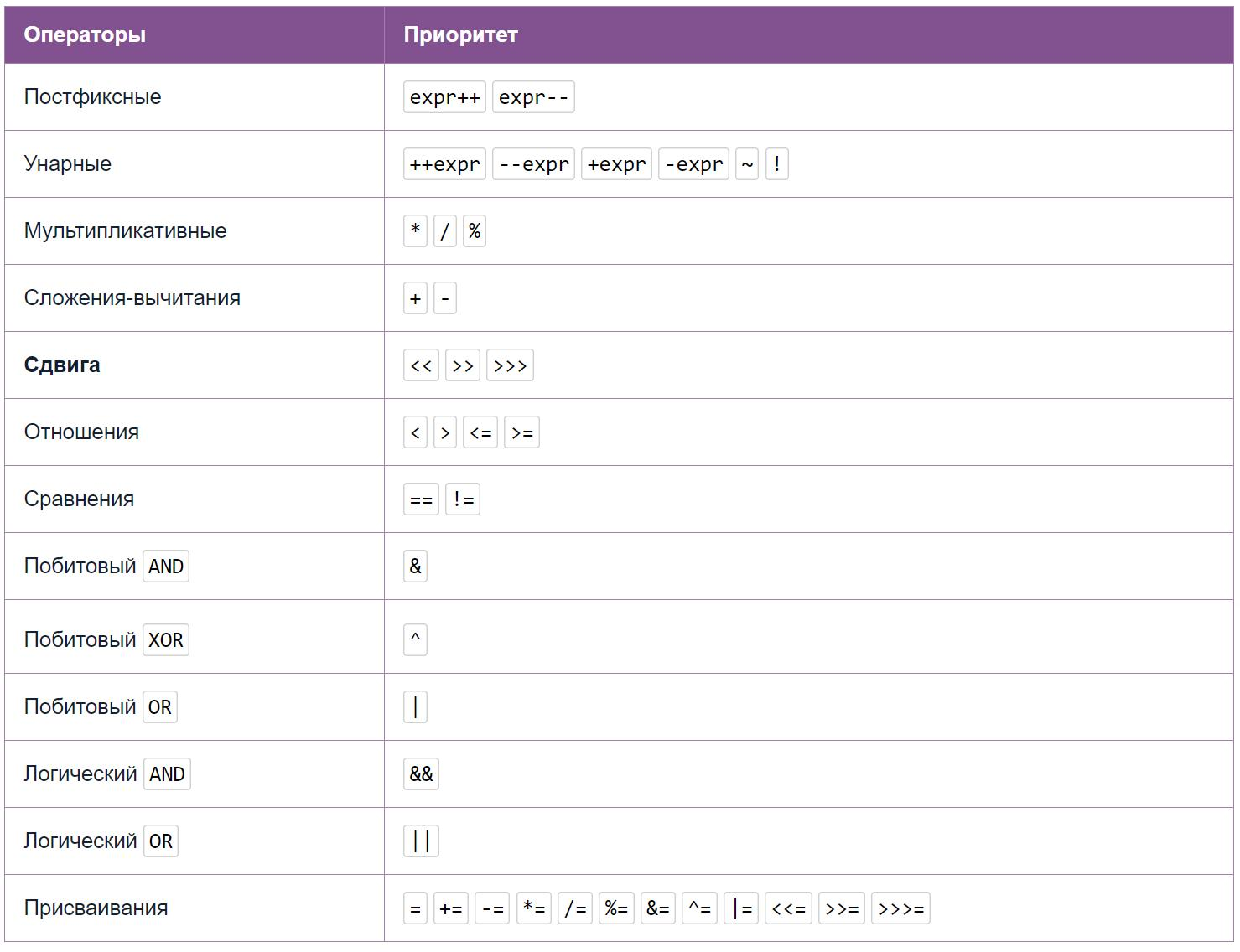
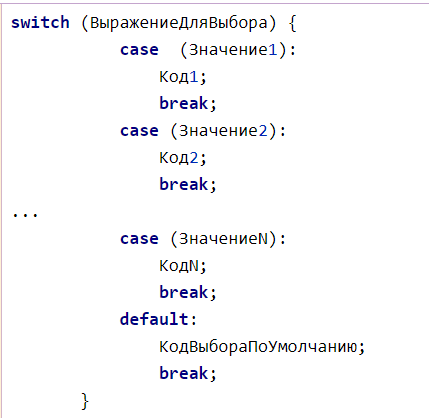
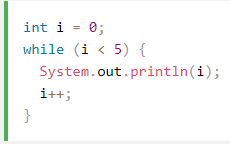
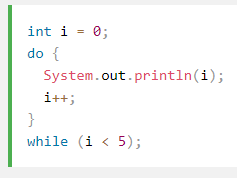
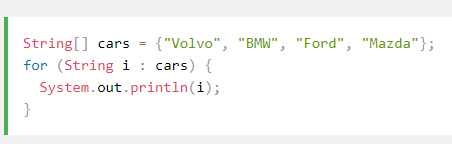
**Basic of software code development**

1. **Java – язык программирования и Java – платформа**
   * **Java – язык программирования.** Межплатформенный объектно-ориентированный язык программирования общего назначения, обладающий широкими возможностями.
