**Garbage Collection is process of reclaiming the runtime unused memory automatically. In other words, it is a way to destroy the unused objects.**

**Features:**

* It makes java **memory efficient** because garbage collector removes the unreferenced objects from heap memory.
* It is **automatically done** by the garbage collector (a part of JVM) so we don't need to make extra efforts.
* In the **Sweep** stage, JVM performs memory reclamation and garbage collection algorithms.
* Example for calling a GC
* Runtime runtime = Runtime.getRuntime();
* runtime.gc ();

But it is not needed, jvm will automatically handle correct timely running of GC.

1. Why Garbage Collection is necessary in Java?

In programming languages such as C and C++, the developer programmatically reclaims the space that is allocated to an object in the memory. In Java programming, the user is not responsible for managing the space memory used by the objects. In JVM, a garbage collection routine is added as its part, which is responsible for identifying and deleting objects that are no longer in use in memory.

### Explain the use of the Finalize () method of the garbage collector.

### The ****Finalize ()**** method is called by the garbage collector before collecting any object that is eligible for the garbage collector. The ****Finalize ()**** method is used to get the last chance to object to cleaning and free remaining resource.

### Can we force the Garbage collector to run at any time?

### No, we cannot force Garbage collection in Java. Although, we can request it by calling ****system.gc ()**** or its cousin ****Runtime.getRunitime(). get()****. It's not guaranteed that GC will run immediately as a result of calling these methods.

### Does Garbage collection occur in permanent generation space in JVM?

### Yes, Garbage collection can occur in PermGen space. If PermGen cross a threshold or it is full, it can trigger Full Garbage Collector.

### What do you mean by mark-and-sweep?

**Mark** and **Sweep** are the two states of garbage collection. In the **Mark** stage, JVM identifies whether an object is still needed or not. The object is marked for garbage collection when the object is not needed.