1. **Опишите процедуру инициализации полей класса и полей экземпляра класса. Когда инициализируются поля класса, а когда – поля экземпляров класса. Какие значения присваиваются полям по умолчанию? Где еще в классе полям могут быть присвоены начальные значения?**

Поля класса могут инициализироваться в месте объявления и статическом блоке, а поля экземпляра класса — в месте объявления, нестатическом блоке и конструкторе. Инициализации полей класса происходит во время загрузки класса в порядке их объявления в классе. Инициализация полей объекта осуществляется при создании объекта в порядке объявления переменных; в нестатическом блоке инициализации происходит раньше, чем в конструкторе; сначала инициализируются все поля базового класса, а затем класса наследника. Поля класса инициализируются до создания объекта, поля экземпляра класса — после. Примитивным типам по умолчанию присваивается значение 0, boolean — false, ссылкам на объект — null.

2. **Дайте определение перегрузке методов. Как вы думаете, чем удобна перегрузка методов? Укажите, какие методы могут перегружаться, и какими методами они могут быть перегружены? Можно ли перегрузить методы в базовом и производном классах? Можно ли private метод базового класса перегрузить public методом производного? Можно ли перегрузить конструкторы, и можно ли при перегрузке конструкторов менять атрибуты доступа у конструкторов?**

Перегрузка методов — использование одноимённых методов с разными параметрами или типом возвращаемого значения. Удобство — выполнение одной и той же операции для разных данных.Статические методы могут перегружаться нестатическими и наоборот, без ограничений. private метод базового класса нельзя перегрузить public методом производного. Конструкторы можно перегружать, атрибуты доступа можно изменять.

3. **Объясните, что такое раннее и позднее связывание? Перегрузка – это раннее или позднее связывание? Объясните правила, которым следует компилятор при разрешении перегрузки; в том числе, если методы перегружаются примитивными типами, между которыми возможно неявное приведение или ссылочными типами, состоящими в иерархической связи.**

Ранее связывание — определения версии вызываемого метод на этапе компиляции. Позднее связывание — способность выбрать метод исходя из типа объекта во время выполнения программы. Перегрузка — раннее связывание. При нескольких перегруженных методах компилятор сначала ищет метод с точным совпадением параметров.

4. **Объясните, как вы понимаете, что такое неявная ссылка this? В каких методах эта ссылка присутствует, а в каких – нет, и почему?**

Ключевое слово this неявно передаётся во все методы, кроме статических, и может быть использовано для обращения к объекту, вызвавшему метод.

5. **Что такое финальные поля, какие поля можно объявить со спецификатором final? Где можно инициализировать финальные поля?**

Это константные поля. Они могут быть инициализированы при объявлении, в логическом блоке {} или в конструкторе, но только в одном из мест.

6. **Что такое статические поля, статические финальные поля и статические методы. К чему имеют доступ статические методы? Можно ли перегрузить и переопределить статические методы? Наследуются ли статические методы?**

Статические поля и методы относятся к классу, а не к объекту класса. Статические методы имеют доступ к статическим полям класса. Статические методы можно перегрузить, но не переопределить, так как статический метод относится к конкретному классу и вызывается через имя этого класса. Все доступные методы наследуются подклассами, в том числе и статические.

7. **Что такое логические и статические блоки инициализации? Сколько их может быть в классе, в каком порядке они могут быть размещены и в каком порядке вызываются?**

Логические и статические блоки используются для инициализации переменных и вызываются в порядке объявления.

8. **Что представляют собой методы с переменным числом параметров, как передаются параметры в такие методы и что представляет собой такой параметр в методе? Как осуществляется выбор подходящего метода, при использовании перегрузки для методов с переменным числом параметров?**

Это методы, которые могут принимать несколько параметров одного типа. На них существуют некоторые ограничения: с переменными нужно обращаться как с массивом, а также они должны идти последним параметром в методе. При использовании перегрузки методы с переменным числом параметров выбираются в последнюю очередь.

9. **Чем является класс Object? Перечислите известные вам методы класса Object, укажите их назначение.**

Класс Object является базовым классом для всех классов в иерархии Java.
Методы: Object clone() — создаёт новый объект, не отличающийся от клонируемого; (в таком случае в классе нужно реализовать интерфейс Cloneable и переопределять метод clone() (clone has protected access in class Object)). Второй способ копирования объекта — создание конструктора копирования (Effective Java);
boolean equale(Object obj) — определяет равенство объектов;
void finalize() — вызывается перед удалением неиспользуемого объекта;
Class<?> getClass() - получает класс объекта во время выполнения
int hashCode() - возвращает хеш-код, связанный с вызывающим объектом
void notify() - возобновляет выполнение потока, который ожидает вызывающего объекта
void notifyAll() - возобновляет выполнение всех потоков, которые ожидают вызывающего объекта
String toString() - возвращает строку, описывающий объект
void wait() - ожидает другого потока выполнения
void wait(long millis) - ожидает другого потока выполнения
void wait(long millis, int nanos) - ожидает другого потока выполнения.

10. **Что такое хэш-значение? Объясните, почему два разных объекта могут сгенерировать одинаковые хэш-коды?**

Хэш — это число, уникальное значение, которое JVM присваивает объекту. Хэш по умолчанию (идентификационный) использует целочисленное представление адреса памяти. Разные объекты не обязательно возвращают разный хэш, иначе получилась бы идеальная хэш-функция, которой не существует (всегда возможны коллизии). Но обратное верное: одинаковые объекты всегда возвращают одинаковые хэши.

11. **Что такое объект класса Class? Чем использование метода getClass() и последующего сравнения возвращенного значения с Type.class отличается от использования оператора instanceof?**

Объект класса Class — это объект, который создаётся JVM автоматически. Примитивные типы и ключевое слово void также представляются как объекты класса Class. instanceof определяет, был ли объект по ссылке, стоящей в левой части выражения, создан на основе класса, стоящего в правой части. getClass() проверяет, являются ли типы идентичными.

12. **Укажите правила переопределения методов equals(), hashCode() и toString().**

equals() — если ссылки указывают на один и тот же объект, возвращаем true; если одна из ссылок null или метод getClass() возвращает разные значения, возвращаем false; затем приводим сравниваемый объект к нужному типу и сравниваем значения всех полей.

hashCode() — при вычисления хеша следует использовать те же поля, которые сравниваются в equals (тогда получим равенство хешей для равных объектов).

toString() — public String toString(); для создания строки следует использовать использовать объект класса StringBuilder; при печати массива обращаться к элементам массива, а не печатать сам объект массива.