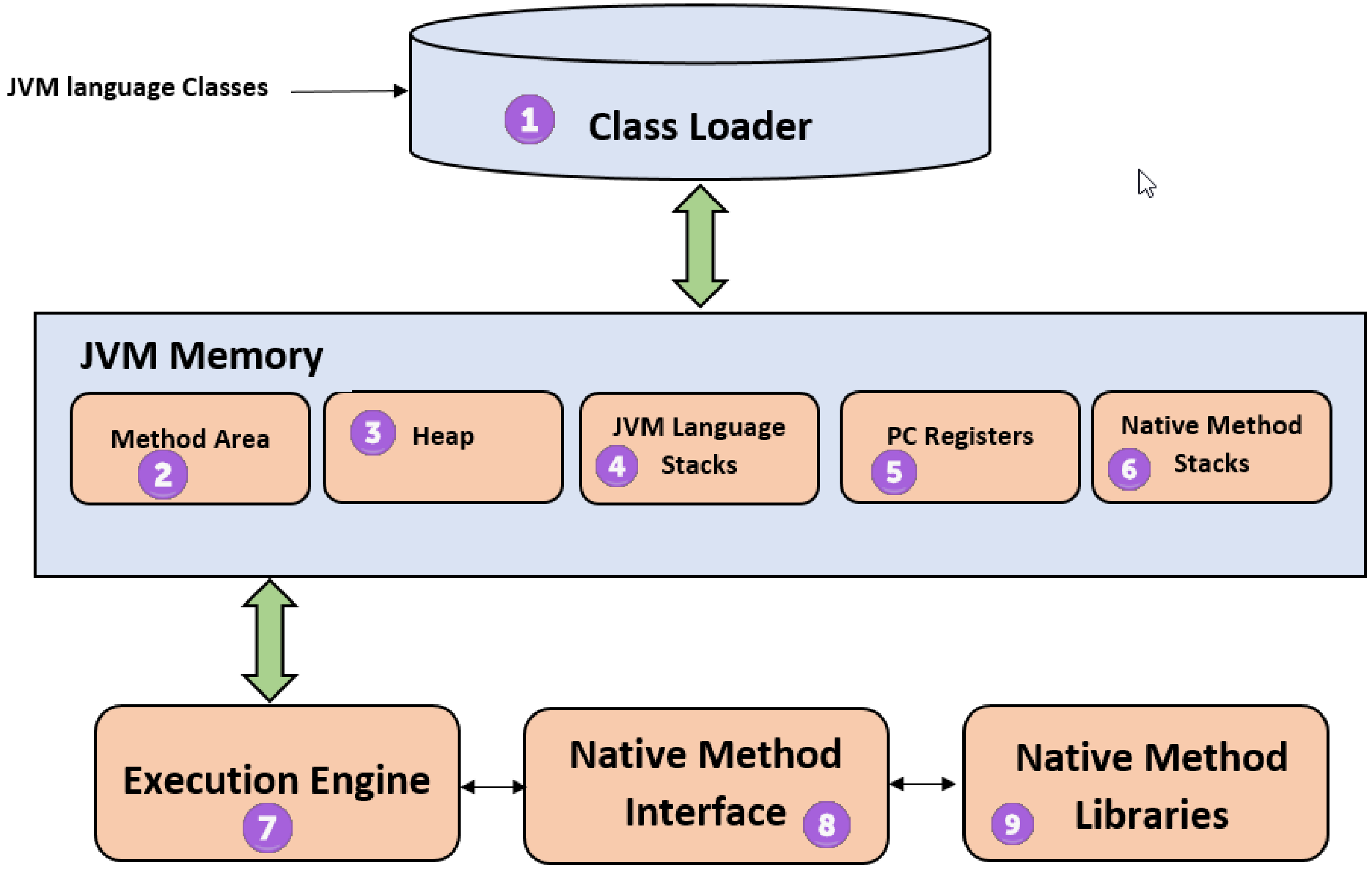
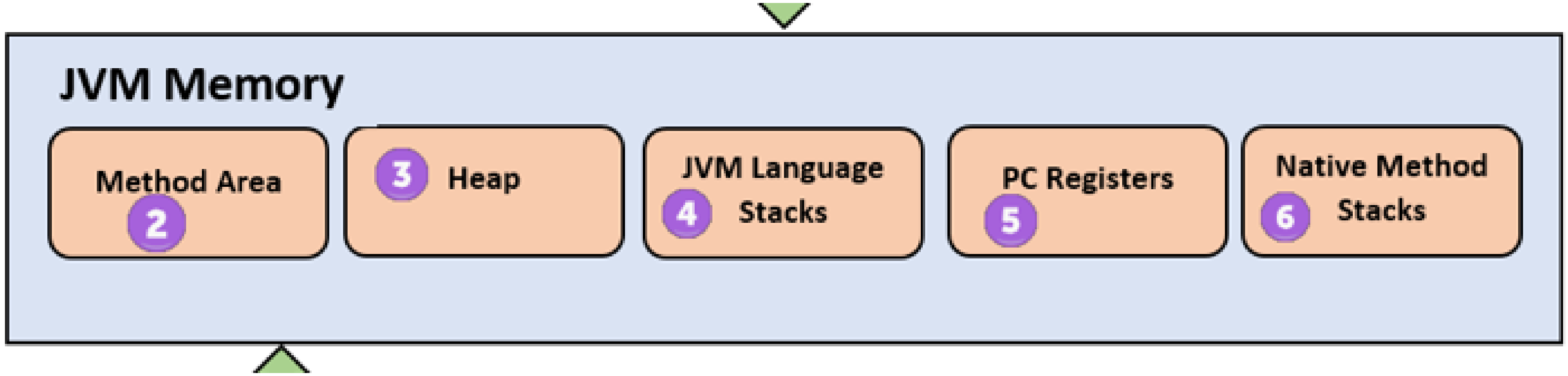
Garbage Collector

GarbageCollector

- 동적으로 할당된 메모리 중 필요없게 된 영역을 해제 하는 기능


JVM language Classes
1 Class Loader
JVM Memory
Method Area 2
3 Heap
JVM Language 4 Stacks
PC Registers 5
Native Method 6 Stacks
Execution Engine 7
Native Method Interface 8
Native Method 9 Libraries



