**1. Какие основные концепции объектно-ориентированного программирования (ООП)?**

**- Наследование: возможность наследовать данные из классов-родителей;** механизм, который позволяет описать новый класс на основе существующего (родительского). При этом свойства и функциональность родительского класса заимствуются новым классом;

**- Абстракция: описание лишь скелета объекта без реализации;** выделение главных, наиболее значимых характеристик предмета и наоборот — отбрасывание второстепенных, незначительных;

- **Инкапсуляция: защита данных от внешнего доступа к ним;** ограничение доступа к данным и возможностям их изменения;

- **Полиморфизм: переодределение данных;** это возможность работать с несколькими типами так, будто это один и тот же тип. При этом поведение объектов будет разным в зависимости от типа, к которому они принадлежат

**2. Что такое класс и объект в Java?**

Класс — это шаблон для объекта. Он определяет, как объект будет выглядеть и какими функциями обладать. Каждый объект является объектом (экземпляром) какого-то класса обладающий уже конкретными свойствами и поведением. Объекты создаются с использованием оператора new и инициализируются с помощью конструктора класса.

**3. Объясните понятие инкапсуляции и зачем оно нужно.**

Это ограничение доступа к данным и возможностям их изменения. Механизм инкапсуляции позволяет нам защитить некоторое поле при помощи сеттера (или получить при геттере), в который мы можем поместить проверку. Инкапсуляция работает при использовании модификаторов доступа private, public и т.д., и прочих инструментов

**4. Что такое наследование? Приведите пример его использования.**

Механизм, который позволяет описать новый класс на основе существующего (родительского). При этом свойства и функциональность родительского класса заимствуются новым классом;

При желании описать животных абстрактно можно сказать, что у них есть некоторые характерные признаки (размера, умственных способностей и пр.). Животным также присущи определенные поведенческие аспекты: они едят, дышат и спят. Такое описание характерных признаков и поведения является определением класса для животных.

**5. Что такое полиморфизм? Как он реализуется в Java?**

Это возможность работать с несколькими типами так, будто это один и тот же тип. При этом поведение объектов будет разным в зависимости от типа, к которому они принадлежат.

Пример: создаем некоторый класс с методом, затем создаем еще один класс, который наследует от прошлого класса все показания, и внутри него с помощью @override переназначаем какой-то метод, например с дефолтного «Привет, я – человек», на – «Здравствуй!»

**6. Чем отличаются примитивные типы данных от ссылочных?**

Примитивные типы данных — это основные типы данных, которые существуют в языке Java. Они используются для представления простых значений, таких как числа, символы и логические значения (Boolean, int, long, float, double)

Ссылочный тип данных используется для хранения ссылок на объекты, а не фактического значения объекта. Ссылочная переменная содержит адрес памяти объекта в Java, наиболее распространенным ссылочным типом данных является объект.

**7. Какие основные типы данных существуют в Java?**

Boolean, int, long, float, double, short, char, Массив, Строка, Объект

**8. Что такое оператор instanceof и для чего он используется?**

Instanceof – “Объект создан на основе класса и/или на основе предков?” – проверяет, является ли созданный объект потомком некоторого класса?

Синтаксис: <объект X> instanceof <объект Y>

**9. В чем разница между == и .equals()?**

== - сравнивает ссылки на объекты (для примитивных типов данных сравнивается значение)

.equals() - для логического сравнения содержимого объектов. По умолчанию это то же, что и «==», но многие классы (например, String, Integer, Date и другие) переопределяют этот метод, чтобы сравнивать содержимое объектов, а не их ссылки

**10. Что такое null в Java?**

Это специальное значение, которое может быть присвоено любому ссылочному типу. Первоначально он был изобретен для обозначения отсутствия чего-либо, например, отсутствие пользователя, ресурса или чего угодно.

Пишется нижним регистром: null; null это значение по умолчанию любых ссылочных типов; null – это просто специальное значение (по факту ничто). null может быть назначен только ссылочному типу. Любой класс-обертка со значением null будет выбрасывать java.lang.NullPointerException когда Java распакует(unbox) его в примитивную переменную.

