1. **Какие бывают внутренние классы?**

Вложенные классы делятся на два вида: статические и не статические. 

* Вложенные классы, объявленные как статические называются вложенными статическими (static nested classes).
* Вложенные не статические классы называются внутренними (inner classes).

**2. Во что компилируется анонимный внутренний класс?**  
Во внутренний не статический класс.

**3. Зачем использовать ключевое слово final при создании анонимных классов?**   
Если определяется анонимный внутренний класс и ему нужно при этом использовать объекты, определенные вне этого внутреннего класса, компилятор требует, чтобы переданные на них ссылки объявлялись неизменными (final). Без такого объявления вы получите сообщение об ошибке при компиляции программы.  
  
**4. Как правильно создать объект внутреннего класса?**  
Внутренние (не статические) классы, как переменные и методы связаны с объектом внешнего класса. Внутренние классы так же имеют прямой доступ к полям внешнего класса. Такие классы не могут содержать в себе статические методы и поля. Внутренние классы не могут существовать без экземпляра внешнего. Для создания объекта:

Outer outer = new Outer();

Innter inner = outer.new Inner();

**5. Как правильно создать объект вложенного класса?**  
Синтаксис создания объекта вложенного класса:

OuterClass.StaticNestedClass nestedObject = new OuterClass.StaticNestedClass();

**6. Можно ли создавать статические методы/переменные во внутреннем классе?**  
Статические методы/переменные объявлять во внутреннем классе (не вложенном) нельзя.  
Внутренние (не статические) классы, как переменные и методы связаны с объектом внешнего класса. Такие классы не могут содержать в себе статические методы и поля.  
  
**7. Назовите три любых внутренних класса?**

1. private static class Holder —вложенный класс HashMap из java.util.
2. В интерфейсе Map есть interface Entry<K,V>, который опять же в HashMap и реализуется в другом вложенном классе static class Entry<K,V> implements Map.Entry<K,V>.
3. private static class IntegerCache в классе Integer .

**8. Как внутренние классы решают проблему множественного наследования в Java?**  
Т.к. множественное наследование классов в Java запрещено, эту проблему решают с помощью внутренних классов: в нужном нам классе мы объявляем внутренний класс и наследуем его от требуемого класса. Пример:

class Tiger extends Cat {

public void tigerRun() { }

public void startTiger() {

TigerThread thread = new TigerThread();

thread.start();

}

class TigerThread extends Thread {

public void run() {

tigerRun();

}

}

}

**9. Чем отличаются анонимные классы, созданные на основе интерфейса и на основе класса?**  
Анонимный класс согласно JLS 15.9.5 представляют собой выражение, в котором объявление нового класса и его инициализация совмещены:  
  
При объявлении класса будет создан новый класс, производный от указанного класса при использовании в качестве базы другого класса, или реализующий интерфейс при использовании в качестве базы интерфейса.  
При инициализации будет создан новый объект и на него будет возвращения ссылка: анонимный класс является конкретным.

Таким образом, единственное отличие в анонимных классах, созданных на основе интерфейса и класса, заключается в количестве абстракных методов, которые необходимо реализовать.  
  
Инициализации анонимного класса на основе интерфейса потребует реализации каждого метода, в то время как при инициализации анонимного класса на основе абстрактного/конкретного класса позволит изменить поведение за счет перегрузки методов и потребует реализации абстракных методов.  
  
**10. Можно ли создать анонимный статический вложенный класс?**  
Нет, статической становится только переменная, но не класс.