JVM Memory

Method Area 2

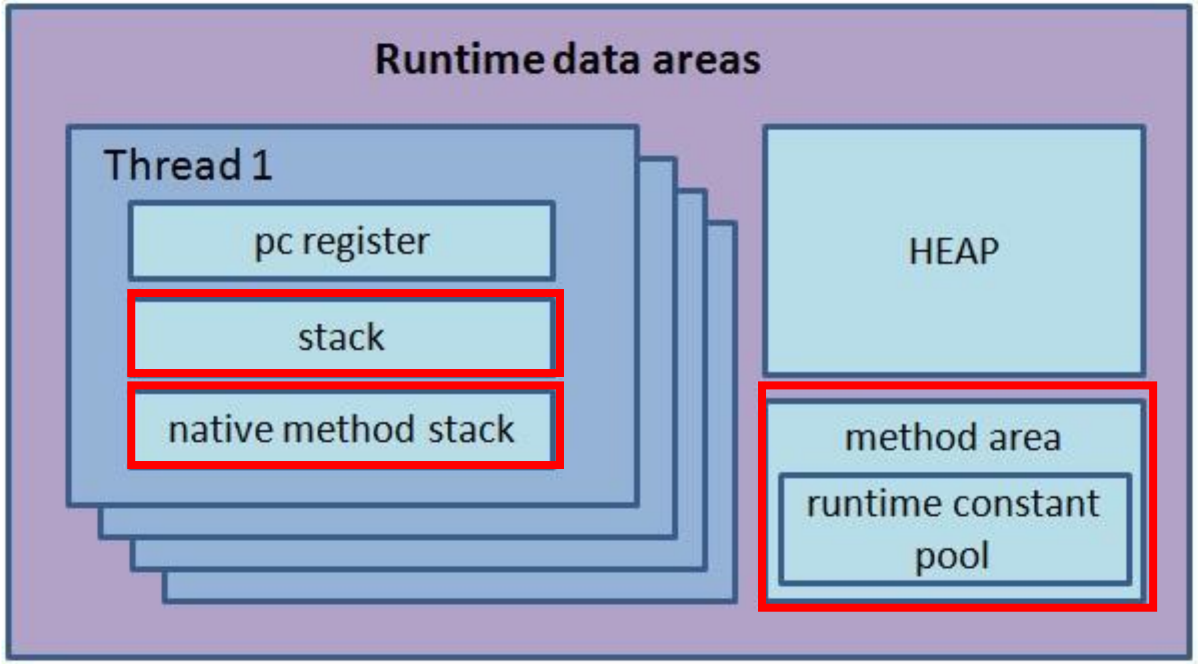
3 Heap

JVM Language 4 Stacks

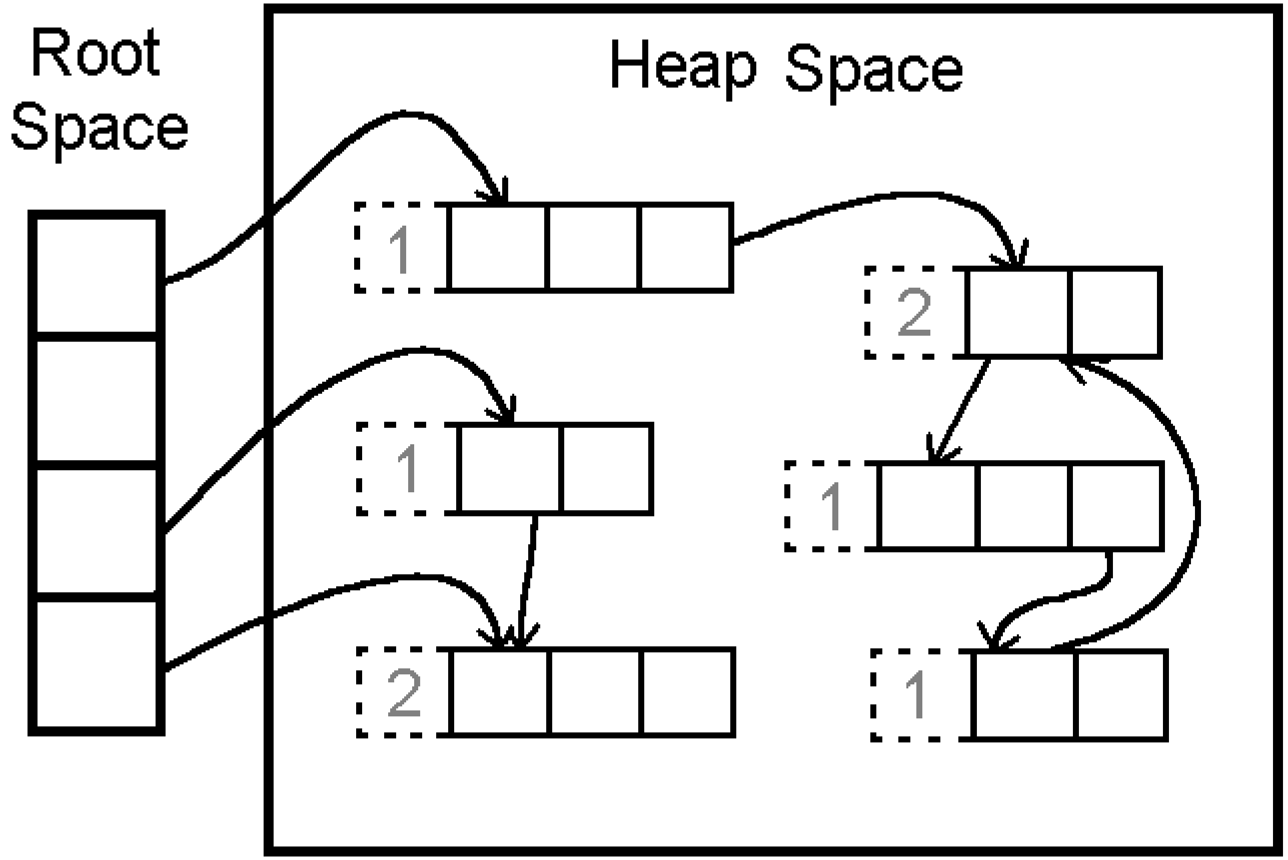
PC Registers 5

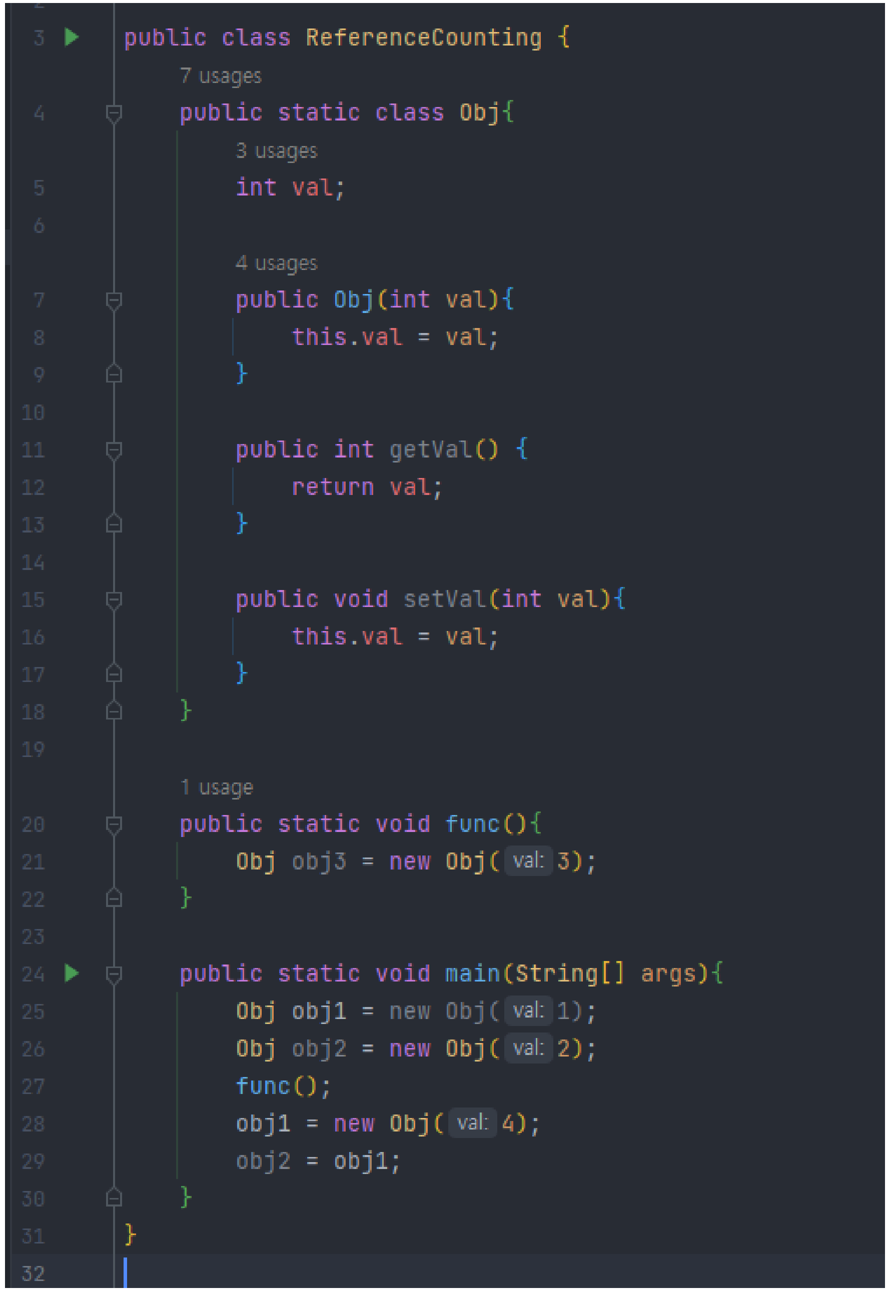
Native Method 6 Stacks

RootSpace

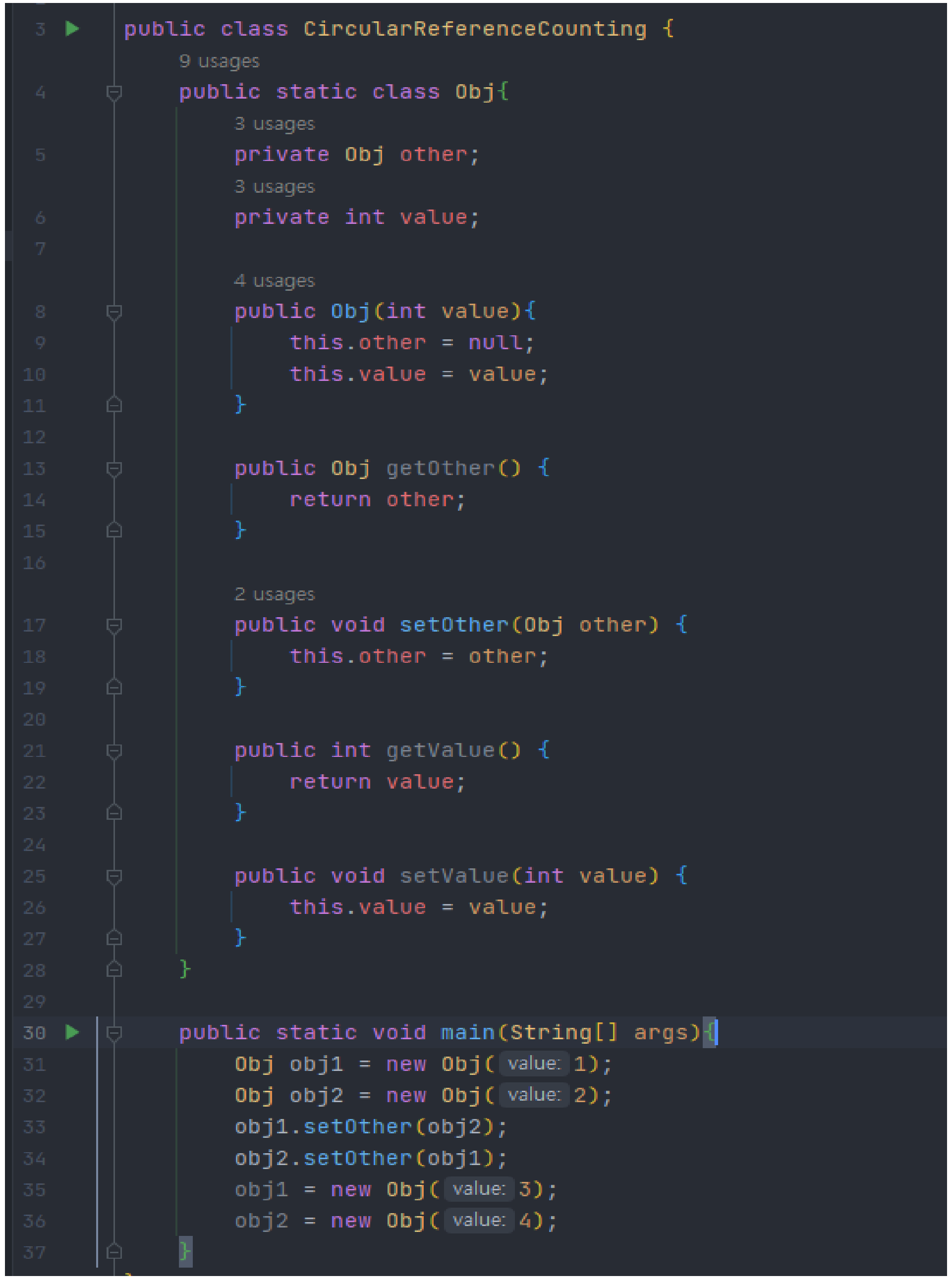

Runtime data areas
Thread 1
pc register
stack
native method stack
HEAP
method area
runtime constant pool

Reference Counting



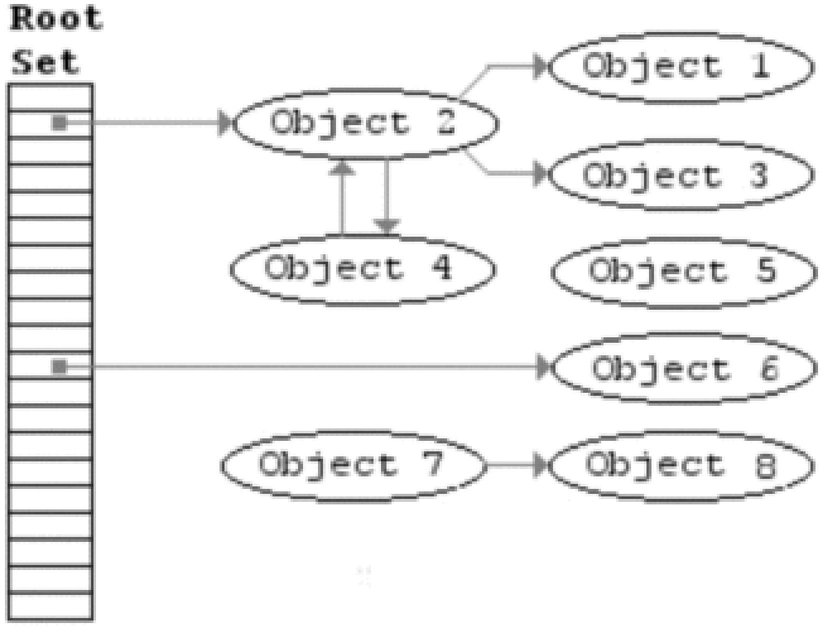


```
public class ReferenceCounting {
    7 usages
    public static class Obj{
        3 usages
        int val;

        4 usages
        public Obj(int val){
            this.val = val;
        }

        public int getVal() {
            return val;
        }

        public void setVal(int val){
            this.val = val;
        }
    }

    1 usage
    public static void func(){
        Obj obj3 = new Obj( val: 3);
    }

    public static void main(String[] args){
        Obj obj1 = new Obj( val: 1);
        Obj obj2 = new Obj( val: 2);
        func();
        obj1 = new Obj( val: 4);
        obj2 = obj1;
    }
}
```

