1. **What is encapsulation**

Encapsulation is one of the core concept in the object oriented programing language the purpose of the encapsulation is binding the data and code in a single unit and restricting the direct access to the internal data from the outside class. It is mainly used for restrict the unauthorized access, Access provided through the public getters and setters

1. **What is inheritance**

Inheritance is one of the principal in the oops, Here the child class acquire the properties and functions from the parent class. It is mainly used to reuse the code instand of writing repeting the code. And it makes an relation between parent and child class by the extends key word.

1. **What is interface**

An **interface** is a **contract** or **blueprint** that defines **abstract methods** without implementation.

1. **What is Polymorphism and types**

Polymorphism is also one of the principle in Oops, Polymorphism is refer as poly means many and morphism is forms. it means you can use the same name for different things

We have two types of polymorphism

1. Run time polymorphism – method overriding
2. Compile timer polymorphism - method overloading
3. **What is difference between abstract and interface**

An **abstract class** allows both **implemented and unimplemented methods**, and is inherited using the extends keyword, And **interface** contains only **unimplemented methods** by default, and a class uses the implements keyword to follow it.

1. **What is the difference between the overriding and overloading**

Overloading can happen in the same class but overriding will not happens in the same class

Overloading we have same method name but different parameters but in overloading have same method name and same parameters but different functionality

1. **How to create a class and how to create an object**

Class is a blue print of data it defines its own properties and functions.

object is an instance of a class when you create an object, you use the class blueprint to make something you can use in your program.

Class creation:

Public Class{

Access modifiers - datatype - Varibales;

Access modifiers – return type – methodname(){};

}

Object Creation:

ClassName obj = new ClassName();

**ClassName** is the class.

**obj** is the reference variable that points to the object.

**new** is a keyword that creates a new object in memory.

**ClassName()** is the constructor that initializes the object.

1. **What is jvm and its functionality**

JVM - java virtual machine is a component that provide the run time environment for executing the bytecode and making java programs platform independent. It generally acts like an translator which translates the byte code to machine level code to which the operation system can understand

Functionality of the JVM is mainly

Run java code

Makes java platform independent

Manages memory

Handles errors

1. **Java memory management**

Java memory management involves both automatic and manual processes, and the JVM is responsible for managing this

When an object is created using the new keyword, it is stored in the heapmemory.

The references to these objects are stored in the stackmemory.

Java has a special process called the GarbageCollector that automatically finds and removes objects that are no longer being used, freeing up memory.

1. **Difference between stack memory and heap memory**

**Stack memory** stores data like local variables and method calls.

**Heap memory** stores objects that are created using the new keyword.

Stack memory is managed automatically in a last-in, first-out (LIFO) order.

Heap memory is managed by the **Garbage Collector**, which frees unused objects.

Stack memory provides faster access compared to heap memory.

Stack memory holds primitive data types and reference variables.

Heap memory holds the actual objects and their data.

1. **What are collections**

Collections are dynamic data structures used to store, manage, and manipulate data. Unlike arrays, where the size is fixed and permanent, collections allow resizing. Another advantage of collections is that we can store different data types, whereas arrays are restricted to only one data type. Java provides a Collection Framework that includes different interfaces and classes to work with collections efficiently.

1. **What is the use of list, set, hashmap**

List, set, and hashmap are collections where they have their own uniqueness. By using these, we can store, manage, and manipulate data. In a list, the data is stored in order and duplicates are allowed. Coming to set, duplicates are not allowed and it only stores unique items. When it comes to hashmap, the data is stored in the form of key and value pairs. These collections help in organizing data based on different requirements.

1. **What is an exception and types**

Exception is used to avoid crashing the program. Instead of crashing, the program will return a message like **NullPointerException** or **ArithmeticException**. There are two types of exceptions: **checked exceptions** and **unchecked exceptions**. Checked exceptions are like **IOException** and **SQLException**, and unchecked exceptions are **NullPointerException**, **ArrayIndexOutOfBoundsException**, and **user-defined exceptions**. Exceptions can be handled using **try-catch** blocks or by using the **throws** keyword.

1. **How to create an custom exception**

Custom exceptions in Java, also known as user-defined exceptions, are classes created by developers to handle specific error.

How to create:

Create a new class that extends Exception (for checked exceptions) or RuntimeException (for unchecked exceptions).

Provide constructors to initialize the exception with custom messages.

Add methods to provide additional details about the exception.

1. **What is exception hierarchy**

In Java, all errors and exceptions are organized in a **family tree-like structure**, which is called the **Exception Hierarchy**. This helps Java understand how to deal with different types of problems that may happen when a program runs.

1. What is difference between try catch, try multi catch, try multiple catch, try finally, try catch finally, try with resource
2. How to create custom exception with checked and unchecked types
3. What is the use of stream api
4. Explain immediate operation in stream
5. Explain terminal operations
6. What is hashcode and equals
7. What is hash collision and how its is resolved
8. How to order collection
9. What is comparator and comparable
10. What is functional interface how to use it
11. What are different lamada expressions
12. What is difference between throw, throws, throwable

Codes:

1. Create a class and object
2. Create parent and child class and implement polymorphism
3. Create private and protected variables and use them
4. Show the relations of One parent two child, two parent single child
5. Show examples null point exception and array out of bond exception
6. Create employee class employee id, name, salary, using streams api print employee salary garter then 1000
7. Convert lowercase to uppercase using steams api
8. Create custom exception username invalid handle it via throw throws and try catch