Теоретическое задание по первой части курса JC.

После каждого вопроса должен быть пункт с ответом и источником (пример - первые 2 вопроса).

Источник выбираете не первый попавшийся, а тот где на ваш взгляд наиболее понятная и полная информация.

Считаются как сайты, так и книги с рабочей тетрадью.

1. Что такое технология Java?

**Ответ.** Java является одновременно языком программирования и платформой. Технология Java это комплекс технологий, состоящий из высокоуровневого строго типизированного объектно-ориентированного языка программирования, компилятора в байт код, виртуальной машины (интерпретатора в машинный код). Технология Java позволяет один код использовать на любом устройстве для которого существует виртуальная машина Java.

**Источник.** <https://www.ibm.com/developerworks/ru/java/newto/>

1. Что такое JVM?

**Ответ.** Java Virtual Machine (Виртуальная машина Java). Основная часть исполняющей системы Java. JVM интерпретирует и выполняет байт код ранее скомпилированный из исходного кода программы.

**Источник.** <https://ru.wikipedia.org/wiki/Java_Virtual_Machine>

1. Что такое JDK?

**Ответ.** Java Development Kit (комплект разработки Java). Бесплатно распространяемый комплект разработчика приложений на языке Java включающий в себя компилятор (javac), стандартные библиотеки классов, примеры, документацию, различные утилиты и исполнительную систему JRE.

**Источник.** Лекция 1.

1. Что такое JRE?

**Ответ.** Java Runtime Environment (исполняющая система Java). Минимальная реализация виртуальной машины, необходимая для исполнения Java-приложений. Состоит из JVM и библиотек.

**Источник.** Лекция 1.

1. Перечислите группы примитивных типов данных, состав каждой группы и их размерность (в байтах).

**Ответ.** Целочисленные: byte (1), short (2), int (4), long (8).

Вещественные float (4), double (8).

Символьные char (2).

Логические boolean (true, false).

**Источник.** Лекция 2.

1. Какими литералами можно инициализировать символьную переменную?

**Ответ.** Символьную переменную можно инициализировать двумя одиночными кавычками, пустыми или содержащими один любой символьный, цифровой или непечатный символ, или юникс представление любого из них.

**Источник.** Лекция 2.

1. Какие есть виды переменных?

**Ответ.** Объекта, локальные, класса (статическая переменная объекта)

**Источник.** <https://javarush.ru/groups/posts/2630-prostihe-peremennihe-v-java-i-konstantih>

1. В какой области памяти хранятся локальные переменные?

**Ответ.** В стэке.

**Источник.** <https://ru.stackoverflow.com/questions/980439/%D0%92-%D0%BA%D0%B0%D0%BA%D0%BE%D0%B9-%D0%BE%D0%B1%D0%BB%D0%B0%D1%81%D1%82%D0%B8-%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D0%B8-%D1%85%D1%80%D0%B0%D0%BD%D1%8F%D1%82%D1%81%D1%8F-%D0%BF%D0%B5%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5-%D0%B2-java>

1. В какой области памяти хранятся аргументы метода?

**Ответ.** В стэке.

**Источник.** <https://topjava.ru/blog/stack-and-heap-in-java>

1. В какой области памяти хранятся объекты?

**Ответ.** В куче.

**Источник.** <https://topjava.ru/blog/stack-and-heap-in-java>

1. В какой области памяти хранятся ссылки на объекты?

**Ответ.** В стэке.

**Источник.** <https://topjava.ru/blog/stack-and-heap-in-java>

1. В какой области памяти хранятся массивы?

**Ответ.** В куче. Так как являются объектами.

**Источник.** <https://topjava.ru/blog/stack-and-heap-in-java>

1. Дайте определение массива.

**Ответ.** Массив — это конечная последовательность однотипных элементов, доступ к объектам осуществляется по их индексу в массиве.

**Источник.** Лекция 4

1. Назовите 3 этапа создания массива. Какие действия выполняет JVM на каждом этапе?

**Ответ.** Объявление. Определяется переменная ссылочного типа на массив, содержащая тип массива.

Создание. Указывается количество элементов массива, выделяется память и переменной присваивается ссылка.

Инициализация. Каждому элементу массива задаётся значение, указанное при инициализации либо по умолчанию.

**Источник.** <http://pr0java.blogspot.com/2015/05/1.html>

1. Приведите различные способы инициализации массива.

**Ответ.** int[] a = new int[3];

int[] b = new int[]{1,3,4};

int[] c = {1,3,4};

**Источник.** <http://pr0java.blogspot.com/2015/05/1.html>

1. Что собой представляет двумерный массив? Многомерный массив?