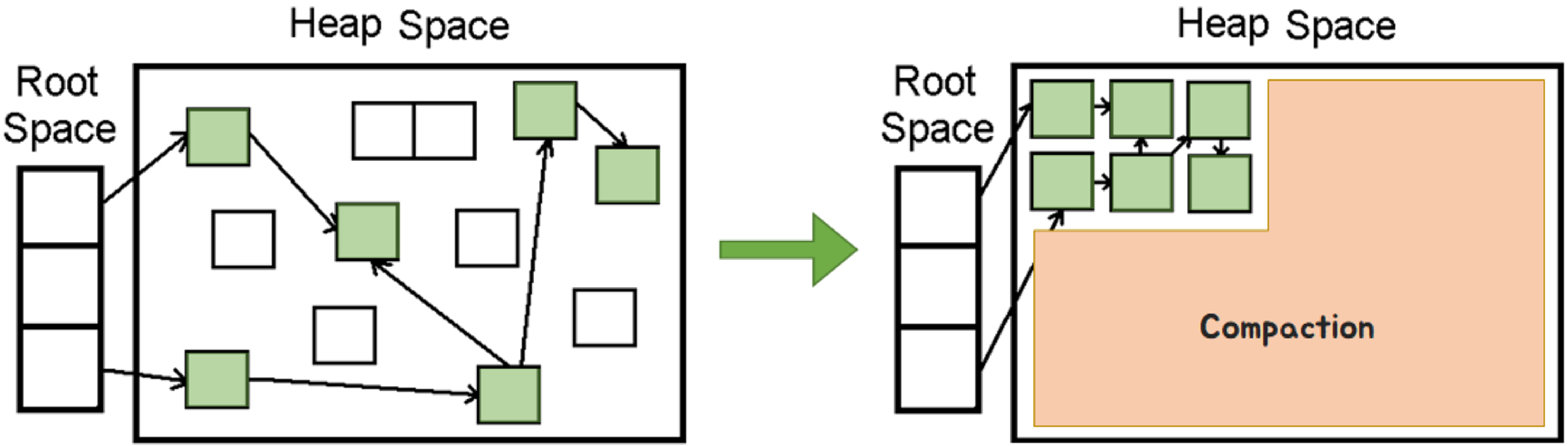


```
public class CircularReferenceCounting {
    9 usages
    public static class Obj{
        3 usages
        private Obj other;
        3 usages
        private int value;

        4 usages
        public Obj(int value){
            this.other = null;
            this.value = value;
        }

        public Obj getOther() {
            return other;
        }

        2 usages
        public void setOther(Obj other) {
            this.other = other;
        }

        public int getValue() {
            return value;
        }

        public void setValue(int value) {
            this.value = value;
        }
    }

    public static void main(String[] args){
        Obj obj1 = new Obj( value: 1);
        Obj obj2 = new Obj( value: 2);
        obj1.setOther(obj2);
        obj2.setOther(obj1);
        obj1 = new Obj( value: 3);
        obj2 = new Obj( value: 4);
    }
```

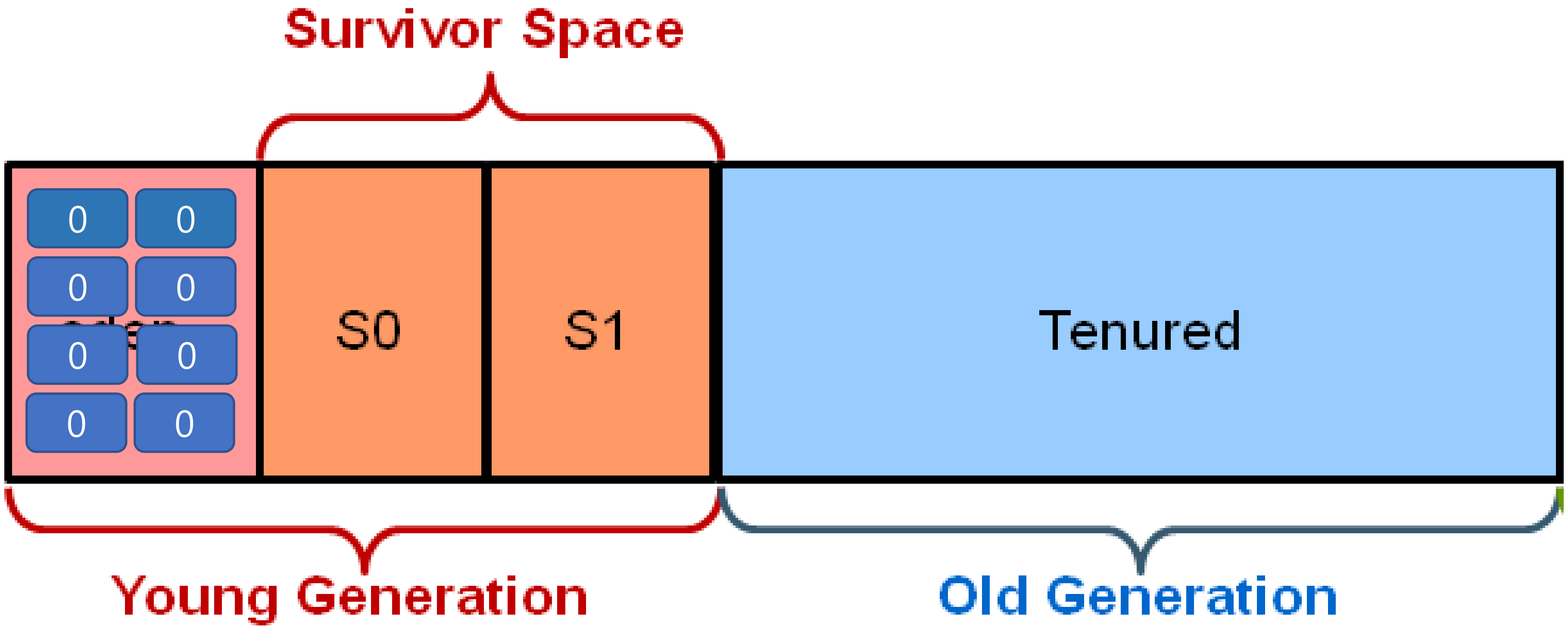
Mark And Sweep



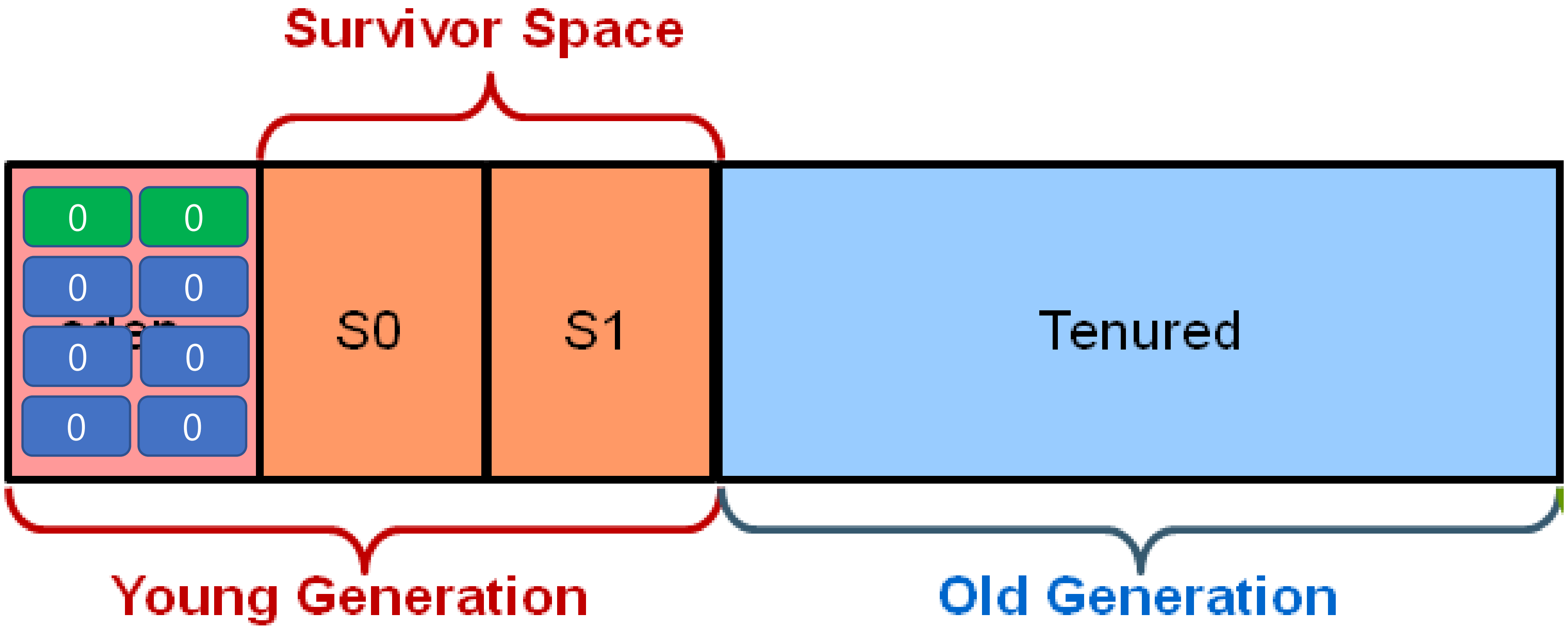
Mark And Sweep



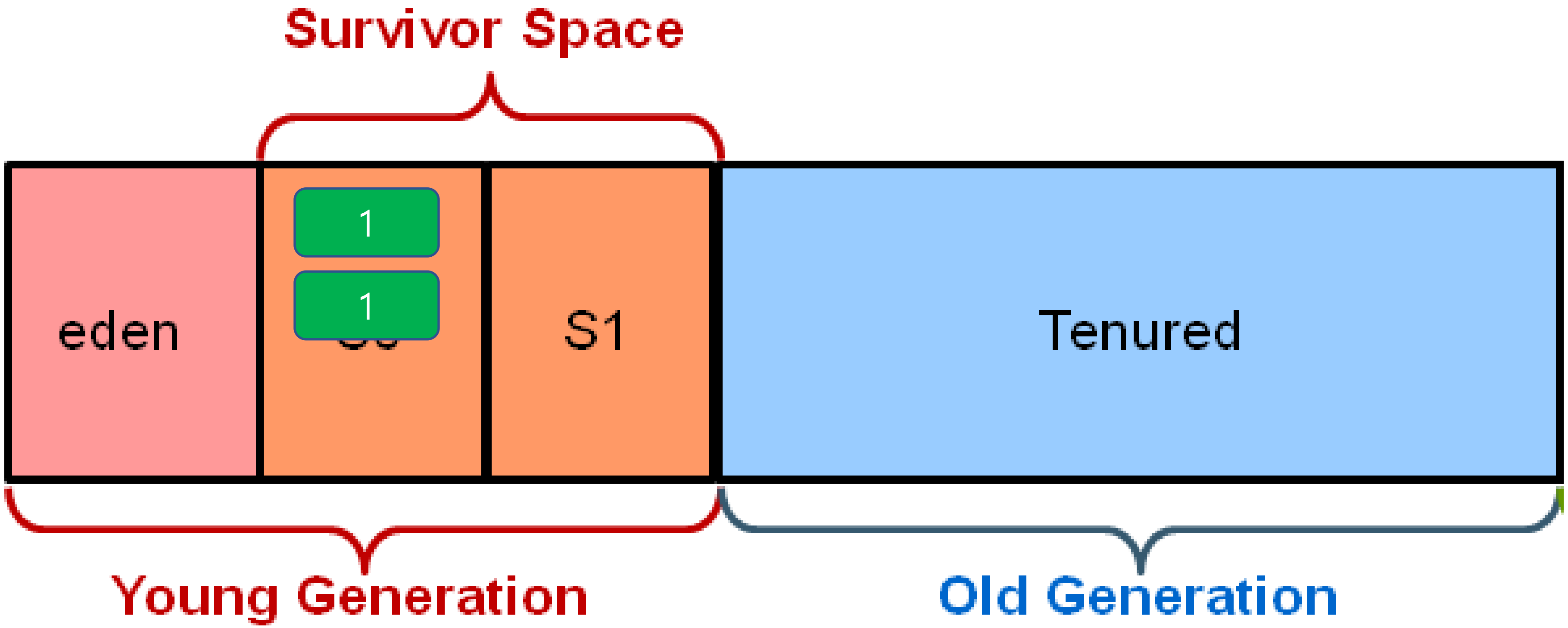
GC 의 실행원리


Survivor Space
0
0
0
0
eden
0
0
0
0
S0
S1
Tenured
Young Generation
Old Generation

