# Memory management:

- Java uses an automatic memory management system called a garbage collector. Thus, we are not required to implement memory management logic in our application.

## 1. Heap Memory

- Heap space is used for the dynamic memory allocation of Java objects and JRE classes at runtime.

- Dynamic memory is allocated with the help of 'new' keyword.

- The JVM automatically invokes garbage collections in order to clean up the heap of unreferenced or dead objects.

- If heap space is full, Java throws java.lang.OutOfMemoryError.

- Heap consist of 3 parts:

1. Young generation - It contains newly created objects.

2. Old generation - It contains those objects that passed the certain threshold of garbage collection in the young generation.

3. Perm generation - It contains metadata of loaded class and stores all the static content.

## 2. Stack Memory

- A stack is created at the same time when a thread is created and is used to store data and partial results which will be needed while returning value for method and performing dynamic linking.

- It's accessed using Last-in First-out (LIFO) memory allocation system.

- If this memory is full, Java throws java.lang.StackOverFlowError.

- Stack memory only exists as long as the current method is running.

- main method is very first pushed inside the main stack.

- Each stack may consist of multiple frames, every time a method gets called a new frame is inserted to Stack.

-

## 3. String Pool

* The String Pool is a special area in memory where JVM stores string literals to conserve memory and optimize performance by reusing identical string values.
* Immutability of String is achieved through the use of String Constant Pool in the heap.
* Before *Java 7*, the JVM placed the Java String Pool in the *PermGen* space, which has a fixed size — it can't be expanded at runtime and is not eligible for garbage collection. The risk of interning *Strings* in the *PermGen* (instead of the *Heap*) is that we can get an ‘*OutOfMemory error’* from the JVM if we intern too many *Strings*.

## 4. MetaSpace

* It stores class metadata, such as class definitions, methods, and fields, as well as interned strings, in native memory rather than in the JVM heap.
* Metaspace automatically manages memory allocation for class metadata, expanding and contracting as needed, avoiding common PermGen-related issues like OutOfMemoryErrors due to class loading.

Indexing is a procedure of creating an index table for database records to make data accessing faster.