**Ответ.** Двумерный массив представляет собой массив однотипных массивов. Многомерный массив состоит из массива однотипных массивов, содержащих в себе так же однотипные массивы до последнего уровня. В последнем уровне массивов содержатся значения.

**Источник.** <http://pr0java.blogspot.com/2015/05/3.html>

1. Какая структура у java-файла (внутренности класса не указывать)?

Какие из элементов структуры являются обязательными?

**Ответ.** Java файл должен содержать указание пакета, в котором находится файл (package ... Обязательный элемент за исключением случая, когда файл лежит в корне проекта). Далее указываются импортируемые классы (import… Не обязательный элемент). Далее записывается идентификатор доступа (не обязательно) сам класс, классов в одном файле может быть несколько (class. Обязательный элемент, желательно не создавать несколько классов в одном файле), идентификатор класса (обязательный).

**Источник.** <http://www.myshared.ru/slide/653672/>

1. Для чего предназначены пакеты? Как выглядит полное имя класса?

**Ответ.** Пакеты используются для логической группировки классов, создания пространства имён и ограничения доступа к классам. Полное имя класса состоит из пакета имени самого класса все это записывается через точку, например, «java.util.Collections».

**Источник.** Лекция 5.

1. Для чего предназначен оператор import?

**Ответ.** Для импорта классов из других пакетов в текущий файл.

**Источник.** <https://pro-java.ru/java-dlya-nachinayushhix/import-paketov-v-java/>

1. Может ли в классе быть два и более метода main()?

**Ответ.** Метод main с которым может работать JVM может быть только один, он должен иметь такой заголовок: «public static void main(String[] args)», однако main тоже метод и тоже может перегружаться. Да, может.

**Источник.** <http://zonakoda.ru/vsyo-o-metode-main.html>

1. Какие есть виды комментариев?

**Ответ.** Однострочные, многострочные и документация (doc-комментарии).

**Источник.** <https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%80%D0%B8%D0%B8_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)#%D0%90%D0%BD%D0%BD%D0%BE%D1%82%D0%B0%D1%86%D0%B8%D0%B8>

1. Какие уровни доступа есть у класса на верхнем уровне, т.е. у внешнего класса?

**Ответ.** Public, default.

**Источник.** <https://javarush.ru/groups/posts/1988-modifikatorih-dostupa-private-protected-default-public>

1. Что является членами класса?

**Ответ.** Поля, методы, вложенные классы и вложенные интерфейсы.

**Источник.** <http://src-code.net/chleny-klassa-java/>

1. Что еще можно определить в классе кроме членов?

**Ответ.** Логические блоки, конструктор.

**Источник.** <http://java-online.ru/java-class.xhtml>

1. Какие уровни доступа есть у членов класса?

**Ответ.** private, default, protected, public.

**Источник.** Лекция 5.

1. Какие модификаторы необходимы для определения константы:

1. Класса,

2. экземпляра класса?

**Ответ.** 1. final static; 2. final.

**Источник.** Лекция 5.

1. Может ли класс не иметь ни одного конструктора? Почему?

**Ответ.** Внутри класса может быть не определено ни одного конструктора, тогда у него будет конструктор по умолчанию.

**Источник.** Книга «Java 8» Шилдт Г. Стр. 170

1. Перечислите отличия конструктора от метода.

**Ответ.** У конструктора не указывается тип возвращаемых данных, идентификатор конструктора всегда совпадает с идентификатором класса, вызывается только с помощью оператора «new» при создании объекта.

**Источник.** Книга «Java 8» Шилдт Г. Стр. 168

1. Как и в каком месте можно вызвать из одного конструктора другой конструктор этого же класса?

**Ответ.** С помощью слова «this» можно вызвать один конструктор в самом начале другого конструктора. (this()).

**Источник.** <https://javarush.ru/groups/posts/1391-konstruktorih-klassov-java-jdk-15>

1. Как и в каком месте можно вызвать конструктор суперкласса?

**Ответ.** С помощью слова «super» можно вызвать конструктор суперкласса в самом начале конструктора. (super()).

**Источник.** <https://javarush.ru/groups/posts/1391-konstruktorih-klassov-java-jdk-15>

1. Что делает компилятор, если в конструкторе нет вызова другого конструктора или конструктора суперкласса?

**Ответ.** Добавляет в начало конструктора вызов конструктора суперкласса.

**Источник.** <https://javarush.ru/groups/posts/1391-konstruktorih-klassov-java-jdk-15>

1. Можно ли одновременно использовать конструкции this() и super() в одном и том же конструкторе?

Обоснуйте ответ.