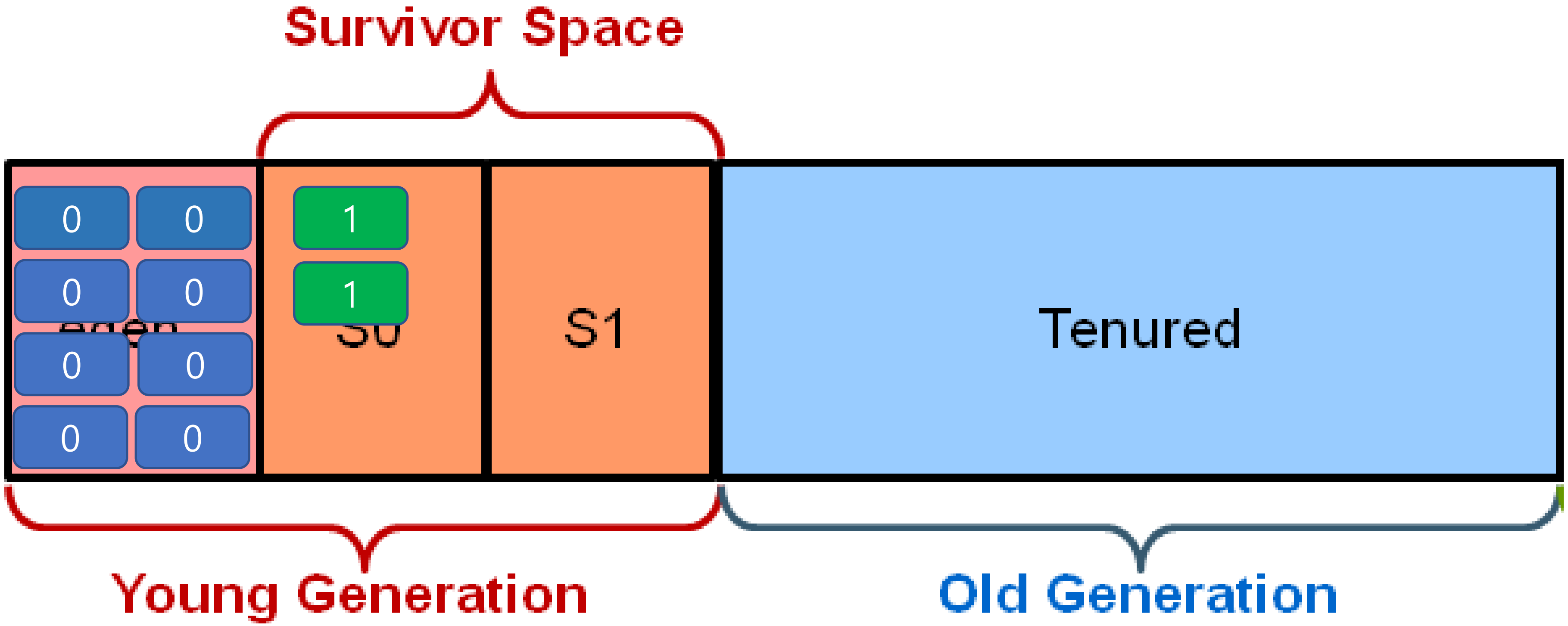
GC 의 실행원리


Survivor Space
0
0
0
0
eden
0
0
0
0
S0
S1
Tenured
Young Generation
Old Generation

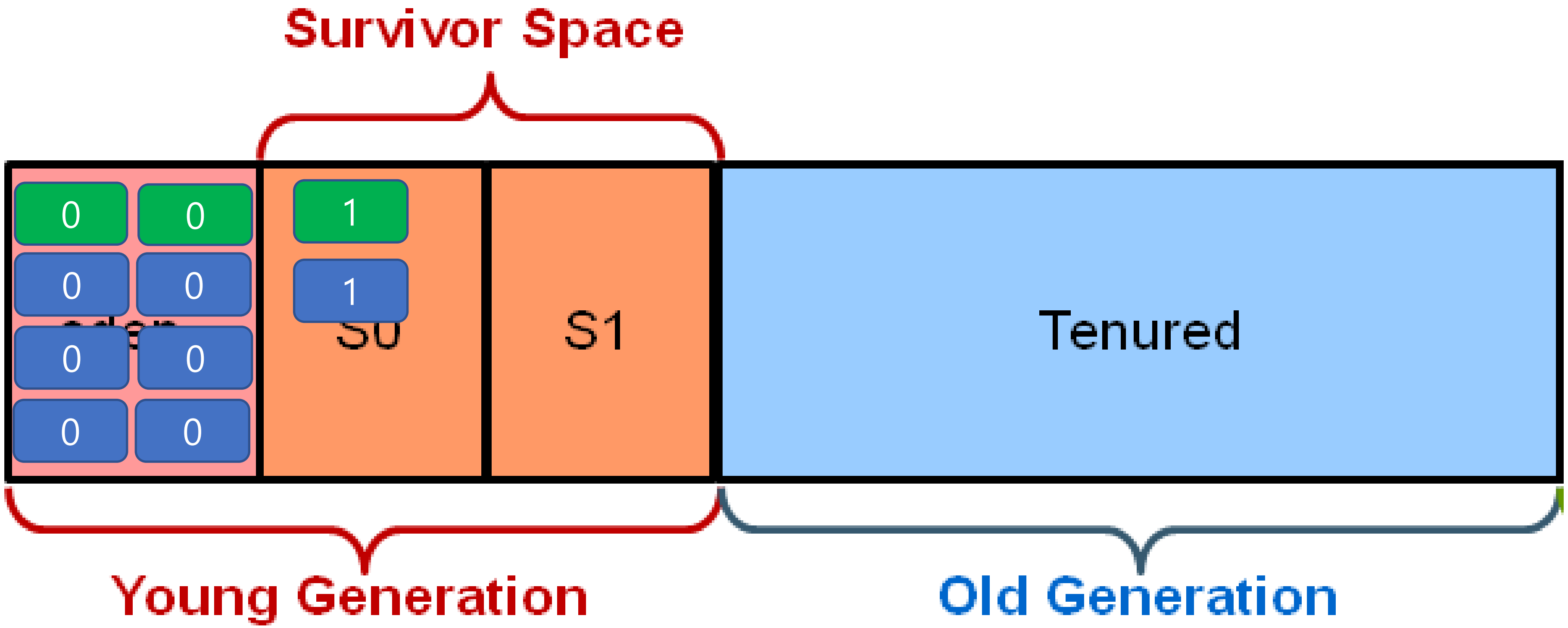
GC 의 실행원리


Survivor Space
eden
1
1
S1
Tenured
Young Generation
Old Generation

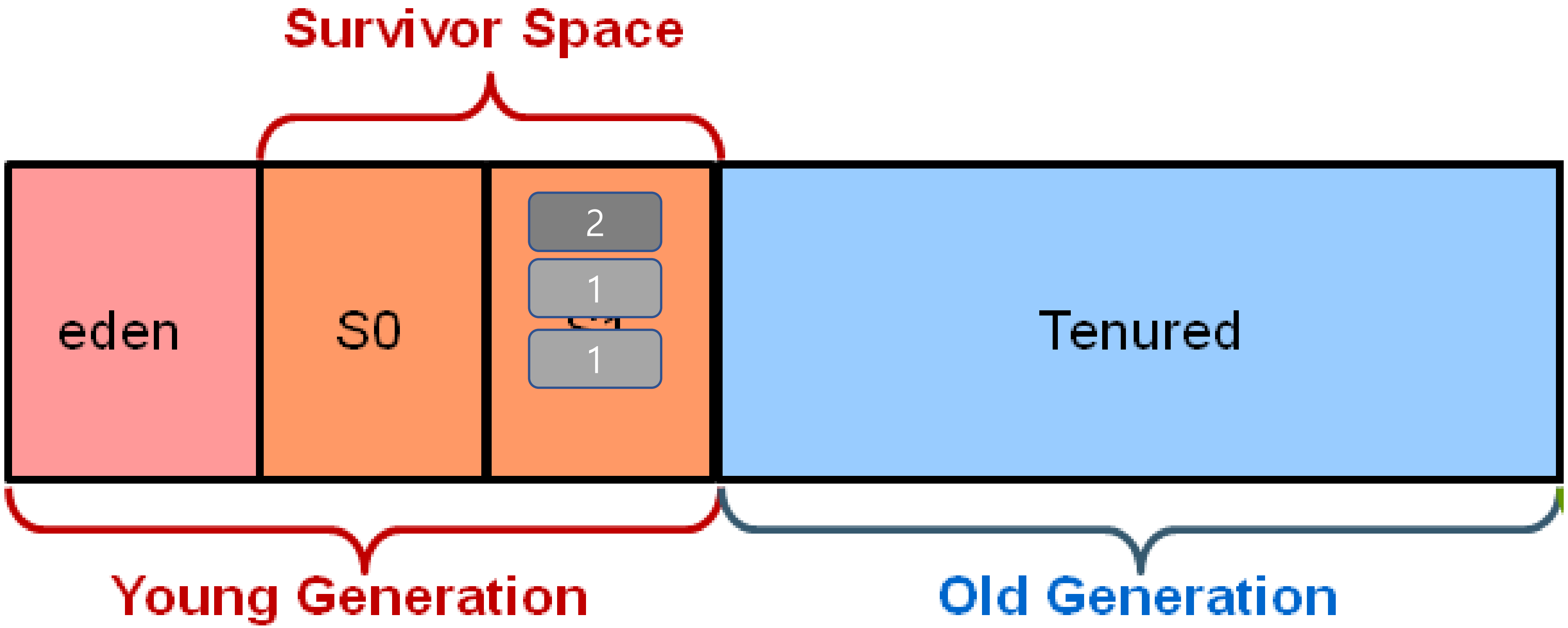
GC 의 실행원리


Survivor Space
0
0
0
0
0
0
0
0
eden
1
1
S0
S1
Tenured
Young Generation
Old Generation

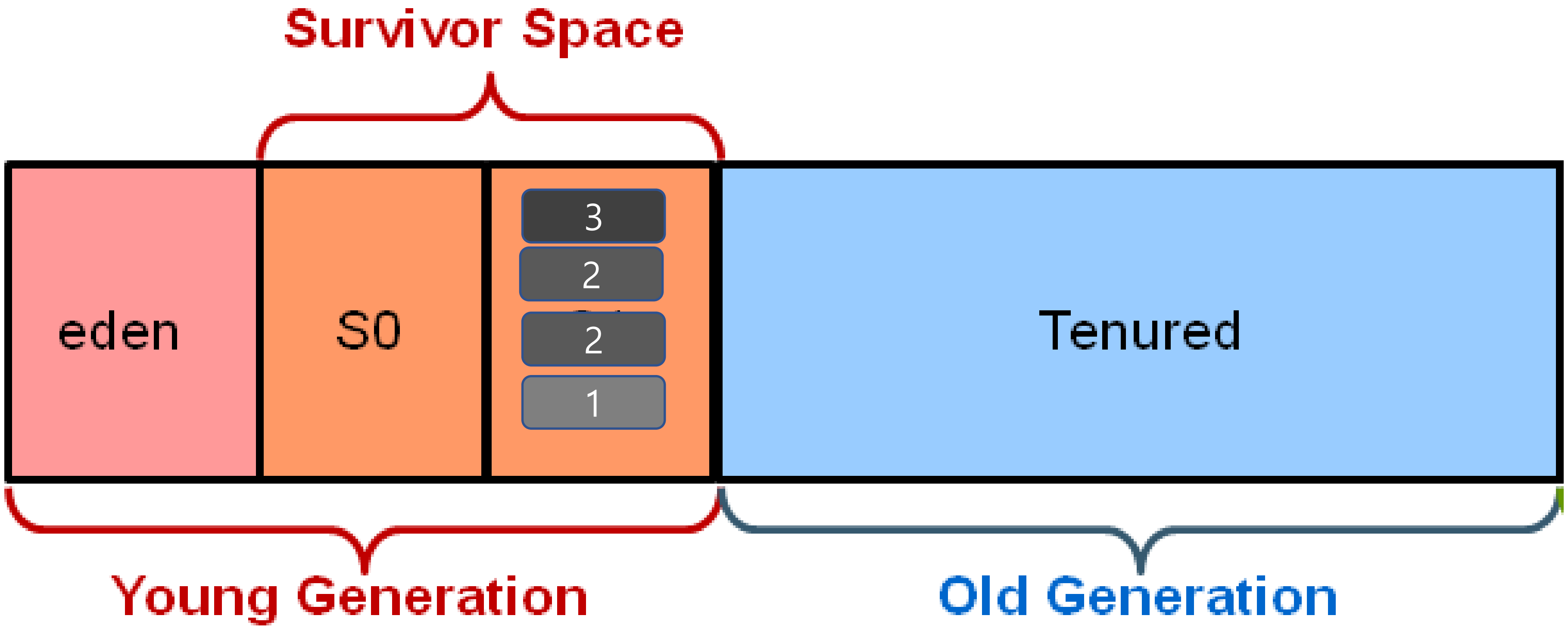
GC 의 실행원리


Survivor Space
0
0
0
0
eden
0
0
0
0
1
1
S0
S1
Tenured
Young Generation
Old Generation

