Абстрактный класс.

Абстрактный класс – это такой тип данных, экземпляры которого запрещено создавать. Чтобы класс был абстрактным, следует перед ключевым словом class добавить abstract. АК могут содержать абстрактные методы. Они представляют собой заголовок метода без тела.

public abstract class MyClass {

protected abstract void method1{};

}

Таким образом, любой наследник АК обязан переопределить АМ и реализовать его тело. В реализации АМ запрещено вызывать метод суперкласса.

JNI(ознакомительно)

Нативные (native) методы также не содержат тело метода. В заголовок помещается слово native что означает что реализация метода находится в dll-библиотеке. Т.О. осуществляется связь Java и C++. Эта связь нужна для обращения к функциям ОС (обращение к файлам, получение событий от клавиатуры или мыши и т.д.). Нативные методы могут быть объявлены в любом классе, в т.ч. неабстрактном.

Интерфейсы.

Интерфейс – это АК, где все методы public и abstract. Объявление полей запрещено.

public interface Interface1 {

void method1{};

}

\*В Java 8 допускается создание методов с реализацией в интерфейсе. Однако, это не рекомендуется.

Использование интерфейсов допускает множественное наследование. Множественное наследование – это наследование от двух и более предков (в явном виде в Java запрещено.) Наследование от интерфейсов в Java именуется реализацией.

public class A extends B implements interface1, interface2, interface3 (){

}

Объект класса, реализующий интерфейс может быть присвоен интерфейсной переменной. У этой переменной можно будет вызвать только методы, объявленные в интерфейсе.

Interface1 i = new A();

Интерфейс может быть наследником одного или более интерфейсов. В этом случае объект, реализующий интерфейс, может быть присвоен переменной интерфейса предка.

Сравнение интерфейсов и АК.

Наследование от АК допускает внесение общего кода и полей в АК. Реализация интерфейсов допускает множественное наследование (программист может имплементировать сразу несколько интерфейсов).

Аннотации.

Аннотации – это методанные, которые помещаются в описание типа объекта – класс Class. В некоторые аннотации необходимо присваивать значения полям аннотации. В качестве значения могут быть примитивные типы, строки и enum. Для создания аннотации следует написать:

public @interface MyAnnotation {

}

Существует три типа аннотации.

1. Доступная компилятору (@override)
2. Доступная эдитору (@Supress warning)
3. Доступная при работе приложения

Помимо области действия аннотации также при объявлении мы должны указать к чему может применяться аннотация:

* тип (класс, интерфейс, enum),
* поле,
* метод,
* конструктор,
* переменная,
* параметр
* и т.д.

Для того чтобы указать область действия и объект применения аннотации следует аннотацию пометить специальными аннотациями. Разработка аннотаций обычно связана с созданиями библиотек фреймворков. Задача junior-программиста является применение этих аннотаций в разрабатываемой программе.