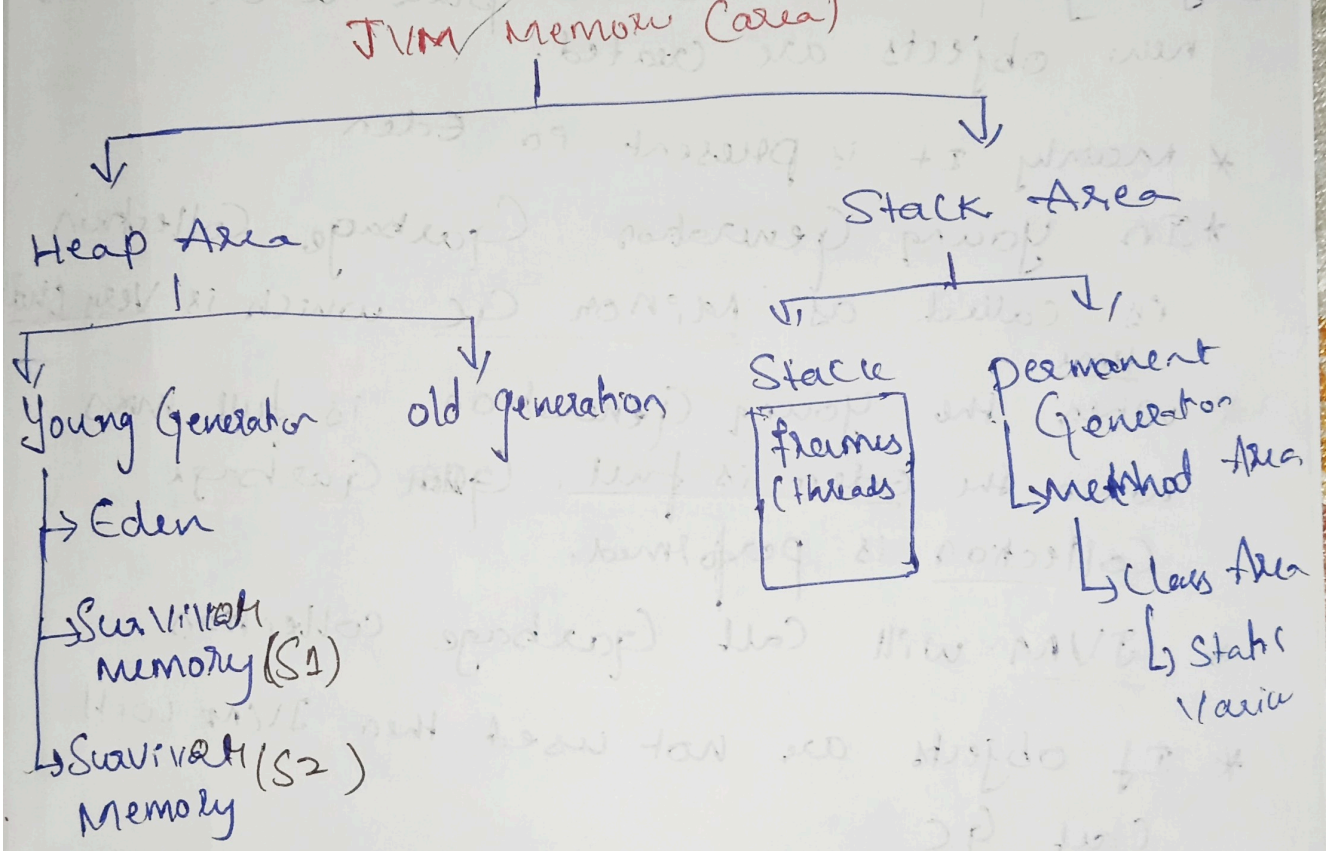
Java Memory Management / JVM Architecture

There are two memory area present in Java

1) Stack area 2) Heap Area



1> Heap Area

* All the object are created on the Heap area
* objects are stored/created into Eden.
* these objects are shifted to Survivor Memory S1 & S2 when they are not used.