   * **Java – платформа.** Это программное обеспечение, предоставляющее нам среду времени исполнения. Это виртуальная машина Java (Java Virtual Machine), линкующая и выполняющая ваш код в том виде, в каком он ей предоставляется. Код предоставляется в виде файлов классов, непригодных для чтения человеком. Иными словами, машина не интерпретирует непосредственно файлы с исходным кодом на языке Java, а требует предварительного преобразования этого кода в файлы классов.
2. **Связь имени Java-файла и классов, в которые в этом классе объявляются.**
   * Имя Java-файла и класс должны совпадать (при нарушении данного закона код компилироваться не будет).
3. **Расшифруйте аббревиатуры JVM, JDK, JRE. Покажите, где «они находятся» и что собой представляют**
   * JavaVirtualMachine - это механизм, предоставляющий среду выполнения для управления Java-кодом или приложениями. Виртуальная машина является независимой оболочкой исполнения кода, благодаря которой возможен её запуск на любой ОС, без влияния ОС на выполняемую программу (обеспечивает независимость от платформы).
   * JavaDevelopmentKit является основным компонентом среды Java и предоставляет все инструменты, исполняемые и бинарные файлы, которые нужны для компиляции, отладки и выполнения программы на Java. JDK является платформо-зависимым программным обеспечением, поэтому есть отдельные инсталляторы для Windows, Mac и Unix-систем. Можно сказать, что JDK является надстройкой JRE, так как он содержит JRE с Java-компилятором, отладчиком и базовыми классами. (для разработки)
   * JavaRuntimeEnvironment - минимальная реализация виртуальной машины, необходимая для исполнения Java-приложений, без компилятора и других средств разработки. (для запуска программ, содержит в себе JVM)
   * JDK -> JVM -> JRE
4. **Как скомпилировать и запустить приложение из командной строки, а также зачем в переменных среды окружения прописывать пути к установленному JDK?**
   * Через команду *javac, указав путь к созданному файлу.*
   * *Так как JDK является основным компонентом среды Java, без него приложение не запустится*
5. **Атрибуты доступа и их действие**
   * Public – объект доступен из любого другого кода проекта
   * Private – объект доступен только внутри класса
   * Protected – объект доступен и для других классов данного пакета, либо в пределах всех классов-наследников
   * Default (package visible) - объект доступен только внутри пакета.
6. **Пакеты в java-программе, что представляют собой пакеты на диске? Каково соглашение по именованию пакетов? Как создать пакет?**
   * Пакет является специальным библиотечным модулем, который содержит группу классов, объединённых в одном пространстве имён (механизм для группировки классов).
   * На диске является каталогом
   * Пакеты всегда пишутся строчными буквами. В отличие от классов, где первая буква обычно является заглавной.
   * Чтобы избежать создания пакетов с такими же именами, как у других общедоступных, рекомендуется начинать иерархию с обратного доменного имени вашей компании. Например, поскольку доменное имя компании — blog.com, надо начать со структуры с именем com.blog. Другими словами, пакет верхнего уровня с именем com с подпакетом внутри называется blog.