**Ответ.** Нельзя. Каждая из этих конструкций должна стоять первой в конструкторе, обе первыми быть не могут.

**Источник.** <https://javarush.ru/groups/posts/1391-konstruktorih-klassov-java-jdk-15>

1. Может ли быть приватный уровень доступа у конструктора? Если да, то для каких целей он необходим?

**Ответ.** Да.

- Для создания только одного объекта через публичный статический метод этого класса.

- При создании статического класса, для запрета создания его экземпляра.

**Источник.** <http://www.quizful.net/interview/java/private-constructor-singleton>

1. Можно ли наследовать конструктор?

Если да, то приведите примеры.

**Ответ.** Конструктор не является членом класса, по этому его нельзя наследовать, однако можно вызвать конструктор родителя при помощи «super()».

**Источник.** <http://proglang.su/java/inheritance>

1. Какие модификаторы может иметь конструктор?

**Ответ.** private, default, protected, public.

**Источник.** <https://javarush.ru/groups/posts/1391-konstruktorih-klassov-java-jdk-15>

1. Какой тип возвращаемого конструктором значения?

**Ответ.** Ссылочный тип текущего класса.

**Источник.** <https://javarush.ru/groups/posts/1391-konstruktorih-klassov-java-jdk-15>

1. Перечислите случаи, когда используется статический блок?

В каких случаях можно обойтись без него. Если можно, то каким образом?

**Ответ.** Статический блок используется для инициализации статических полей (констант) класса, позволяют вызывать статические методы и выводить информацию. Обойтись без статического блока можно, когда не требуется проверять условия для инициализации статических полей (констант) класса или выводить информацию. Инициализацию полей (констант) можно осуществлять сразу во время объявления или в конструкторе.

**Источник.** <http://javaway.info/chto-takoe-staticheskij-blok-i-staticheskaya-initsializatsiya-v-java/>

1. Перечислите случаи, когда используется логический блок?

В каких случаях можно обойтись без него. Если можно, то каким образом?

**Ответ.** Логический блок используется для инициализации полей (констант) класса, позволяют вызывать методы и выводить информацию. Обойтись без логического блока можно, когда не требуется проверять условия для инициализации полей (констант) класса или выводить информацию. Инициализацию полей (констант) можно осуществлять сразу во время объявления или в конструкторе.

**Источник.** <http://javaway.info/chto-takoe-staticheskij-blok-i-staticheskaya-initsializatsiya-v-java/>

1. Что входит в сигнатуру метода?

**Ответ.** Сигнатура метода — это имя метода плюс параметры (причем порядок параметров имеет значение). В сигнатуру метода не входит возвращаемое значение, а также бросаемые им исключения.

**Источник.** <https://javarush.ru/groups/posts/1249-signatura-metoda>

1. Что значит ключевое слово native?

**Ответ.** Java Native Interface (JNI) — стандартный механизм для запуска кода, под управлением виртуальной машины Java (JVM), который написан на языках С/С++ или Ассемблера, и скомпонован в виде динамических библиотек, позволяет не использовать статическое связывание. Это даёт возможность вызова функции С/С++ из программы на Java, и наоборот.

Модификатор native сигнализирует о том, что метод реализован в платформо-зависимом коде, часто на языке С.

Этот модификатор может быть применен только к методам, но не классам и переменным.

Тело нативного метода должно заканчиваться на (;) как в абстрактных методах, идентифицируя то, что реализация опущена.

**Источник.** <http://www.quizful.net/interview/java/native-keyword>

1. К каким данным можно обратиться в статическом методе?

**Ответ.** К статическим.

**Источник.** <https://javarush.ru/groups/posts/modifikator-static-java>

1. Почему нельзя объявить статическое поле внутри метода?

**Ответ.** Статическое поле принадлежит классу, что противоречит локальной переменной.

**Источник.** <https://javarush.ru/groups/posts/modifikator-static-java>

1. Перечислите, какой код в классе можно выполнить до создания объекта.

**Ответ.** Статические блоки родителя, статические блоки текущего класса, логические блоки родителя, конструктор родителя, логические блоки текущего класса, конструктор текущего класса.

**Источник.** <https://javarush.ru/groups/posts/2137-porjadok-deystviy-pri-sozdanii-obhhekta>

1. Для чего предназначен метод toString()?

В каких случаях он вызывается?

**Ответ.** Для преобразования объекта в строку. В случае необходимости предоставления данных объекта в виде строки.

**Источник.** <https://javarush.ru/groups/posts/695-10-podskazok-po-pereopredeleniju-metoda-tostring-v-java-chastjh-1>

1. Как реализован метод toString() в классе Object?