Garbage Collection: Java Garbage collection is the process to identify and remove the unused objects from memory and free space to be allocated to objects

1) Young Generation

* This Contains 3 types
  1) Eden
  2) Survival memory (S1)
  3) Survivals Memory (S2)

~~Eden~~

* Young Generation is the place where all the new objects are created.
* Mainly it is present in Eden
* In Young Generation Garbage Collection is called as Minor GC which is Very Short time
* When the Young Generation is full means when the Eden is full, Garbage Collection is performed.
* JVM will Call Garbage Collection
* If objects are not used then JVM will Call GC
* GC is called to Erase Dereference Values

* **Dereference:**

  Ex: Transport trans = new Transport();
  trans = null;

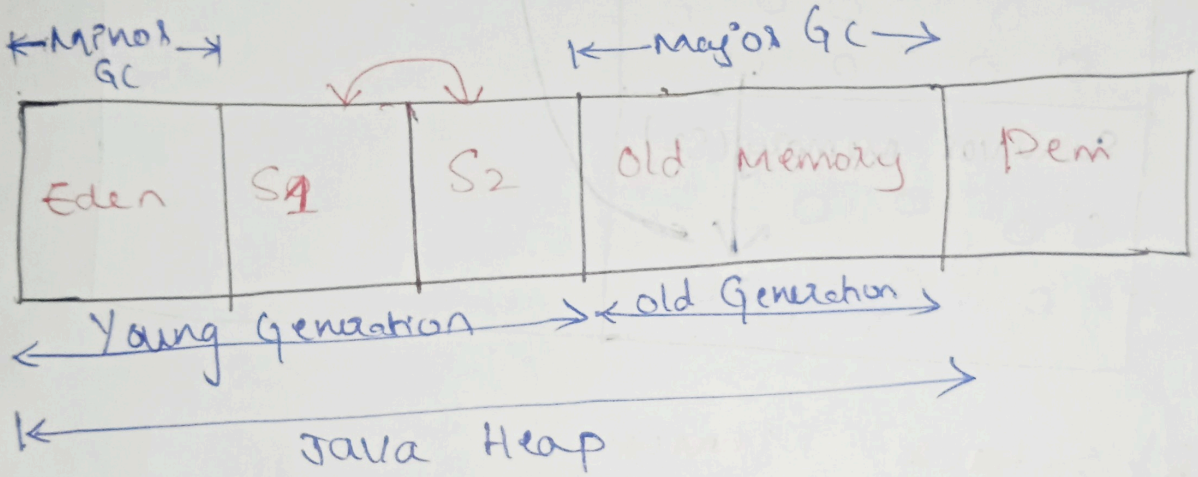
  *Now trans is an Dereference Value hence this will be erased by GC

* Dereference values are the values which does not contain any information and it occupies space & no use

* Once these values are identified by JVM. JVM will push these into Survivor memory from Eden

* Minor GC also checks the Survivor objects and move them to the other Survivor space. So at a time, one of the Survivor Space is always Empty.