   * Создать каталог (папку) через проводник или же в среде разработке ПКМ -> New -> Package
7. **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?**
   * Ключевое слово import
   * импорт пакета позволяет получать доступ к классам данного пакета и подпакетов, лежащих в них, по короткому имени, при условии, что они имеют различные имена
   * пакет java.lang импортируется неявно
8. **Объясните различия между терминами «объект» и «ссылка на объект»**
   * Объект – экземпляр класса
   * Ссылка на объект – переменная, позволяющая обратиться к объекту или классу, но не являющаяся его экземпляром
9. **Какие примитивные типы в Java вы знаете, как создать переменные примитивных типов? Объясните процедуру, по которой переменные примитивных типов передаются в методы как параметры?**
   * Типы данных:
     1. Byte
     2. Short
     3. Int
     4. Long
     5. Float
     6. Double
     7. Char
     8. Boolean
   * Тип данных Переменная (= ее значение) // int a = 6;
   * передача по значению. В этом случае значение аргумента копируется в формальный параметр функции. Поскольку создается копия аргумента в функции, то все изменения над копией не повлияют на значение аргумента;
10. **Какой размер примитивных типов в зависимости от разрядности платформы, что такое преобразование (приведение) типов и зачем оно необходимо? Какие примитивные типы не приводятся ни к какому другому типу?**
    * От разрядности платформы не зависит:
      1. byte (целые числа, 1 байт)
      2. short (целые числа, 2 байта)
      3. int (целые числа, 4 байта)
      4. long (целые числа, 8 байтов)
      5. float (вещественные числа, 4 байта)
      6. double (вещественные числа, 8 байтов)
      7. char (символ Unicode, 2 байта)
      8. boolean (значение истина/ложь, 1 байт)
    * Приведение типов в Java — это присвоение значения одного примитивного типа данных другому. Вы должны знать о совместимости этого типа данных. Если они совместимы, тогда Java выполнит преобразование, известное как автоматическое преобразование типов, а если нет, то их необходимо явно преобразовать.
      1. В Java есть два типа приведения:
      2. Расширение приведения (автоматически) — преобразование меньшего типа данных в больший размер типа. byte -> short -> char -> int -> long -> float -> doubl
      3. Сужение приведения (вручную) — преобразование данных большего размера в тип меньшего размера. double -> float -> long -> int -> char -> short -> byte
    * Boolean не приводится ни к какому другому типу
11. **Объясните, что такое явное и неявное приведение типов, приведите примеры, когда такое преобразование имеет место?**
    * Неявное приведение выполняется автоматически (int a; double b; a +=b);
    * Явное приведение требует специального оператора типа, к которому происходит приведение ((int a; double b; a = a + (int) b);
    * Используется в случае, когда необходимо провести действие с различными типами данных, при условии, что они совместимы для приведения.
12. **Что такое литералы в Java-программе, какую классификацию литералов вы знаете, как записываются литералы различных видов и типов в Java-программе?**
    * Литералы — это явно заданные значения в коде программы — константы определенного типа, которые находятся в коде в момент запуска.
    * Все литералы – примитивные типы данных (числовые (целочисленные, с плавающей точкой), строковые, символьные, логические). Нельзя создать литерал с объектом, кроме *null*
    * Целочисленные:
      1. Int – как обычно (двоичная система – b, восьмеричная – ведущий 0 (05), десятичная – по умолчанию, шестнадцатеричная – префиксы 0x 0X (от 0 до 15)
      2. Long – на конца L
    * С плавающей точкой
      1. Обычная десятичная дробь (3.12321)
      2. Научный вид (4.05E-13 = 4.05 \* 10-13
      3. Float - на конце f
    * Строковые
      1. Строка в кавычках (“Hello world”)
    * Символьные
      1. Символьные литералы в Java представлены кодовой таблицей Unicode, то есть каждый символ — 16-битовое значение. Для обозначения символа в коде его обособляют одинарными кавычками.
      2. По опыту использования бывают два вида символов:
      3. Те, что можно ввести с клавиатуры — обычные символы;
      4. Символы, которые просто так с клавиатуры не ввести (символы различных языков, фигуры и так далее).
      5. Обычные символы можно указать в явном виде: ‘,’ или ‘@’.
      6. Если символ служебный (например, перенос строки или табуляция), такой символ необходимо экранировать обратным слешем.
      7. Символы, которые нельзя просто так ввести с консоли, можно задать в их 16-битовом виде. Для этого необходимо указать код символа с префиксом \u, например ‘\u00F7'.
