с наименьшего отрицательного числа. Т.о. существует возможность сравнить исходные данные и полученный результат, если при сложении он меньше начальных данных, то произошел выход за пределы.

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

Объекты классов-оболочек не изменяются (как и класс String). Они используются в Java для контроля точности вычислений. Если вычисления выходят за рамки диапазона конкретного типа, то это основание для использования класса-оболочки.

Вычисления происходят не в процессоре, поэтому они работают медленнее и именно по этой причине до сих пор не отказались от использования примитивных типов.

**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.**

*Примитивные типы описываются одним значением! Ссылочные типы так описать невозможно (комплексное значение).*

*Примитивные типы нельзя использовать всегда, например, в Коллекциях, их нужно обернуть в класс-оболочку.*

*Вычисления с примитивными типами выполняются быстрее в несколько десятков раз, т.к. выполняются процессором!*

В метод передается параметр:

**для ссылочных типов** – в стеке выделяется память, туда записывается адрес ссылки на объект в куче. **Происходит изменение объекта в методе и вне метода объект по ссылке будет уже измененный!**

Для **примитивного** типа аналогично копируется значение переменной (но в ней значение, а не ссылка), поэтому фактические параметры не изменяются!

**Все параметры в метод передаются по значению независимо от типов!!!**

**16. Поясните, что такое автоупаковка и автораспаковка.**

Автоупаковка и распаковка это функция преобразования примитивных типов в объектные и наоборот.

Автоупаковка применяется компилятором Java в следующих условиях:

- Когда значение примитивного типа передается в метод в качестве параметра метода, который ожидает объект соответствующего класса-оболочки.
- Когда значение примитивного типа присваивается переменной, соответствующего класса оболочки.

**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.**

+, -,/, *, |, ||, &, &&,>,<,==,!=,>>,<<,&(логическое умножение), |(логическое сложение), ^(логическое исключение)

Приоритет выполнения операторов распределен в соответствии с математическими правилами.

**18. Укажите правила выполнения операций с плавающий точкой в Java (согласно стандарту IEEE754). Как определить, что результатом вычисления стала бесконечность или нечисло?**

Например, бесконечность и любое число в сумме дают бесконечность. Конечное число, деленное на бесконечность, равно 0. Любое конечное число, разделенное на 0, стремится к бесконечности.

А что получится, если бесконечность разделить на бесконечность? Результат не определен. Для такого случая существует другой специальный формат - не число (Not a Number, NaN). Его тоже можно использовать в качестве операнда.

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

Статический импорт – импортирует (подключает) статические поля и методы. Static private импортировать нельзя (это логично, т.к. нарушится принцип инкапсуляции)!!!

При статическом импорте, как и при обычном, есть возможность обращаться к переменным/методам только по имени. Пользоваться этим или нет, решает разработчик.

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

Операторы управления:

Оператор условного ветвления (условного выбора) **if** имеет вид:

if (логическое условие){ блок 1}

else {блок 2}

Если в блоке 1/2 всего один оператор, то Java позволяет убрать скобки, но по Cod Convetion этого делать нельзя!

Если условие истинно, то выполняется код блока 1, если условие ложно – код блока 2. Одновременно 2 блока выполняться не могут. Else может отсутствовать, в этом случае, если условие ложно, то ничего не произойдет.

Оператор **if** может быть вложенным (как в блоке if, так и в блоке else), но работает по такому же принципу.

Код по ветке if должен быть ожидаемый!!! Чтобы было удобно читать.

**Тернарный оператор:** условие?истина:ложь (краткая запись if - else)

Оператор **switch(множественный оператор выбора)** имеет вид:

**switch** (выражение){

**case** значение 1: операторы; **break**;

…

**case** значение n: операторы; **break**;

**default**: операторы; **break**;

}

Выражение сравнивается со значениями в кейсах. Если значение совпало, то выполняются операторы до break; если ни одно значение не совпало, то выполняются операторы по default (но он может отсутствовать).

Необходимо следить за тем, чтобы после каждого кейса стоял break, иначе будут выполняться по порядку другие кейсы.

Switch работает не со всеми типами данных: только целочисленные и char! Но можно использовать String или enum(перечисления)

Циклический оператор (цикл с предусловием) **while** имеет вид:

while (условие ){

операторы;}

пока условие истинно, выполняются операторы. Если условие ложно, то операторы не выполнятся ни разу!

Циклический оператор (цикл с постусловием) **do - while** имеет вид:

do {операторы;

} while (условие)

Отличие данного цикла от предыдущего в том, что операторы выполнятся хотя бы 1 раз!!!

Если в цикле встретится оператор **break;**, то компилятор прервет выполнение цикла и выйдет из него.

Если в теле цикла встретится оператор **continue;**, то компилятор пропускает все операции после и начинает выполнение цикла с новой итерации (надо быть внимательным, чтобы условие изменилось и снова не попасть на **continue**).

Цикл for имеет вид:

**for (**условие начала цикла; условие выходы из цикла; условие изменения шага цикла) {

операторы;
}

Можно использовать break; continue; Цикл for может не выполниться ни один раз, а если оставить все скобки пустыми, то может зациклиться, хотя IDE это отследит и не даст скомпилировать файл.

