## Контрольная работа №1 Красовского Егора

# Перечислите атрибуты доступа, объясните их действие.

Всего в Java есть четыре модификатора доступа:

* **private;**
* **protected;**
* **default (friendly);**
* **public.**

Модификатор доступа **private** является наиболее ограничивающим область видимости, а именно класса. Используется для “закрытия” доступа напрямую к данным и вспомогательных методов класса. Таким образом модификатор private используется для реализации инкапсуляции. Доступ к таким элементам возможен через public getter/setter. Кроме того, классы (не вложенные в другой класс) и интерфейсы не могут иметь модификатор доступа private.

Модификатор **protected** дает видимость внутри пакета, в котором он объявлен и в пределах всех классов-наследников класса, члены класса, объявленные как protected, в рамках пакета доступны как через наследование, так и через ссылку. Используется в тех случаях, когда нужно ограничить данные от изменения вне класса наследника и другого пакета. Например, данные банковских карт можно защитить таким образом. Так же используется для реализации наследования.

Модификатор **friendly** видим внутри всего пакета. Чаще всего используется для классов-утилит, по сути, служебные классы, которые реализуют вспомогательный функционал и в них нет необходимости во всех классах программы.

Модификатор **public** используется для “работы” с пользователем. Он виден внутри всей программы. Однако если метод/переменные объявлены как public, а класс default, то видимость этих данных будет ограничена видимостью класса. В public классах реализован интерфейс для взаимодействия класса с окружающей средой и/или управления работы с классом.

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

Пакет - группа классов, объединённых в одном пространстве имён. Позволяют организовать классы в логические наборы, а также избегание конфликта имен между классами и гарантировать однозначность. Названия пакетов соответствуют физической/логической структуре проекта, то есть организации каталогов, в которых находятся файлы с исходным кодом. Такая структура позволяет облегчить поддержку проекта и делает поиск и использование классов, интерфейсов, нумераций и аннотаций более удобным. Также Пакеты могут быть использованы как инкапсуляция данных или же для того, чтобы прятать данные. Все что нужно это добавить классы в пакет. После этого мы можем использовать в своей программе эти классы, используя директиву import. В Пакетах некоторые классы могут быть доступны из вне, а другие для внутреннего использования в пакете. Путь к файлам внутри проекта соответствует названию пакета этих файлов. Если для класса пакет не определен, то считается, что данный класс находится в пакете по умолчанию, который не имеет имени. По умолчанию импортируется текущий пакет и java.lang.

Правила наименования:

* Имена пакета написаны во всем нижнем регистре, чтобы избежать конфликта с именами классов или интерфейсов.
* Компании используют свое инвертированное имя Интернет-домена, чтобы начать их имена пакета — например, com.example.mypackage для названного пакета mypackage создаваемый программистом в example.com.
* Коллизии имени, которые происходят в пределах единственной компании, должны быть обработаны условно в пределах той компании, возможно включением области или названия проекта после названия компании (например, com.example.region.mypackage).
* Пакеты на языке самого Java начинаются java. или javax.

В некоторых случаях имя интернет-домена, возможно, не допустимое имя пакета. Это может произойти, если доменное имя содержит дефис или другой специальный символ, если имя пакета начинается с цифры или другого символа, который недопустим, чтобы использовать в качестве начала имени Java, или если имя пакета содержит зарезервированное ключевое слово Java, такое как "интервал". В этом случае предложенное соглашение состоит в том, чтобы добавить подчеркивание.

# 3. 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?

Импортировать в программу следует те классы, интерфейсы и перечисления, которые необходимы для решения поставленной задачи и/или в зависимости от предметной области. Для импортирования пакетов необходимо прописать оператор импорт import name.subname; где пакет name обозначает имя пакета верхнего уровня, а пакет subname - имя подчиненного пакета из внешнего пакета, отделяемое знаком точки (.). Глубина вложенности пакетов практически не ограничивается ничем, кроме файловой системы. Имя класса может быть задано явно или с помощью знака ‘\*’ который указывает компилятору Java на необходимость импорта всего пакета. Основные языковые средства хранятся в пакете java.lang, входящем в пакет java. Для импорта класса не обязательно импортировать пакет для этого достаточно написать имя пакета перед именем класса. Импортирование пакета позволяет повысить читабельность кода. При множественном импорте со \* можно столкнуться с проблемой что несколько классов имеют одинаковое название, но лежат в разных пакетах, тем самым компилятор не сможет определить конечный класс. Так же импорт пакета не влияет на импорт подпакетов, их необходимо импортировать отдельно. По умолчанию импортируется текущий пакет и java.land т.к там содержится все основные средства языка.