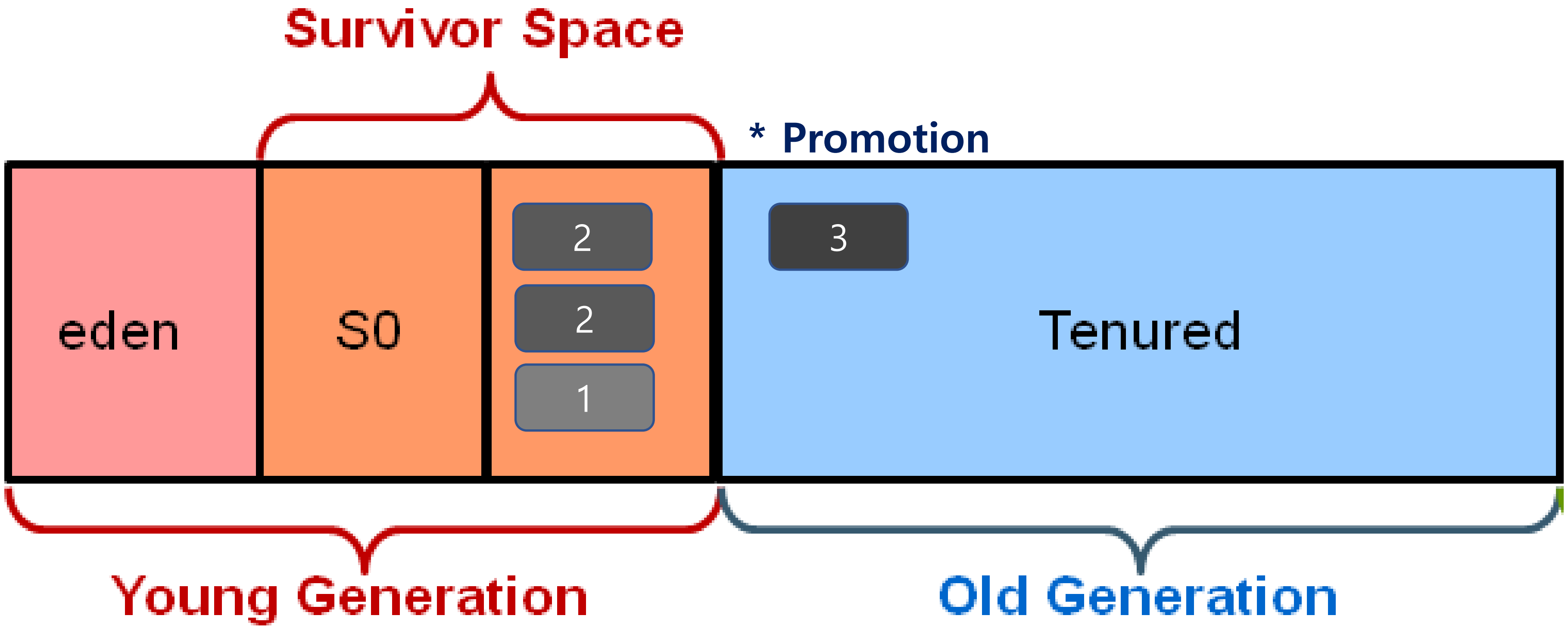
GC 의 실행원리


Survivor Space
eden
S0
2
1
1
Tenured
Young Generation
Old Generation

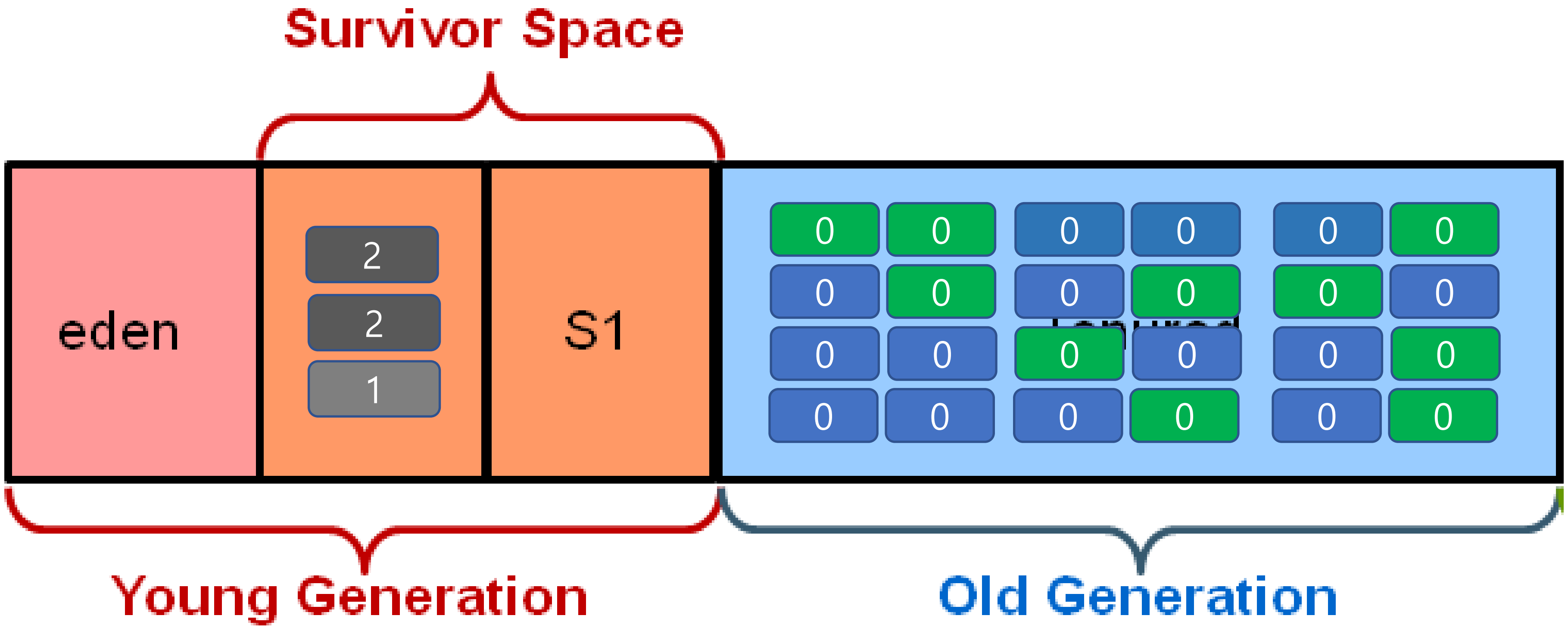
GC 의 실행원리

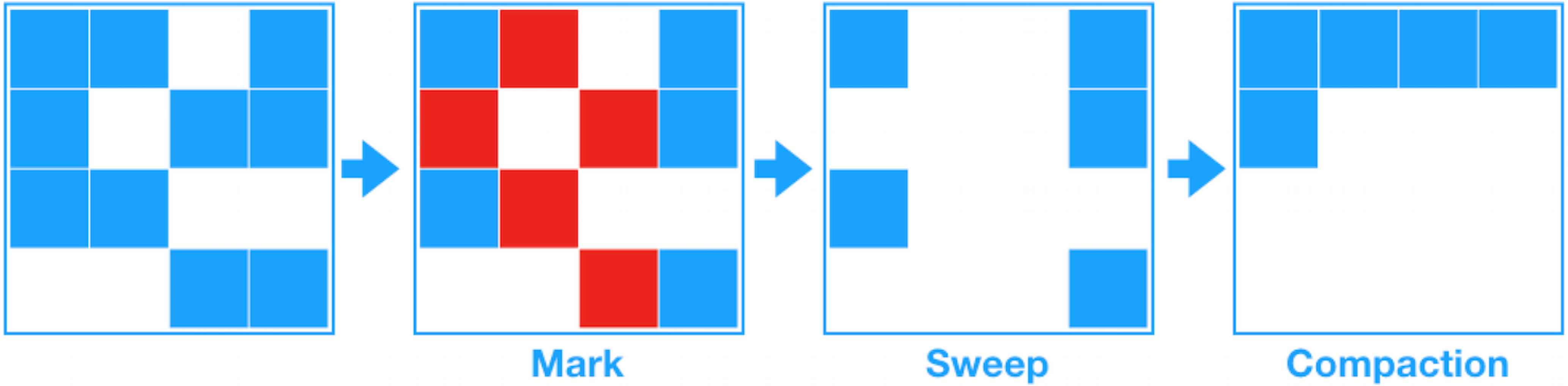

Survivor Space
eden
S0
3
2
2
1
Tenured
Young Generation
Old Generation

GC 의 실행원리

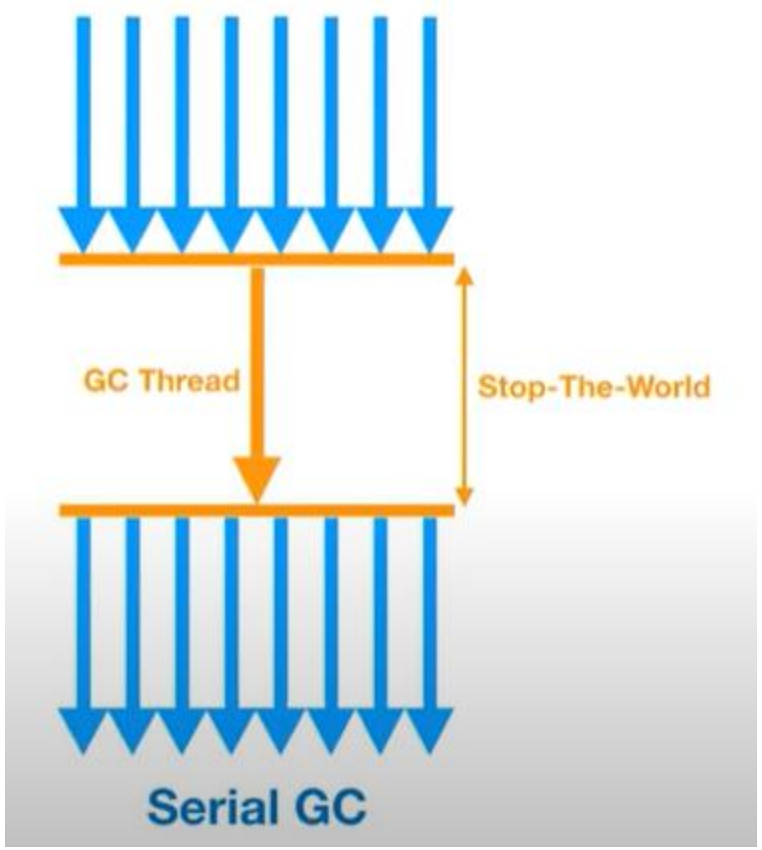

Survivor Space
* Promotion
eden
S0
2
2
1
3
Tenured
Young Generation
Old Generation

