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

Атрибуты доступа Java:

*Public* – доступ из любой части программы.

*Protected* – доступ из всего пакета, плюс возможен доступ из другого пакета при наследовании класса.

*Default (Friendly)* – доступ из всего пакета, доступ извне при помощи наследования невозможен. Если атрибут не указать – будет установлено это значение.

*Private* – доступ только внутри класса.

Атрибуты доступа используются для сохранения принципа ООП инкапсуляции, также может косвенно использоваться для простоты работы с чужим кодом, подразумевая что публичные методы (выставлены наружу) предназначены для использования, а приватные осуществляют внутреннюю логику.

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

Классы в Java программе объединяются в пакеты. Они позволяют организовать классы логически, сделать интуитивно-понятную архитектуру. Физически на диске они представлены в качестве папок.

Также помогает избежать дублей наименований классов. Полное имя класса = имя пакета+имя класса (например java.lang.String).

Существует соглашение о наименовании пакетов. Основные принципы:

- Именуются нижним регистром в единственном числе

- Начинается с домена, далее идет наименование компании, подразделение, имя проекта, далее подразделы. Например by.epam.homework.algoritm

Структура пакетов в проекте зависит от используемых паттернов, может быть логическая или событийная.

Итоговая вложенность рекомендуется от 4 до 8.

**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?**

Импорт классов позволяет «загрузить» их в класс, тем самым убрав необходимость писать полное имя класса. По умолчанию в каждый класс загружен пакет Java.lang.

Для множественного импорта можно использовать символ «\*». Например import java.util.\* импортирует весь пакет util. Можно использовать в небольших программах, однако если поставлена задача экономии памяти делать не рекомендуется.

Возможен также статический импорт для упрощения кода. Например static import Math. Упрощает код и вместо Math.sqrt(); достаточно указать просто sqrt();

Если импортированы пакеты с одинаковыми именами классов, необходимо будет указывать полное имя класса.

Также импорты могут использоваться для работы с чужим кодом: посмотрев какие пакеты были импортированны в класс, можно примерно определить что и как класс делает, какова его логика.

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

Cat **cat** *// создание ссылки типа Cat*

= **new** Cat(); *// создание объекта типа Cat (объект №1)*

Cat **cat1** *// создание ссылки типа Cat*

= **new** Cat(); *// создание объекта типа Cat (объект №2)*

cat = cat1; *// присваивание ссылке* ***cat*** *ссылку* ***cat1****. То есть теперь ссылка* ***cat*** *указывает на тот же объект 2. Теперь мы имеем два объекта и две ссылки, но обе ссылки указывают на объект №2. Первый объект не содержит на себя ссылок и будет убран GC при сборке мусора.*

cat.name = “vasek”;

cat1.name = “murzik”;

Теперь мы используя ссылки управляем объектом. Сначала объекту присваивается одно имя, потом – другое. В итоге объект №2 будет иметь имя “murzik”. То есть образно говоря мы имеем два «пульта», которые управляют одним объектом. Независимо от того, какую ссылку мы будем использовать, будет изменен все тот же объект.

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

Приведение типов происходит неявно от меньшего к большему и явно наоборот. Например:

**int** x = 10;  
**long** y = x;  
  
**long** t = 10;  
**int** d = (**int**)t;

Здесь все логично, т.к. если значение x поместилось в int, то логично что оно поместится и в более «широкий» тип, поэтому явное преобразование не требуется. Если происходит преобразование наоборот, из большего к меньшему, требуется явное указание.

Пример, когда это реально нужно. Получить случайное однозначное число:

**int** x = (**int**)(Math.*random*()\*10);

Приведение обязательно т.к. метод возвращает значение типа double, а нам (для примера) нужен int. Поэтому мы сначала получаем дробное число от 0 до 1, умножаем его на 10, далее приведением типа отбрасываем дробную часть и получаем нужный ответ.

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

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

**public class** Test {  
 **public static void** main(String[] args) {  
 **int** x = 2; // создание примитива типа int и присваивание ему значение 2  
 *method1*(x); // передача в метод **значение** примитива x.  
 System.***out***.println(x); // вывод примитива x; (будет ответ 2)  
 }  
  
 **private static void** method1(**int** x){  
 x++;  
 }  
}

Как видно в примере, примитивные типа в Java всегда передаются по значению. Ссылочные по факту тоже (передается копия ссылки).