**11. Какие значения по умолчанию присваиваются переменным в Java?**

- boolean: false

- byte: 0

- char: '\u0000' (нулевой символ)

- short: 0

- int: 0

- long: 0L

- float: 0.0f

- double: 0.0

Для всех ссылочных типов (например, объекты, массивы и т.д.) значение по умолчанию - null

**12. Объясните, как работает сборщик мусора (Garbage Collector) в Java.**

Сборщик мусора автоматически освобождает память, удаляя объекты, которые больше не используются программой.

Процесс сборки мусора состоит из двух основных этапов:

- Маркировка (Marking). Сборщик мусора определяет, какие объекты являются достижимыми и какие нет. Он проходит по дереву ссылок, начиная с корневых ссылок, и маркирует все достижимые объекты.

- Очистка (Sweeping). На этом этапе сборщик мусора удаляет недостижимые объекты и освобождает память, занятую ими. В зависимости от алгоритма сборки мусора, это может включать перемещение достижимых объектов и сжатие памяти для более эффективного использования.

**13. Что такое массив и как он объявляется в Java?**

Массив – это структура данных, в которой хранятся элементы одного типа. Его можно представить, как набор пронумерованных ячеек. Способы объявления:

Int[] a = new int[]

Int a[] = new int[]

Int[] a = {значения через запятую}

**14. Объясните разницу между статическими и нестатическими методами.**

Статические методы принадлежат классу. Они объявляются с использованием ключевого слова static. Такие методы могут быть вызваны без создания экземпляра класса и обычно используются для выполнения операций, которые не зависят от состояния конкретного объекта класса. К ним можно обращаться через имя класса, а не через объект класса.

Нестатические методы принадлежат отдельным объектам (экземплярам класса). Они могут иметь доступ к переменным экземпляра и изменять их состояние. Для вызова нестатического метода обычно требуется создать экземпляр класса

**15. Что такое перегрузка методов?**

Перегрузка метода означает создание другого метода с тем же именем в том же классе, но с другим списком параметров.

**16. Чем отличается String, StringBuilder, и StringBuffer?**

String — неизменяемый класс, его объекты сохраняют строку в виде неизменяемого массива байт. При изменении объекта String каждый раз создаётся совершенно новый объект.

StringBuilder создаёт мутабельный (изменяемый) набор символов. Класс не обеспечивает синхронизацию: экземпляры класса StringBuilder не могут совместно использоваться несколькими потоками.

StringBuffer создаёт изменяемый строковый объект. Класс потокобезопасный и синхронизированный: экземпляры класса StringBuffer могут совместно использоваться несколькими потоками.

**17. Что такое интерфейс? В чем его отличие от абстрактного класса?**

Интерфейсы описывают только часть функциональности объекта (поведение) — определённые признаки. Абстрактный класс же может описывать целую категорию разных объектов, а его характеристики имеют право наследовать только те объекты, которые являются частью этой категории. Например, собаки и волки — часть общей категории «Животные», а интерфейс, описывающий умение бегать, может реализовать и человек, и робот, и собака

**18. Как работают циклы for, while, и do-while в Java?**

for (int I – 0; I < n; i++) { } while (условие) { } do { выполняется и далее если удовлетворяет условию, то это выполняется повторно } while (условие)

**19. Объясните, как работает оператор switch и какие типы он поддерживает.**

switch (ВыражениеДляВыбора) {

case (Значение1):

Код1;

break;

default:

Код;

break;

byte, short, char, int, обертки: Byte, Short, Character, Integer; String и перечисление (Enum)

**20. Что такое исключения (exceptions) и как они обрабатываются?**

Исключения – это возникновение ошибок и непредвиденных ситуаций при выполнении программы;

Обрабатываются при помощи ключевых слов try, catch, finally, throws;

try { обработка кода, в котором может выйти исключение } catch { происходит обработка исключения } finally { необязательный блок, но при его наличии в нем выполнится обработка другого кода в не зависимости от того, что произошло в try };

throw – используется для возбуждения исключения;

throw new MyException("Some error")

throws – используется в сигнатуре методов для предупреждения, о том что метод может выбросить исключение;

**21. Для чего используются блоки try, catch, finally?**

try { обработка кода, в котором может выйти исключение } catch { происходит обработка исключения } finally { необязательный блок, но при его наличии в нем выполнится обработка другого кода в не зависимости от того, что произошло в try };