GC 의 실행원리

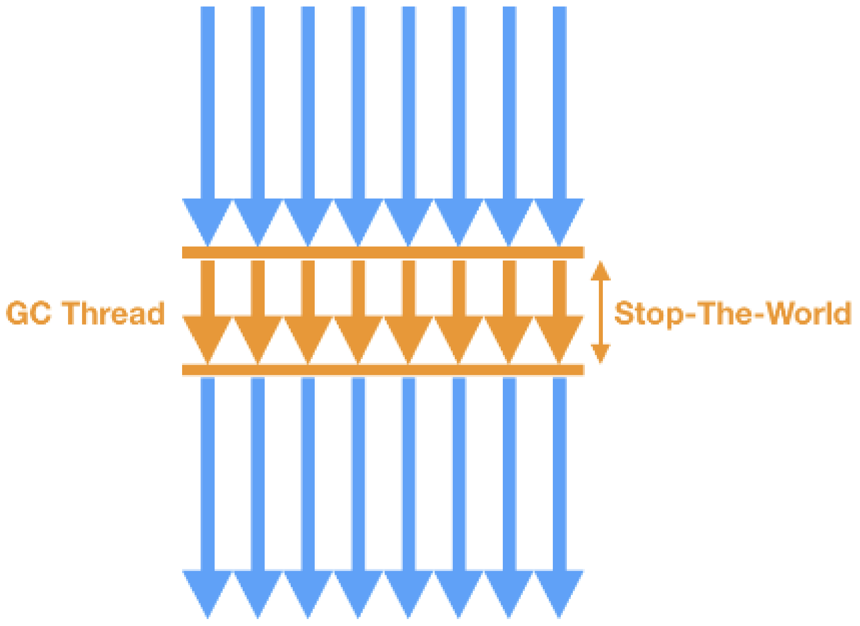

Survivor Space
eden
2
2
1
S1
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
Young Generation
Old Generation


Mark
Sweep
Compaction

Serial GC



Parallel GC

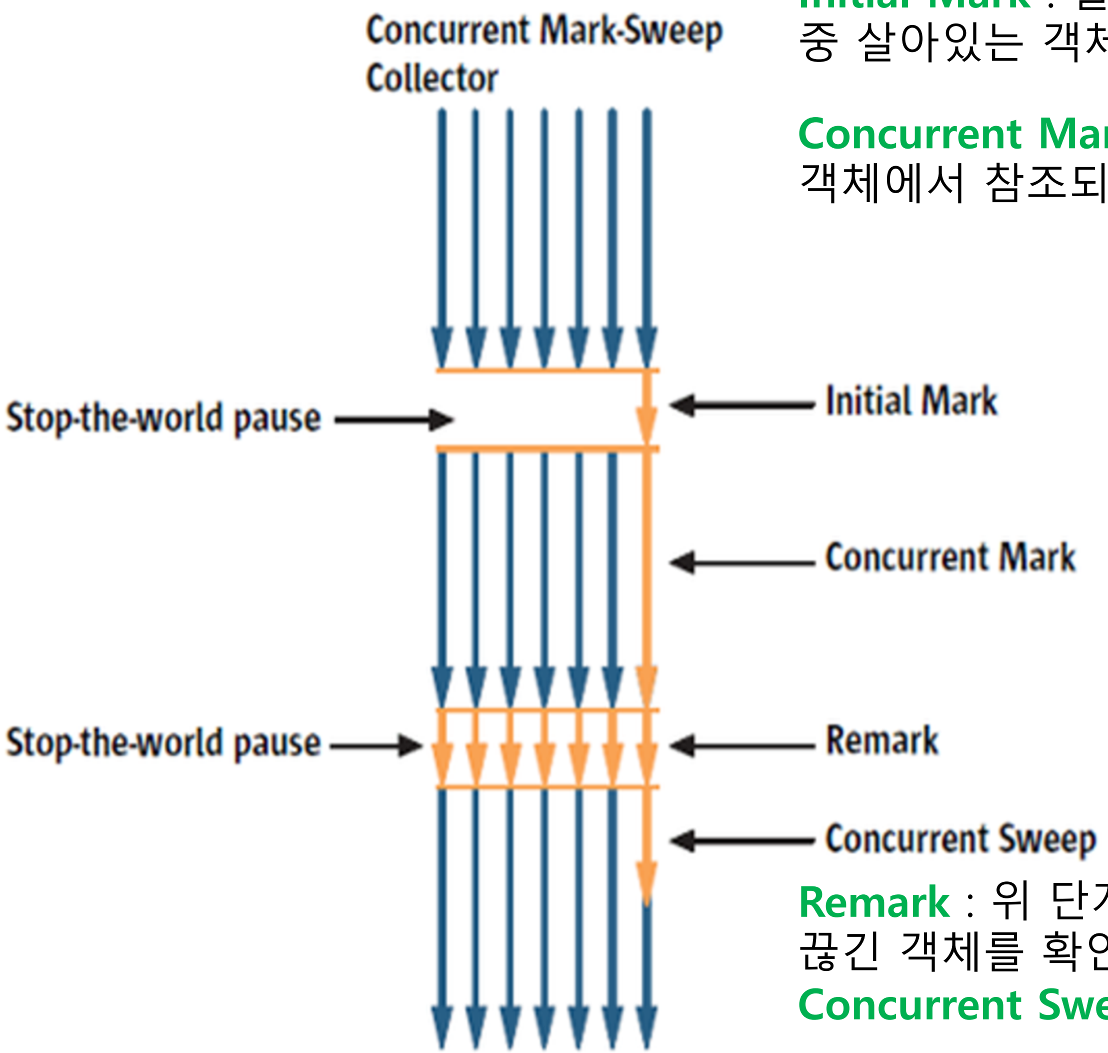


Parallel GC

CMS(Concurrent Mark Sweep) GC

Initial Mark : 클래스 로더에서 가장 가까운 객체 중 살아있는 객체만 찾는다.

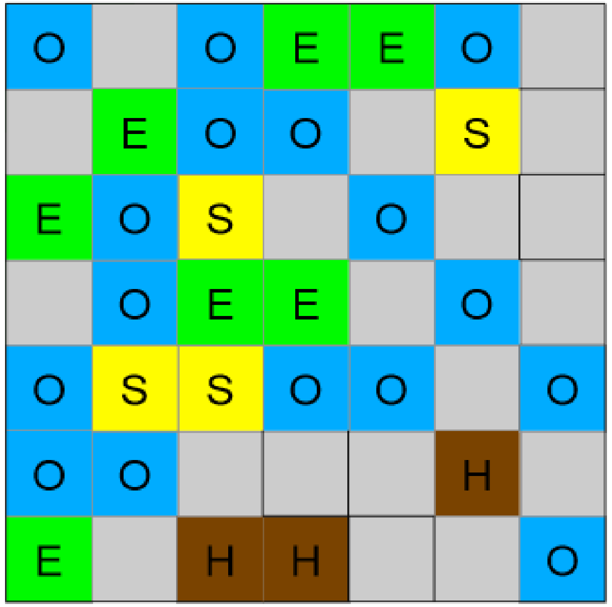
Concurrent Mark : 위에서 살아있다고 확인한 객체에서 참조되어 있는 객체를 확인한다.



