**Наследование**

Наследование — механизм языка, позволяющий описать новый класс на основе уже существующего (родительского, базового) класса или интерфейса. Потомок может добавить собственные методы и свойства, а также пользоваться родительскими методами и свойствами. Позволяет строить иерархии. Является одним из пяти типов ассоциации. Является одним из основных принципов объектно-ориентированного программирования.

Синтаксис наследования в Java:

class ClassName2 extends ClassName1

ClassName2 - наследник/потомок/производный

ClassName1 - родитель/предок/базовый

**См. файл Nasl.java**

При наследовании:

Потомки наследуют(приобретают) ВСЕ методы и атрибуты предка.

Методы и атрибуты могут быть переопределены.

В потомках можно добавлять свои методы и атрибуты.

**См. файл Nasl2.java**

**Инициализация базового класса**

Так как в наследовании участвуют как два класса (базовый и производный).

То очевидно, что часть относящаяся к базовому классу должна быть инициализирована.

Гарантировать правильную инициализацию можно только одним способом: вызвав конструктор базового класса. Java автоматически вставляет вызов конструктора базового класса в конструктор производного.

**См. файл Nasl3.java**

Что будет если нет конструктора по умолчанию (определён конструктор/ры с параметром, но не определён конструктор без параметра)?

В таком случае требуется в производном классе явно вызывать конструктор из базового класса.

**См. файл Nasl4.java**

**Спецификаторы доступа "инкапсуляция"**

Существуют спецификаторы доступа для методов и атрибутов класса. Спецификаторы доступа обычно пишут перед объявлением метода/атрибута.

public (по умолчанию) - метод/атрибут доступен всем пользователям класса

protected - метод/атрибут доступен ТОЛЬКО в иерархии наследования

private - метод/атрибут доступен только в самом классе, потомки и пользователи не имеют прав доступа.

**См файл Product.java, CD.java**