* Non premitive data types are stored in Heap



2) old Generation

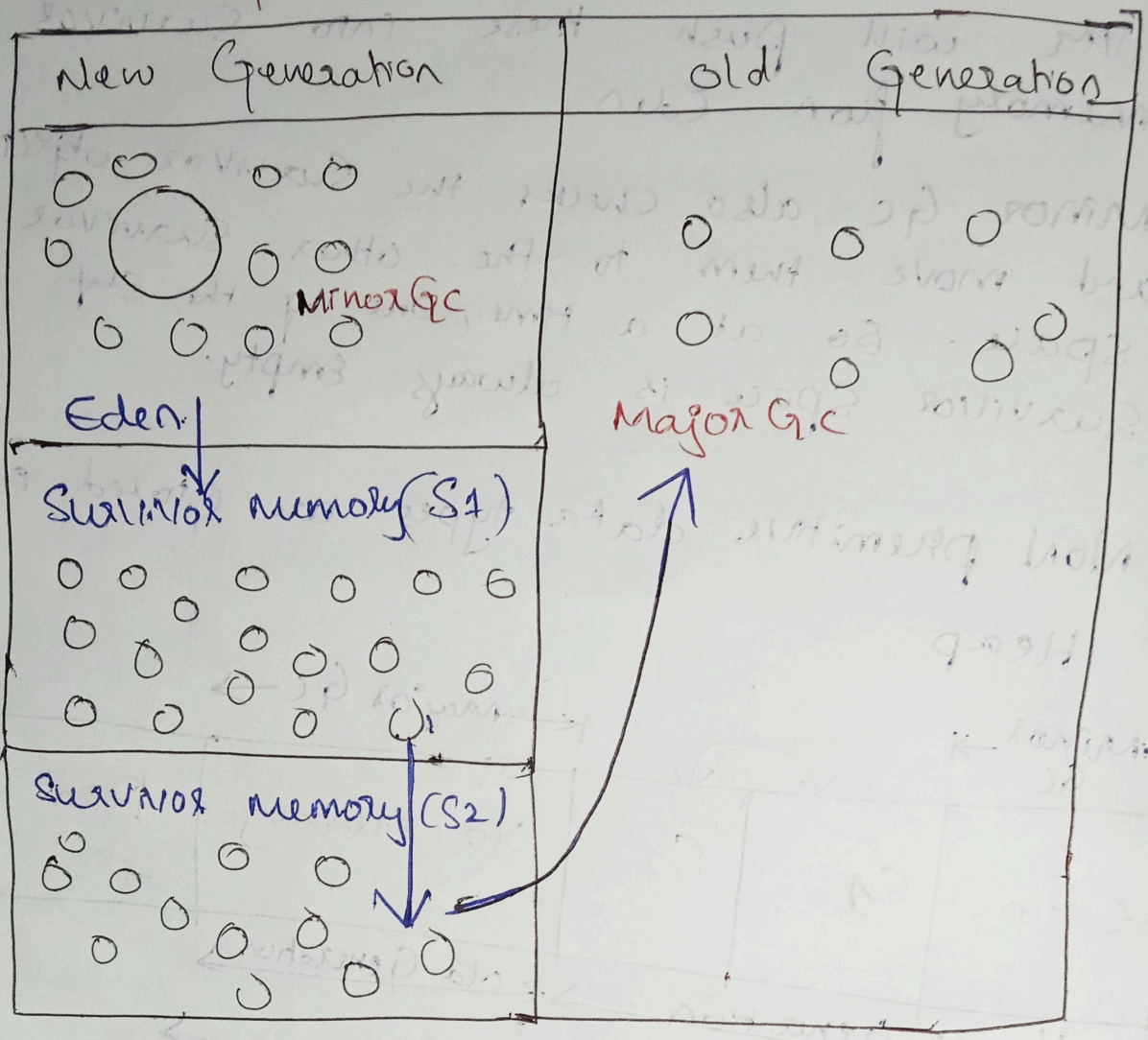
* objects that are survived after many cycles of GC, are moved to the old Generation memory space.

* old Generation memory contains the objects that are long-lived and survived after many rounds of minor GC.

* Garbage collection is performed in old Generation memory when it's full.

* old Generation Garbage collection is called as Major GC and usually take a longer time.

Heap

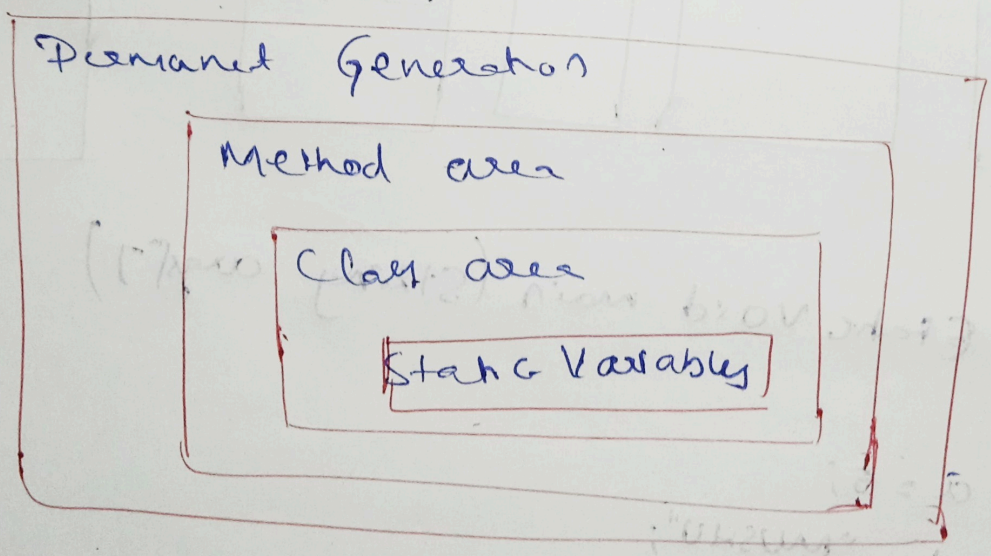


2) Stack Memory :