Remark : 위 단계에서 새로 추가되거나 참조가 끊긴 객체를 확인

Concurrent Sweep : 쓰레기를 정리

G1 GC



O		O	E	E	O	
	E	O	O		S	
E	O	S		O		
	O	E	E		O	
O	S	S	O	O		O
O	O				H	
E		H	H			O

E Eden regions

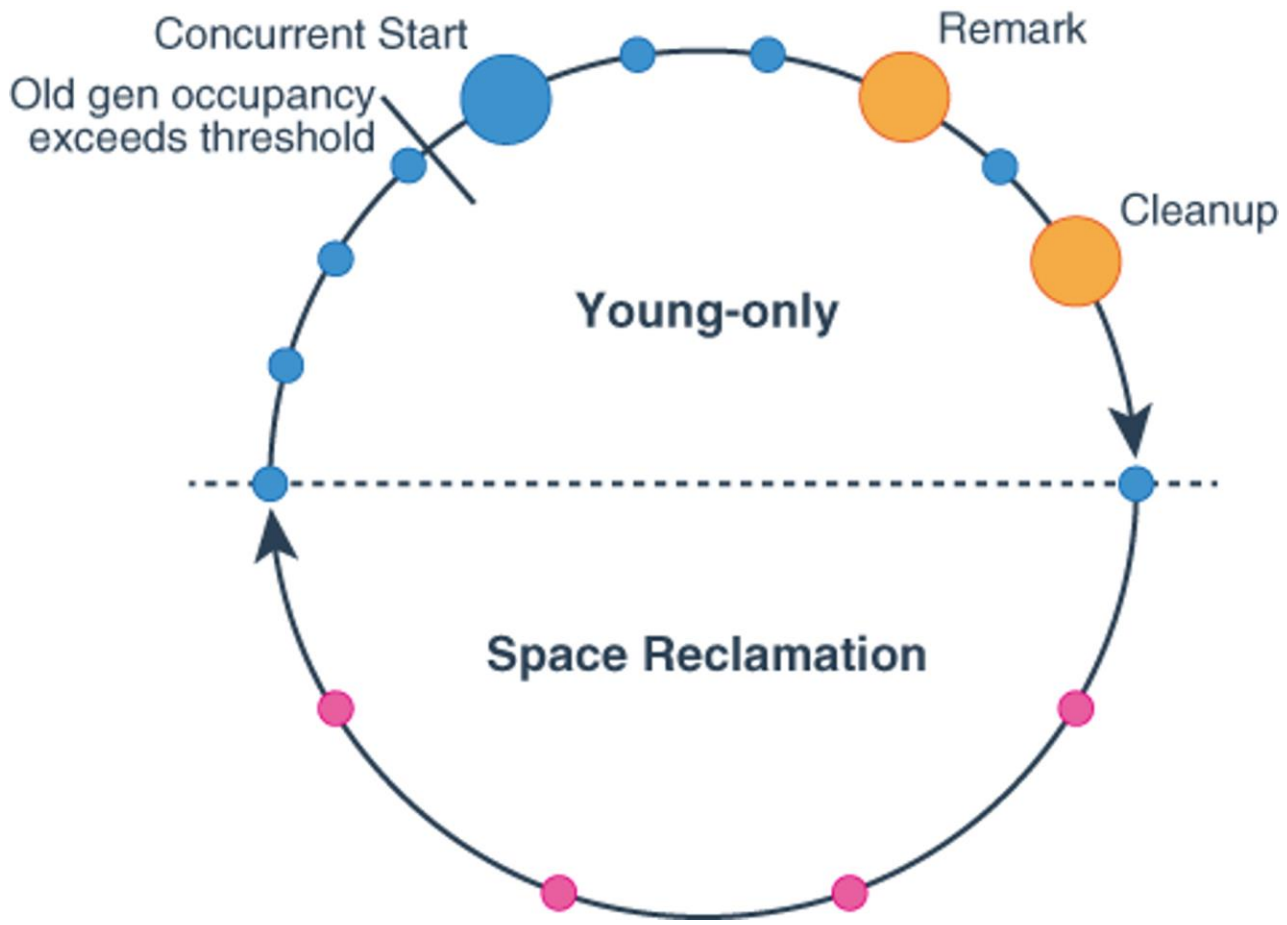
S Survivor regions

O Old generation regions

H Humongous regions : Region 크기의 50%를 초과하는 큰 객체를 저장하기 위한 공간

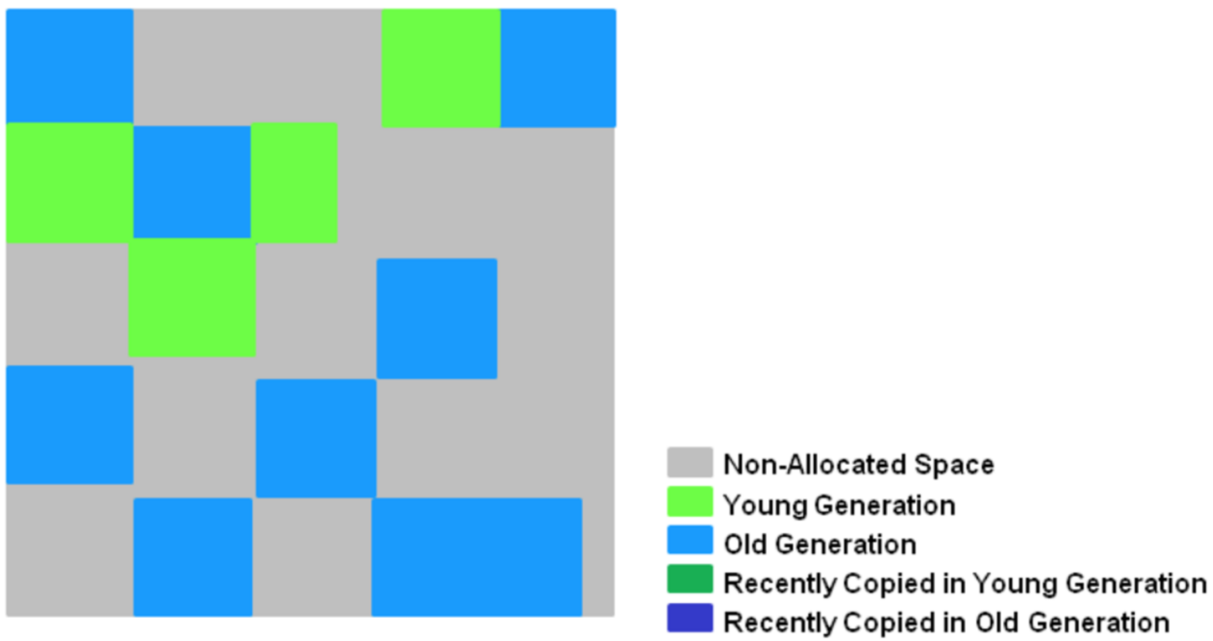
⬜ Available / Unused regions : 아직 사용되지 않은 Region

G1 GC의 Cycle

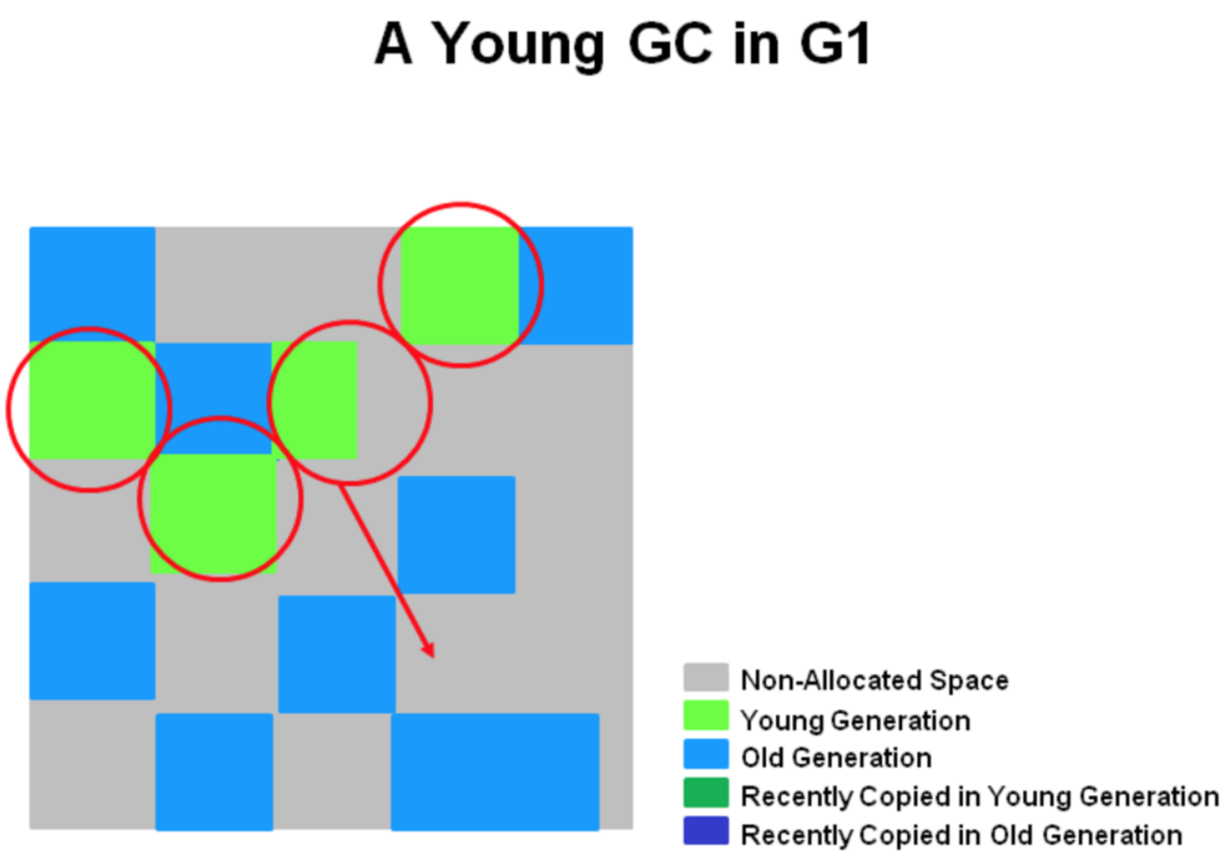

Concurrent Start
Old gen occupancy exceeds threshold
Remark
Cleanup
Young-only
Space Reclamation

Minor GC

Young Generation in G1

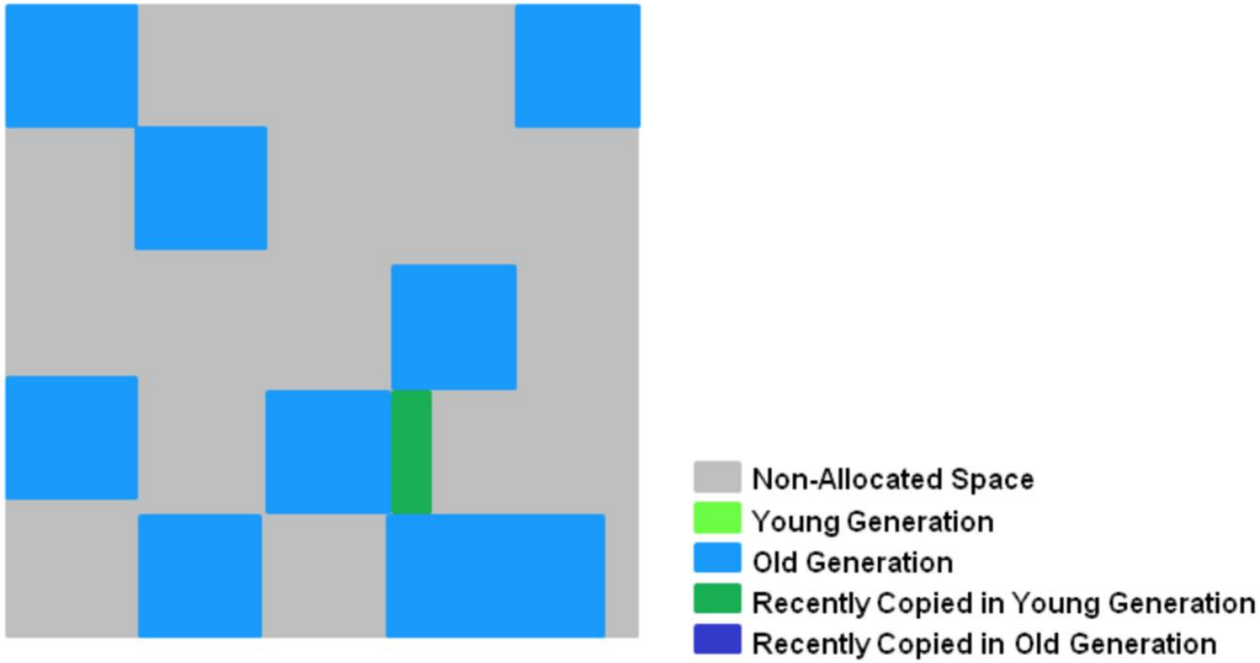

Non-Allocated Space
Young Generation
Old Generation
Recently Copied in Young Generation
Recently Copied in Old Generation

Minor GC

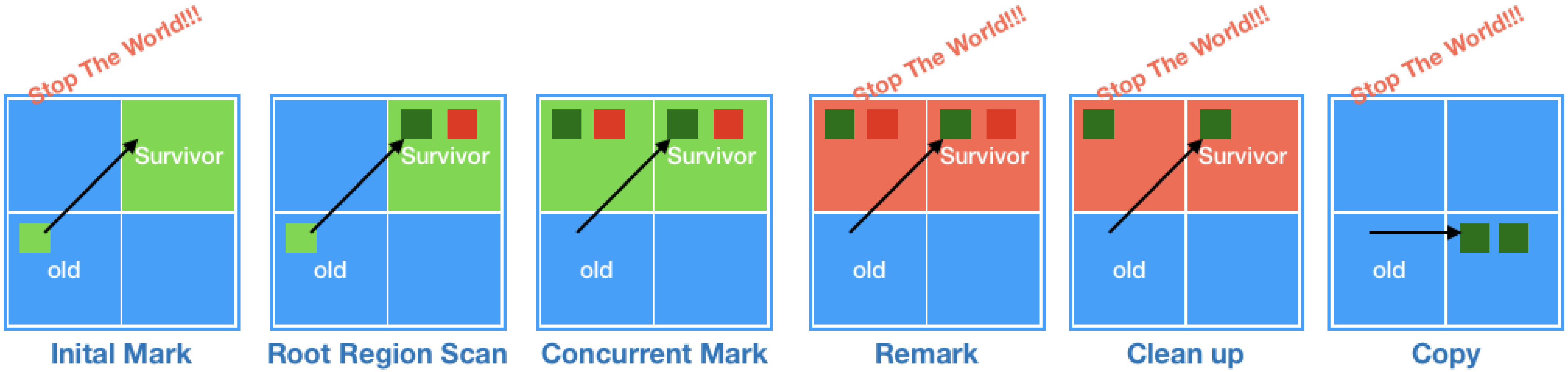

A Young GC in G1
Non-Allocated Space
Young Generation
Old Generation
Recently Copied in Young Generation
Recently Copied in Old Generation