**22. Что такое аннотации в Java? Приведите примеры часто используемых аннотаций.**

Аннотации в Java — это специальные формы синтаксических метаданных, которые могут быть добавлены в исходный код. Они служат для определения дополнительной информации для классов, интерфейсов, методов или полей

Примеры:

@Override – переназначение методов;

@Deprecated – отмечает, что метод устарел и не рекомендуется к использованию;

@SuppressWarnings – указывает компилятору подавить предупреждения компиляции, определённые в параметрах аннотации;

@SafeVarargs – указывает, что никакие небезопасные действия, связанные с параметром переменного количества аргументов, недопустимы;

**23. Модификаторы доступа — это чаще всего ключевые слова, которые регулируют уровень** доступа к разным частям твоего кода

private - ограничивает видимость данных и методов пределами одного класса (реализует Инкапсуляцию)

default (package visible) – по дефолту устанавливается для всех полей и методов, если внутри пакета модификатор для класса не писать, то он будет иметь доступ «только в пакете» (default (package visible))

protected – поля и методы будут видны в пределах всех классов, находящихся в одном пакете, а также буду видны в пределах всех классов-наследников данного класса

public – модификатор публичного доступа к методам и полям

**24. Что такое перегрузка операторов и поддерживается ли она в Java?**

Перегрузка операторов — это возможность выполнять разные операции в зависимости от контекста. Она позволяет операторам действовать по-разному в зависимости от операндов.

В Java перегрузка операторов не поддерживается. 23 В языке поддерживается только ограниченный и фиксированный набор перегруженных операторов, которые нельзя изменить или добавить программисту. Для пользовательских типов (например, классов) нет встроенной поддержки перегрузки операторов

**25. Как работает оператор this?**

Используется для ссылки на текущий объект

**26. Что такое пакет (package) и зачем он нужен?**

Это группа связанных классов и интерфейсов, организованных в древовидной структуре. Они помогают управлять пространством имён, обеспечивая структурирование и упорядочивание кода

Нужны для:

- Избежание конфликтов имён;

- Обеспечение контролируемого доступа;

- Инкапсуляция данных;

**27. JVM – это виртуальная машина, которая выполняет байт-коды Java. JVM не понимает исходный**

код Java, поэтому нужно скомпилировать \*.java файлы, чтобы получить \*.class файлы, содержащие байт-коды, понятные JVM;

Среда выполнения Java (JRE) предоставляет библиотеки, виртуальную машину Java и другие компоненты для запуска апплетов и приложений, написанных на языке программирования Java;

JDK — это расширенная версия JRE, которая содержит всё, что есть в JRE, а также такие инструменты, как компиляторы и отладчики, необходимые для разработки апплетов и приложений

**28. Как компилируется и выполняется Java-программа?**

Компиляция Java-программы происходит следующим образом:

- Исходный код программы (файл с расширением .java) преобразуется компилятором в байт-код. На выходе компилятор выдаёт файл с расширением .class, содержащий байт-код, необходимый для исполнения программы виртуальной машиной;

- Виртуальная машина Java (JVM) интерпретирует получившийся байт-код. Она построчно преобразует байт-код в машинный код и тут же его исполняет

**29. Что такое ключевое слово final и где оно может быть использовано?**

Для класса это означает, что класс не сможет иметь подклассов, т.е. запрещено наследование. Это полезно при создании неизменяемых объектов.

Для метода final означает, что он не может быть переопределен в подклассах. Это полезно, когда мы хотим, чтобы исходную реализацию нельзя было переопределить.

Для переменных примитивного типа это означает, что однажды присвоенное значение не может быть изменено.

Для ссылочных переменных это означает, что после присвоения объекта, нельзя изменить ссылку на данный объект. Ссылку изменить нельзя, но состояние объекта изменять можно.

**30. Что такое Lambda-выражения в Java и зачем они используются?**

Lambda-выражения – это анонимные функции. Проще говоря, это метод без объявления, т.е. без модификаторов доступа, возвращающие значение и имя. Короче говоря, они позволяют написать метод и сразу же использовать его. Особенно полезно в случае однократного вызова метода, т.к. сокращает время на объявление и написание метода без необходимости создавать класс.

Lambda-выражения в Java обычно имеют следующий синтаксис (аргументы) -> (тело)