    * Логический
      1. True / False
13. **Как осуществляется работа с типами при вычислении арифметически выражений в Java?**
    * «+» - сложение
    * «-» - вычитание
    * «\*» - умножение
    * «/» - деление
    * «%» - деление по остатку
    * Приоритет арифметических операций
      1. Одни операции имеют больший приоритет, чем другие, и поэтому выполняются вначале. Операции в порядке уменьшения приоритета:
      2. ++ (инкремент), -- (декремент)
      3. \* (умножение), / (деление), % (остаток от деления)
      4. + (сложение), - (вычитание)
14. **Что такое классы-оболочки, для чего они предназначены? Объясните, что значит: объект класса оболочки – константный объект?**
    * Класс-оболочка обертывает вокруг типа данных и дает ему вид объекта. Они используются для преобразования любого примитивного типа в объект.
    * Так как классы-оболочки неизменные => их объекты тоже являются неизменными. В случае, если нам необходимо перезаписать значение объекта, то переменная будет указывать уже на другой объект с необходимым нам 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 передачу в метод**
    * Примитив содержит значение переменной непосредственно в памяти, выделенной для нее. Ссылочный типа отличается еще называется ссылкой. Не содержит объект, но содержит ссылку на него, которая указывает на другое место в памяти, где он хранится. Через такую ссылку можете получить доступ к полям и методам ссылочного объекта. Допустимо иметь много разных переменных, ссылающихся на один и тот же объект. Это невозможно с примитивами.
    * Передача в методы выполняется по принципу «передача по значению» (скопировать значение и передать копию). Примитивные типы – передается копия текущего значения, ссылки – передается копия значения.
    * Конечная переменная может быть явно инициализирована только один раз. Ссылочная переменная, объявленная final, никогда не может быть переназначена для ссылки на другой объект. Однако данные внутри объекта могут быть изменены. Таким образом, состояние объекта может быть изменено, но не ссылка. С переменными конечный модификатор часто используется со static, чтобы сделать константу переменной класса.
16. **Поясните, что такое автоупаковка и автораспаковка**
    * Автоупаковка — Это автоматическая инкапсуляция примитивного типа в эквивалентную ему класс-обёртку всякий раз, когда требуется объект данного типа.
    * Автораспаковка — Это преобразование класса-обёртки в соответствующий ему примитивный тип. Если при распаковке класс-обёртка был равен null, произойдет исключение java.lang.NullPointerException.
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 их них имеют одинаковый приоритет**
    * Арифметические
      1. +
      2. –
      3. \*
      4. /
      5. %
    * Логические
      1. операция отрицания (NOT);
      2. операция конъюнкции, логическое И (AND);
      3. операция дизъюнкции, логическое ИЛИ (OR);
      4. операция сложения по модулю, исключающее ИЛИ (XOR).
      5. ! — оператор отрицания;
      6. && — оператор логическое И (сокращенный);
      7. || — оператор логическое ИЛИ (сокращенный);
      8. & — оператор побитовое И;
      9. | — оператор побитовое ИЛИ;
      10. ^ — оператор побитовое исключающее ИЛИ.
    * Битовые
      1. ~ - побитовый унарный оператор NOT
      2. & — оператор побитовое И;
      3. | — оператор побитовое ИЛИ;
      4. ^ — оператор побитовое исключающее ИЛИ.
      5. >> - сдвиг вправо
      6. >>> - сдвиг вправо с заполнением нулями
      7. << - сдвиг влево
    * Приоритет
      1. 
18. **Укажите правила выполнения операций с плавающей точкой в Java (согласно стандарту IEEE754). Как определить, что результатом вычисления стала бесконечность или нечисло?**
    * +
    * -
    * \*
    * /
    * public static boolean isInfinite(double v)
    * Double.isNaN(a) (0.0/0.0)
19. **Что такое статический импорт, какие элементы можно импортировать при статическом импорте?**
    * Чтобы улучшить читабельность кода можно импортировать статические члены класса почти так же, как и обычные классы, и получить прямой доступ к статическим членам без указания имени класса. Для импорта используется оператор Java import static, после которого указывается полное имя класса и метод или переменная:
    * import static java.lang.Math.PI;
20. **Объясните работу операторов if, switch, while, do-while, for-each. Напишите корректные примеры этих операторов**
    * if (20 > 18) { System.out.println("20 is greater than 18");}
    * 
    * 
    * 
    * 
21. **Объясните работу оператора instanceof. Что будет 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.
    * False