Java is a high-level, class-based, object-oriented programming language

In 1990 there is a need of platform independent language, at that time c and c++ was popular. In 1991 James Gosling and his team invented Java firstly named was **oak** and then named java in 1995, is platform-independent because it is compiled to a **bytecode** that can be run on any device that has a Java Virtual Machine (JVM). This means that you can write a Java program on one platform (such as Windows) and then run it on a different platform (such as macOS or Linux) without making any changes to the code.

* Syntax is similar to c and c++
* Automatically memory management (Java's runtime system automatically handles the allocation and deallocation of memory for objects)
* Write once run everywhere
* C and c++ code compile in machine language but its code is compile in bytecode

**It is used for:**

* Mobile applications (specially Android apps)
* Desktop applications
* Web applications
* Web servers and application servers
* Games
* Database connection

## **Why Use Java?**

* Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
* It is one of the most popular programming languages in the world
* It is open-source and free
* It is secure, fast and powerful
* Java is an object oriented language which gives a clear structure to programs and allows code to be reused

What are the differences between C++ and Java?

C++ is compiled and run using the compiler which converts source code into machine code so, C++ is platform dependent. Java uses both compiler and interpreter. Java source code is converted into bytecode at compilation time. The interpreter executes this bytecode at runtime and produces output

Why Java is not a pure object oriented language?

Java is not considered a pure object-oriented programming language. The main reason is it supports primitive type values. For an object-oriented programming language, data should be represented in the form of objects. As Java uses primitive data types, it is not considered a pure object-oriented programming language. (Java includes eight primitive data types (int, byte, short, long, float, double, boolean, and char) which are not objects. This allows for more efficient memory usage and performance but breaks the pure object-oriented paradigm where everything should be an object.)

List the features of the Java Programming language?

* Simple.
* Object-Oriented.
* Platform Independent.
* Portable. (Java programs can execute in any environment for which there is a Java run-time system)
* Secure.
* Interpreted.
* Multi-Threaded.

Class: A class is a blueprint or template that defines the properties (attributes) and behaviors (methods) of objects.

**Parameters:**

* **Definition:** Variables listed inside the parentheses in the function definition.

void printSum(int a, int b) { // a and b are parameters

**Arguments:**

* **Definition:** The actual values passed to the function when it is called.

printSum(7, 8); // 7 and 8 are arguments

**Abstraction:**

1. **Definition**: Abstraction is the process of hiding the complex implementation details and showing only the essential features of an object or system.
2. **Purpose**: It focuses on what an object does rather than how it does it.

**Encapsulation:**

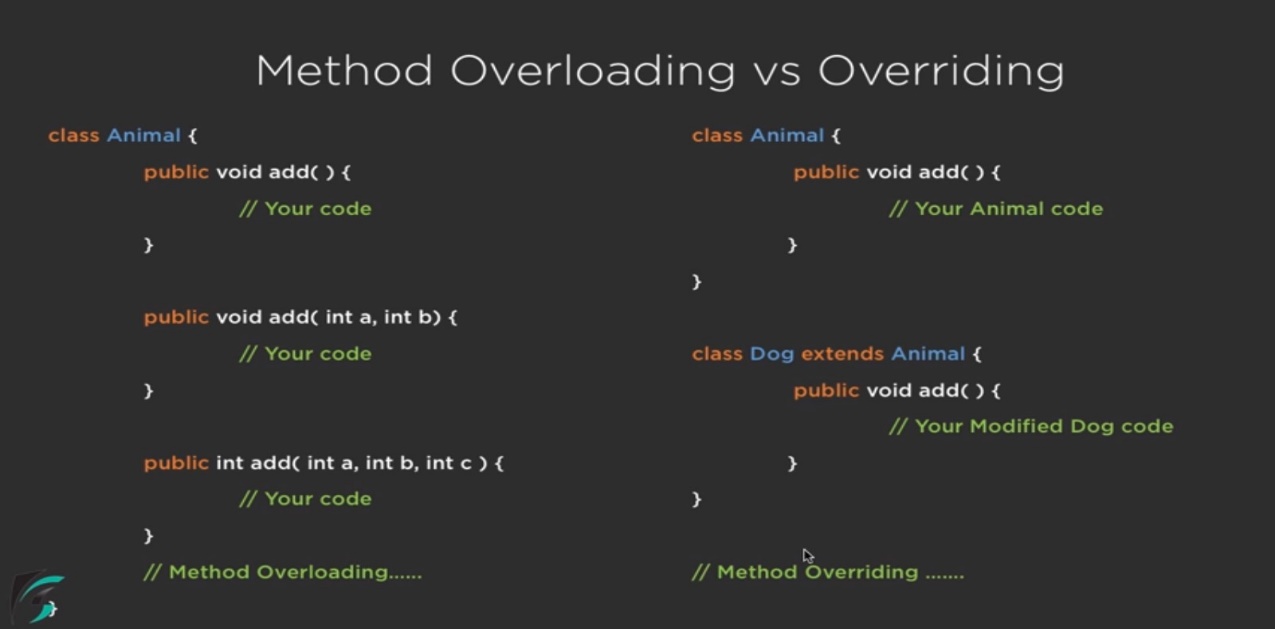
1. **Definition**: Encapsulation is the technique of wrapping the data (variables) and code (methods) together as a single unit and restricting direct access to some of an object's components.
2. **Purpose**: It focuses on protecting the object's state from unauthorized access and modification by exposing only necessary parts through public methods.