# 4.Объясните различия между терминами “объект” и “ссылка на объект”.

Ссылка — это число в 16-ричной системе исчисления, в которой записан адрес ячейки в памяти, где объект располагается. Если две ссылки содержат одно и то же число, то они ссылаются на один и тот же адрес памяти и, значит, на один и тот же объект. Null это предопределённое число, которое используется чтобы показать, что ссылка не ссылается на реальный объект. Действия над ссылками:

* Присваивание — позволяет определить новое значение ссылки;
* Обращение к полям и методам объекта;
* Операция приведения типа;
* Проверка принадлежности к определенному типу.

Объект — это конкретный экземпляр класса, который содержит: состояние - некоторые данные, которые хранит объект, и поведение - действия, которые может совершать объект.

При создании каждого объекта выделяется память при этом запускается конструктор класса. При создании ссылки конструктор не вызывается. Так же на 1 объект может быть несколько ссылок, на каждую ссылку приходится 1 объект. Объекты сохраняются в «куче», а ссылки на объекты сохраняются в стеке.

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

Не явное приведение – приведение типа, в котором меньшее значение присваивается к большему. Схема возможных преобразований byte-short-int-long, так же float-double и char-int. Пр. float a=0.5F; double b=a+0.7. При сложении char c int результат приводится к int, однако следует помнить, что будет произведено сложение кода символа char а не его символьного значения. Остальные приведения типов ведет к потери точности. Такое преобразование называется явным. Его необходимо указывать явно оператором(тип). Пр. int a=9999999; short ab=(short)a; в этом случае в a будет отрицательным числом т. к. будет выход за пределы диапазона значений short и в бит, отвечающий за знак запишется 1 (в 16-бит запишутся первые 16 бит 32 битного значения int). Явное приведение типа как бы снимает ответственность с компилятора за правильность приведения и перекладывает её на программиста.

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

В Java есть 8 примитивных типов, которые делят на 4 группы:

1. Целые числа - byte, short, int, long
2. Числа с плавающей точкой (вещественные) - float, double
3. Логический - boolean
4. Символьный – char

Целочисленные константы могут задаваться в программе одним из трех способов: в виде десятичных, шестнадцатеричных и восьмеричных значений. По умолчанию все числа интерпретируются как десятичные и относятся к типу int. Явно указать принадлежность к типу long можно, добавив в конце числа букву "l" или букву "L". Шестнадцатеричное значение задается с помощью символов "0x" или "0X", за которым значение числа (цифры 0-9 и буквы A-F или a-f), например: 0x7FFF. Число в восьмеричной записи должно начинаться с нуля, за которым следует одна или несколько восьмеричных цифр, например 077777. Восьмеричные и шестнадцатеричные числа могут быть как положительными, так и отрицательными и изменяются в тех же диапазонах, что и числа в десятичном представлении (например, шестнадцатеричные числа типа byte имеют максимальное значение 0x7F и минимальное значение – 0x80, а восьмеричные – соответственно 177 и – 200).

Вещественные типы переменных типы с плавающей точкой обычной и двойной разрядности – типы float и double. Их можно задать в экспоненциальной форме или указанием целой и дробной части. Реализованы по стандарту IEEE754. Данные примитивные типы используются в математике, хотя не рекомендованы для подсчета большого кол-ва не целых чисел и подсчёта крайне малых чисел из-за округления. Для этого есть такой класс как java.math.BigDecimal. При делении на 0 результат будет бесконечность, при сложении таких бесконечностей результат будет не числом.

Символы определяются с помощью ключевого слова char и реализованы с использованием стандарта Unicode. Можно задать константу-символ в про­грамме или как обычный символ. Символьное значение должны быть заключено в пару одиночных апострофов.

Переменная — это контейнер, со значением в нем. В переменной примитивных типов лежит значение. Поэтому при передаче в метод создается локальная копия значения в стеке метода. Копия уничтожается при уничтожении стека, где она хранится. В ссылочных типах данных, когда передается ссылка на объект в метод, передается копию ссылки в стек метода. Объект все еще сидит в куче, где был создан, ожидая того, что будут использованы ссылки на него. Поэтому при изменении объекта через ссылки его измененное состояние сохраняется. Объект уничтожается только тогда, когда кол-во ссылок на него становится равным 0.