**Ответ.** Он возвращает полное имя класса объекта (с пакетом), собаку и хэшкод объекта в шестнадцатеричном представлении в виде строки.

**Источник.** <https://javarush.ru/groups/posts/695-10-podskazok-po-pereopredeleniju-metoda-tostring-v-java-chastjh-1>

1. Для чего предназначен метод equals()?

В каких случаях он вызывается?

**Ответ.** Метод предназначен для определения равенства двух объектов. Вызывается кода нужно сравнить значения полей объектов.

**Источник.** <https://javarush.ru/groups/posts/equals-java-sravnenie-strok>

1. Как реализован метод equals() в классе Object?

**Ответ.** Метод сравнивает ссылки двух объектов на равенство.

**Источник.** <https://javarush.ru/groups/posts/equals-java-sravnenie-strok>

1. В чем различие между методом equals() и операции ==?

В каких случаях применение метода equals() и операции == эквивалентно?

**Ответ.**

**Источник.**

1. Что такое garbage collection?

**Ответ.**

**Источник.**

1. Для чего предназначен метод finalize()?

**Ответ.**

**Источник.**

1. Для чего предназначены оболочечные классы?

**Ответ.**

**Источник.**

1. Что такое autoboxing и unboxing?

**Ответ.**

**Источник.**

1. Какой класс является родительским для всех классов?

**Ответ.**

**Источник.**

1. Какой класс является родительским для всех перечислений?

**Ответ.**

**Источник.**

1. Какой интерфейс является родительским для всех интерфейсов?

**Ответ.**

**Источник.**

1. Для каких целей необходим интерфейс, в котором не объявлено ни одного метода?

**Ответ.**

**Источник.**

1. Можно ли переопределить метод в том же самом классе? Можно ли его перегрузить в этом классе?

**Ответ.**

**Источник.**

1. Можно ли переопределить метод в подклассе? Можно ли его перегрузить в подклассе?

**Ответ.**

**Источник.**

1. Можно ли переопределить статический метод? Поясните ответ.

**Ответ.**

**Источник.**

1. Как запретить переопределение метода в подклассе? Назовите 2 способа.

**Ответ.**

**Источник.**

1. С помощью какого механизма реализуется полиморфизм в Java?

**Ответ.**

**Источник.**

1. В чем заключается отличие сравнения принадлежности к классу через операцию instanceof и метод getClass()?

**Ответ.**

**Источник.**

1. Могут ли в интерфейсе быть поля?

**Ответ.**

**Источник.**

1. Можно ли в интерфейсе

1. объявить метод с пакетным уровнем доступа?

2. объявить конструктор?

3. определить конструктор?

**Ответ.**

**Источник.**

1. Можно ли в интерфейсе объявить статический метод? Поясните ответ.

**Ответ.**

**Источник.**

1. Можно ли создать экземпляр класса, у которого есть абстрактный метод? Если да, то зачем?

**Ответ.**

**Источник.**

1. Должен ли иметь абстрактный класс хотя бы один абстрактный метод? Если нет, то зачем объявлять такой класс абстрактным?

**Ответ.**

**Источник.**

1. Можно ли объявить метод одновременно финальным и абстрактным? Поясните ответ.

**Ответ.**

**Источник.**

1. На какие две группы разделяются классы, объявленные внутри другого класса?

**Ответ.**

**Источник.**

1. Для каких целей они используются?

**Ответ.**

**Источник.**

1. Какие уровни доступа применяются к таким классам?

**Ответ.**

**Источник.**

1. Какие существуют варианты внутренних классов?

**Ответ.**

**Источник.**

1. Можно ли из вложенного класса обратиться к членам внешнего класса?

**Ответ.**

**Источник.**

1. Можно ли из внутреннего класса обратиться к экземпляру внешнего класса?

**Ответ.**

**Источник.**

1. Можно ли определить экземпляр вложенного класса, не определяя экземпляры внешнего класса?

**Ответ.**

**Источник.**

1. Какие существуют варианты внутренних интерфейсов?

**Ответ.**

**Источник.**

1. Что представляют собой элементы перечисления?

**Ответ.**

**Источник.**

1. Как образуются имена вложенных и внутренних .class-файлов после компиляции?

Приведите примеры.

**Ответ.**

**Источник.**

1. В какой версии Java появились параметризованные типы?

**Ответ.**

**Источник.**

1. Какие типы данных запрещены в качестве параметров классов?

**Ответ.**

**Источник.**

1. Почему нельзя создать generic-поле?

**Ответ.**

**Источник.**

1. Почему статический метод не может иметь generic-параметр?

**Ответ.**

**Источник.**