* 1. **Garbage Collection:**

In the Java programming language, dynamic allocation of objects is achieved using the new operator. An object once created uses some memory and the memory remains allocated till there are references for the use of the object. When there are no references for an object, it is assumed to be no longer needed and the memory occupied by the object can be reclaimed. There is no explicit need to destroy an object as java handles the de-allocation automatically. The technique that accomplishes this is known as Garbage Collection. Programs that do not de-allocate memory can eventually crash when there is no memory left in the system to allocate. These programs are said to have memory leaks.

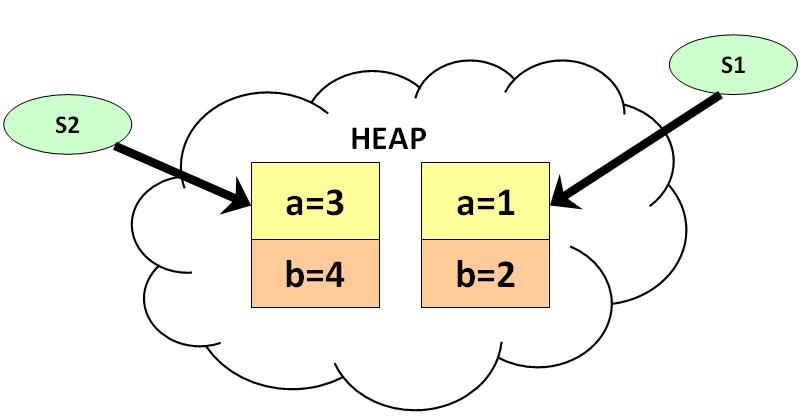
In Java, Garbage collection happens automatically during the lifetime of a java program, eliminating the need to de-allocate memory and avoiding memory leaks.

In C language, it is the programmer’s responsibility to de-allocate memory allocated dynamically using free() function.

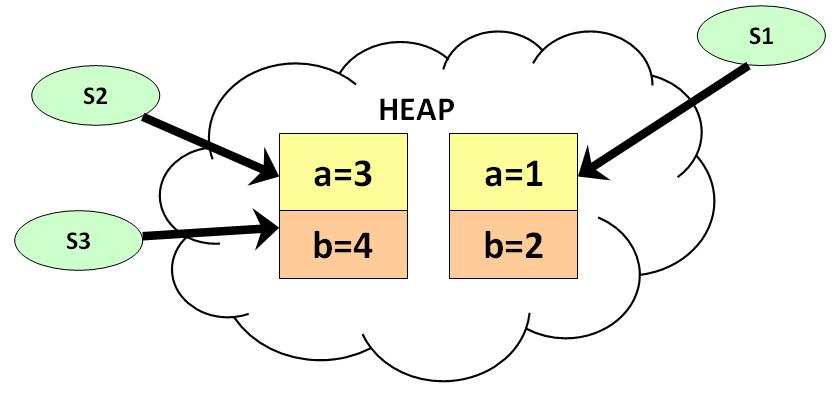
**Note: All objects are created in Heap Section of memory.**

1. class Student{
2. int a;
3. int b;
4. public void setData(int c ,int d){
5. a=c;
6. b=d;
7. }
8. public void showData(){
9. System.out.println("Value of a = "+a);
10. System.out.println("Value of b = "+b);
11. }
12. public static void main(String args[]){
13. Student s1 = new Student();
14. Student s2 = new Student();
15. s1.setData(1,2);
16. s2.setData(3,4);
17. s1.showData();
18. s2.showData();
19. //Student s3;
20. //s3=s2;
21. //s3.showData();
22. //s2=null;
23. //s3.showData();
24. //s3=null;
25. //s3.showData();
26. }
27. }

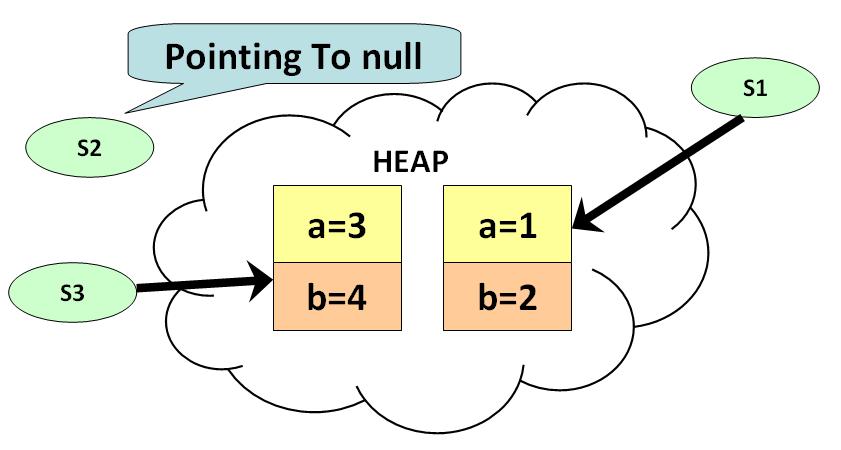
**Save, Compile and Run the code. As shown in the diagram, two objects and two reference variables are created.**



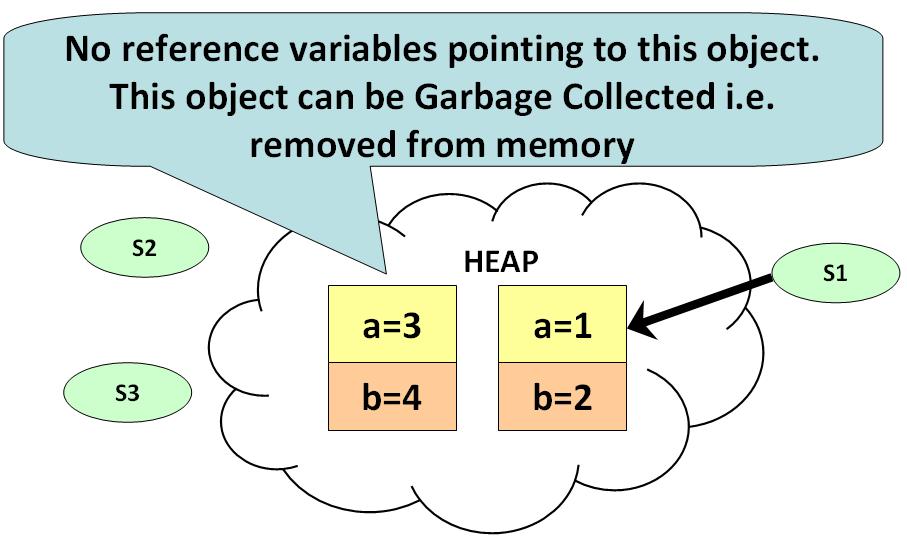
**Uncomment line #19, #20, #21. Save, Compile and Run the code. Two reference variables are pointing to the same object as shown below:**



**Uncomment the line #22, #23. Save, Compile and Run the code. As show in diagram below, s2 becomes null, but s3 is still pointing to the object and is not eligible for garbage collection.**



**Uncomment line # 24 & 25. Save, Compile & Run the Code. At this point there are no references pointing to the object and becomes eligible for garbage collection. It will be removed from memory and there is no way of retrieving it back.**



* 1. **The finalize() Method:**

The java.lang.Object.finalize() is called by the garbage collector on an object when garbage collection determines that there are no more references to the object. A subclass overrides the finalize method to dispose of system resources or to perform other cleanup.

**Syntax:**

protected void finalize()