* Reference Variable or local Variable are created in Stack is →
* Stack ~~has~~ contain frames
* For Each information, one frame is Created
* Stack ~~memory~~ memory in Java is used for Static Memory allocation.
* Access to this memory is ~~is~~ Last-in-first-out (LIFO).

*

* Whenever a new Method is Called a new block on top of the Stack is Created which Contains values specific to that method, like Primitive Variables and Reference to objects
* It Contains permanent Generation in which Method area which Contains Class Area in which Static Variable are Stored
* Variables inside Stack Exist only as long as the Method that Created them is Running.
* Access to this memory is fast when Compared to heap memory.
* primitive data types are Stored in Stack.
* Reference Variable are Created in Stack
* Method Area which actually has Class area.
* This Method Area is present in Permanent Generation



NOTE: In Java String is a class.

* At the End of main '}' method the information under Stack will be Erased.

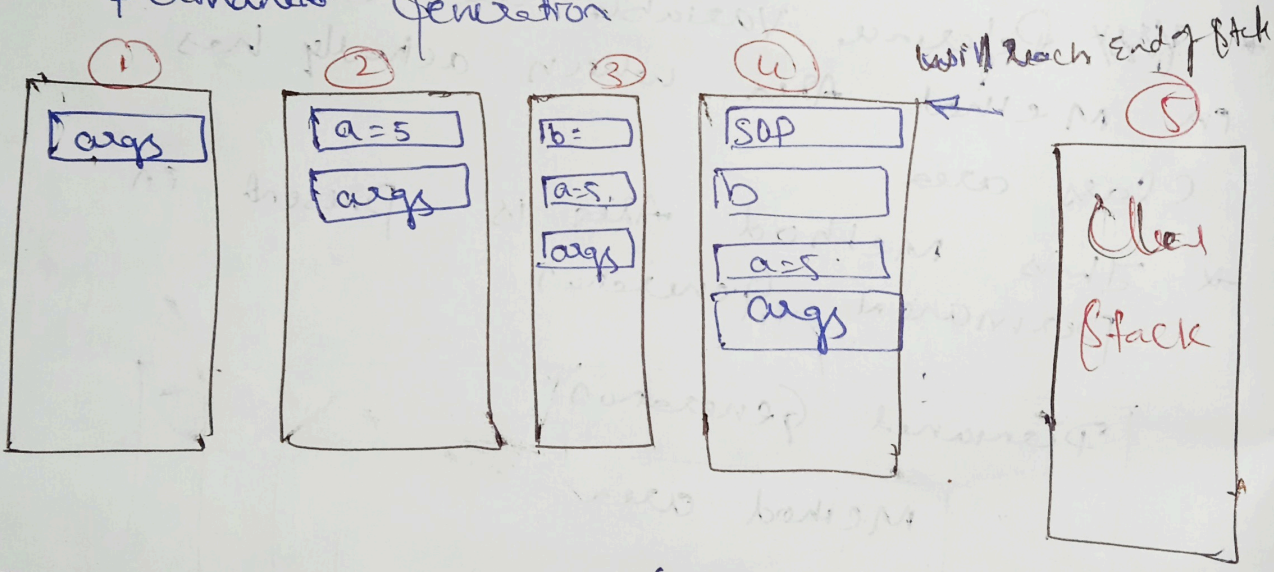
* Each main is Considered as frame and Each frame is Considered as thread

Hence Main is Considered as thread.

NOTE:

* class is Stored in permanent Generation
* Object is stored in Heap (young generation, Eden)
* Referred Variable / local Variable is stored in Stack.

* Stack is actually available outside the Permanent Generation




```
Public static void main (String args[])
{
   int a = 5;
String b= "MUSHU";
System.out.println(" a +" " + b );
}
```