**Method overloading:**

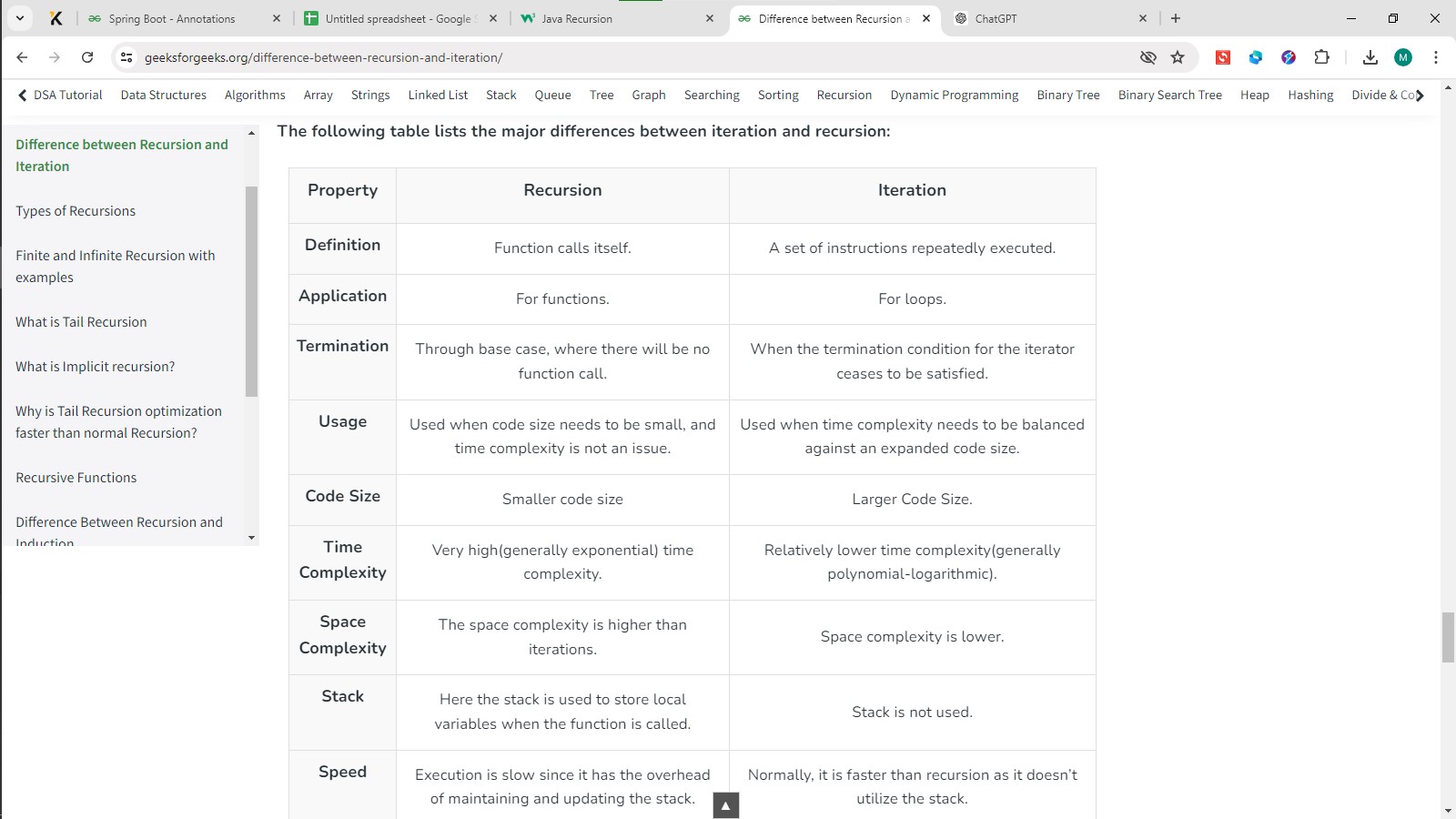
Method overloading in Java occurs when two or more methods in the same class have the exact same name, but different parameters, Methods can have different return types, but this alone is not sufficient for overloading.

**Method Overriding:**

Occurs in TWO classes: Super class and sub class i.e. Inheritance is involved. Name and Parameters both are same. Use the method in the Child class which is already present in Parent class. Return type is always same. It is an example of Run time Polymorphism.

****

**Recursion vs iteration:**

****

## **Halting Condition**

Just as loops can run into the problem of infinite looping, recursive functions can run into the problem of infinite recursion. Infinite recursion is when the function never stops calling itself. Every recursive function should have a halting condition, which is the condition where the function stops calling itself.