**For-each** используется для массивов и коллекций, чтобы пробежаться по всем элементам, *но в нем нельзя изменить сами элементы???????*

Цикл for используется, если заранее известно количество итераций, а циклы while? когда не известно!

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

## МАССИВЫ

Вывод на экран **System.out.printf("%-5.3f ", mas[i])** – сохранение равных промежутков для красивого представления!!!

**1. Дайте определение массиву. Как осуществляется индексация элементов массива. Как необходимо обращаться к i-му элементу массива?**

Массив – конечная последовательность упорядоченных элементов одного типа, доступ к каждому элементу в которой осуществляется по его индексу.

Массив является объектом, поэтому объявление **int [] mas = new int[5];**

Индексирование массива начинается с нуля. Обращение к элементу по индексу: mas[i]. Длина массива (количество элементов) не изменяется mas.length – длина массива!

**2. Приведите способы объявления и инициализации одномерных и двумерных массивов примитивных и ссылочных типов. Укажите разницу, между массивами примитивных и ссылочных типов.**

**int [] mas1, mas2;** -- ссылки на массивы типа int;
**int mas1[], mas2;** -- ссылка на массив и объявление переменной типа int.
**int [] mas = new int[5];**
**int [] mas = new int[]{1, 0, 3, 5};**
**int [] mas = {1, 0, 3, 5}; -- не всегда работает. Только в таком виде**
**int n = 5; int [] mas = new int[n];**

**Многомерные массивы:**

Объявление и инициализация многомерного массива
**int[][] array = new int[3][3];**
Начальная инициализация массива
**int[][] array = { {1,2,3},{4,5,6},{7,8,9}};**
Доступ к элементам многомерного массива
**array[i][j]** – где **i** и **j** индексы

**3. Объясните, что представляет собой двумерных массив в Java, что такое "рваный массив". Как узнать количество строк и количество элементов в каждой строке для "рваного" массива?**

В Java по сути нет понятия двумерного массива, есть понятие массива массивов, т.е. в каждой ячейке внешнего массива может находиться массив, причем разной длинны. Это и есть понятие "рваного массива", т.е. нет конкретного представления двумерного массива в виде таблицы и тд.

Количество элементов в каждой строчке
**array.length** – кол-во элементов в массиве
**array[i].length** – кол-во элементов в строке

**4. Объясните ситуации, когда в java-коде могут возникнуть следующие исключительные ситуации java.lang.ArrayIndexOutOfBoundsException и java.lang.ArrayStoreException.**

**java.lang.ArrayIndexOutOfBoundsException –** Это исключение возникает, когда программа пытается адресовать элементы за пределами массива. Выход за пределы массива. Такое бывает при переборе всех элементов массива, когда неверно выставлены границы массива. В цикле for условие истинно, а индекс перешагнул фактическое значение количества элементов в массиве.

**java.lang.ArrayStoreException –** Если попытаться записать в ячейку массива ссылку на объект неправильного типа, возникнет исключение ArrayStoreException.

**5. Что такое динамические структуры данных? Какие динамические структуры данных вы знаете? Приведите пример реализации любой динамической структуры 'с нуля'.**

**Динамический массив** — это массив, который умеет изменять свой размер во время выполнения программы. В Java эту роль играют в основном классы ArrayList и LinkedList. В отличие от массивов, ArrayList и LinkedList содержат только ссылочные типы данных, то есть сохранить в них можно только объекты. К счастью, в Java существуют механизмы автоупаковки и автораспаковки, которые позволяют хранить в динамических массивах примитивные типы. Подобно статическому массиву, динамический однороден, то есть может хранить один-единственный тип данных. Однако, благодаря механизму наследования и грамотному использованию интерфейсов,

можно сохранять в одном динамическом массиве целый спектр разнообразных классов, которые унаследованы от одного общего.

**6. Дайте определение таким понятиям как ”класс” и “объект”. Приведите примеры объявления класса и создания объекта класса. Какие спецификаторы можно использовать при объявлении класса.**

Java – файл имеет с классом одинаковое имя. Описание свойств (поля) и алгоритмов действий (методы) какой-то предметной области.

Объект – конкретный экземпляр класса.

```
public class User{
int age;
String name;
public void work(){}
}

User user1 = new User();
user1.age=25;
```

Для класса можно использовать public, friendly, private

**7. Как вы определяете, какие поля и методы необходимо определить в классе, приведите пример. Какие спецификаторы можно использовать с полями, а какие с методами (и что они значат)?**

**Поля (private, public, )**

**8. Что такое конструктор? Как вы отличите конструктор от любого другого метода? Сколько конструкторов может быть в классе? Что такое конструктор по умолчанию, может ли в классе совсем не быть конструкторов? Объясните, что делает оператор this() в конструкторе?**

Конструктор – метод класса, который объявляет переменные класса (по умолчанию) или инициализирует их в момент создания экземпляра класса.

Если конструктор не определен, то предоставляется конструктор по умолчанию без параметров.

В классе может быть несколько конструкторов (перегруженные) с различными параметрами.

Имя конструктора совпадает с именем класса и он не выполняет никаких действий, кроме инициализации.

this.x=x

**9. Приведите правила, которым должен следовать компонент java-bean.**

**Формальные и фактические параметры**