Minor GC

End of Young GC with G1

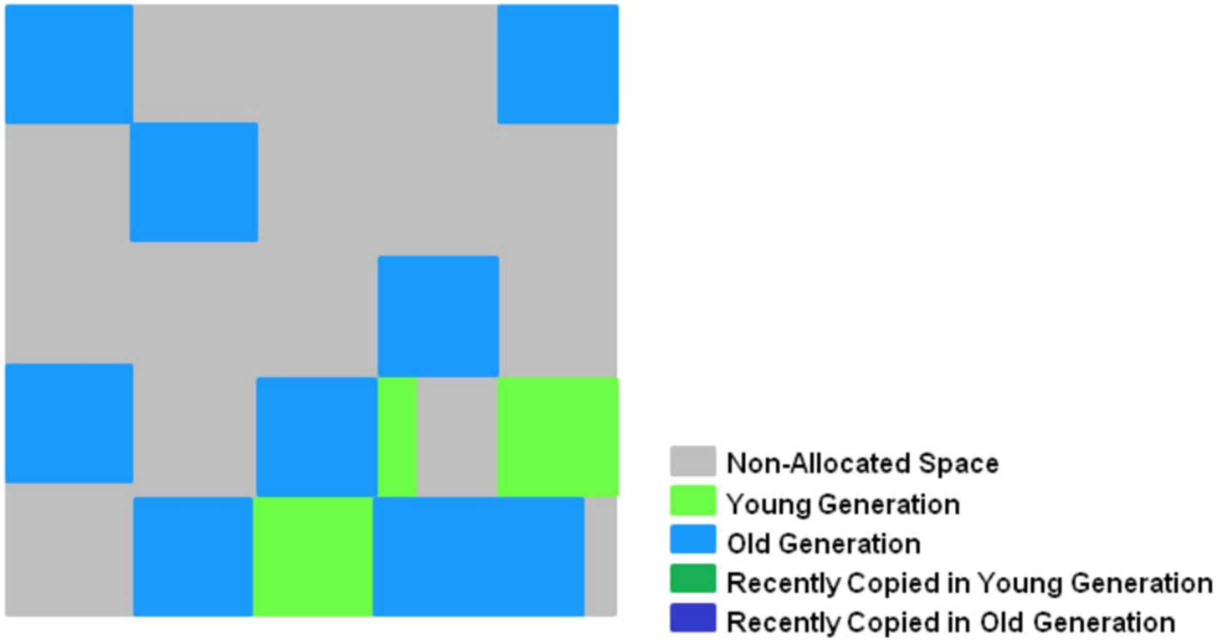

⬜ Non-Allocated Space
🟩 Young Generation
🟦 Old Generation
🟩 Recently Copied in Young Generation
🟦 Recently Copied in Old Generation

Major GC


Stop The World!!!
Survivor
old
Inital Mark
Survivor
old
Root Region Scan
Survivor
old
Concurrent Mark
Stop The World!!!
Survivor
old
Remark
Stop The World!!!
Survivor
old
Clean up
Stop The World!!!
old
Copy

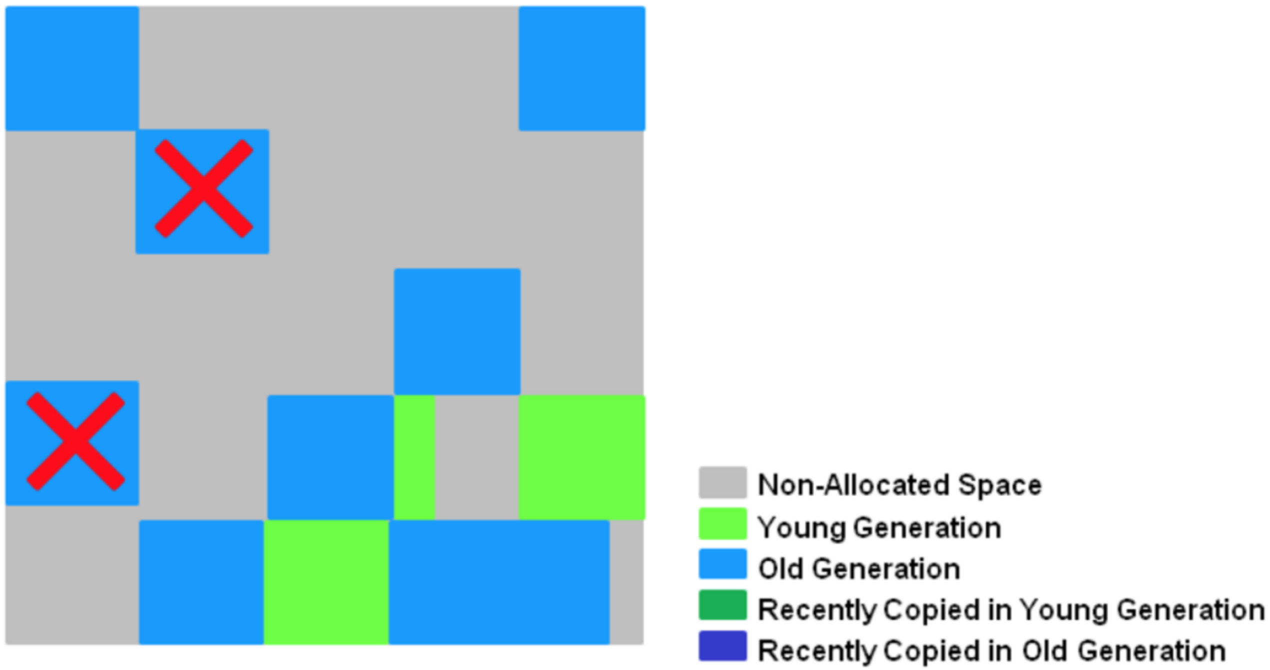
Major GC

Initial Marking Phase

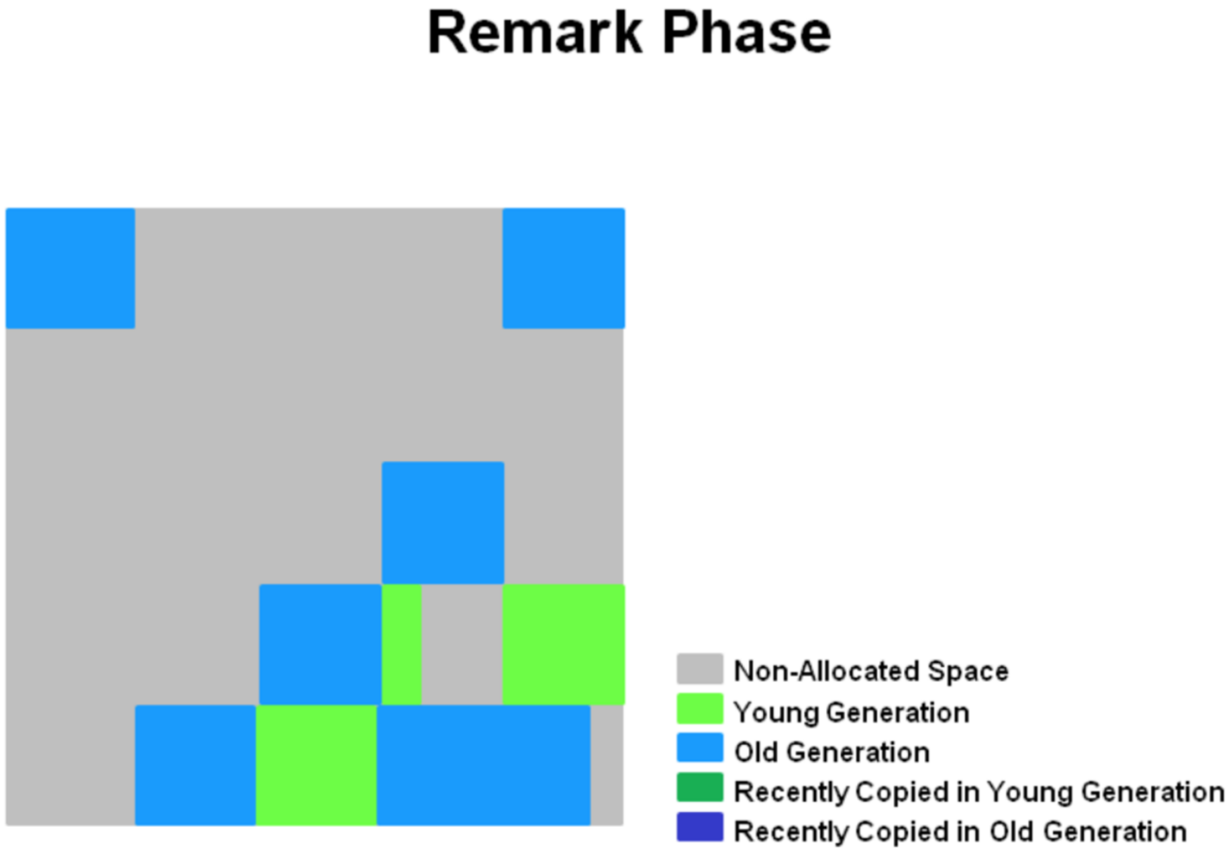

Non-Allocated Space
Young Generation
Old Generation
Recently Copied in Young Generation
Recently Copied in Old Generation

Major GC

Concurrent Marking Phase

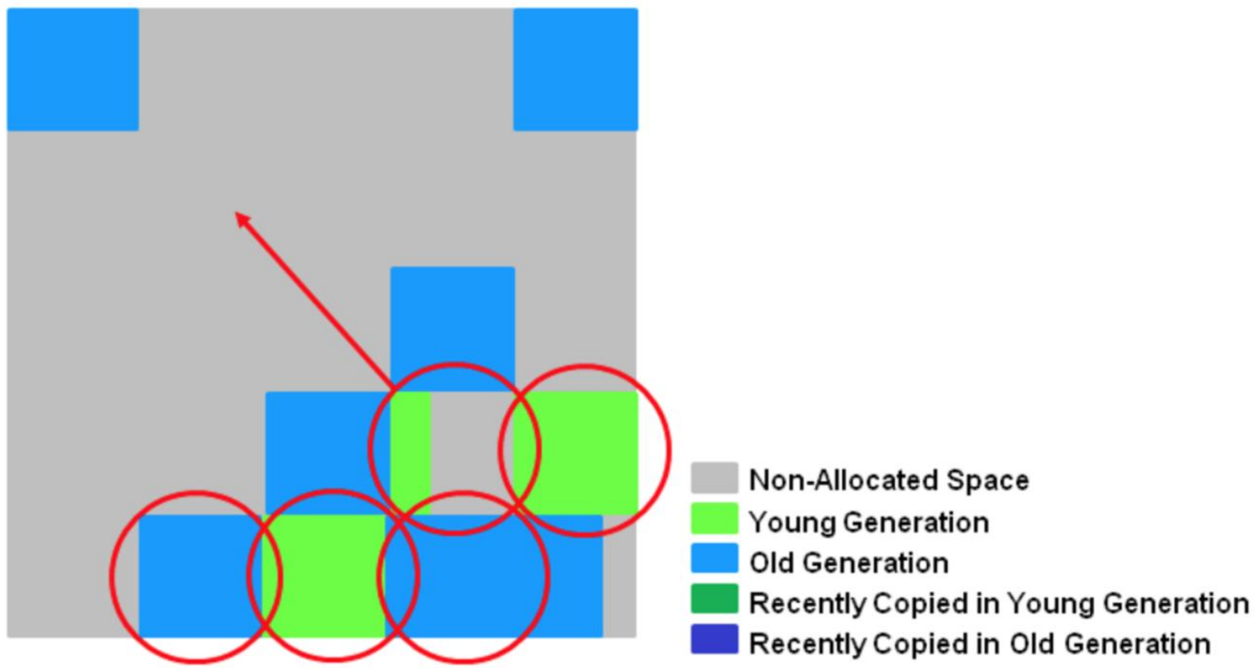

⬜ Non-Allocated Space
🟩 Young Generation
🟦 Old Generation
🟩 Recently Copied in Young Generation
🟦 Recently Copied in Old Generation

Major GC


Remark Phase
Non-Allocated Space
Young Generation
Old Generation
Recently Copied in Young Generation
Recently Copied in Old Generation

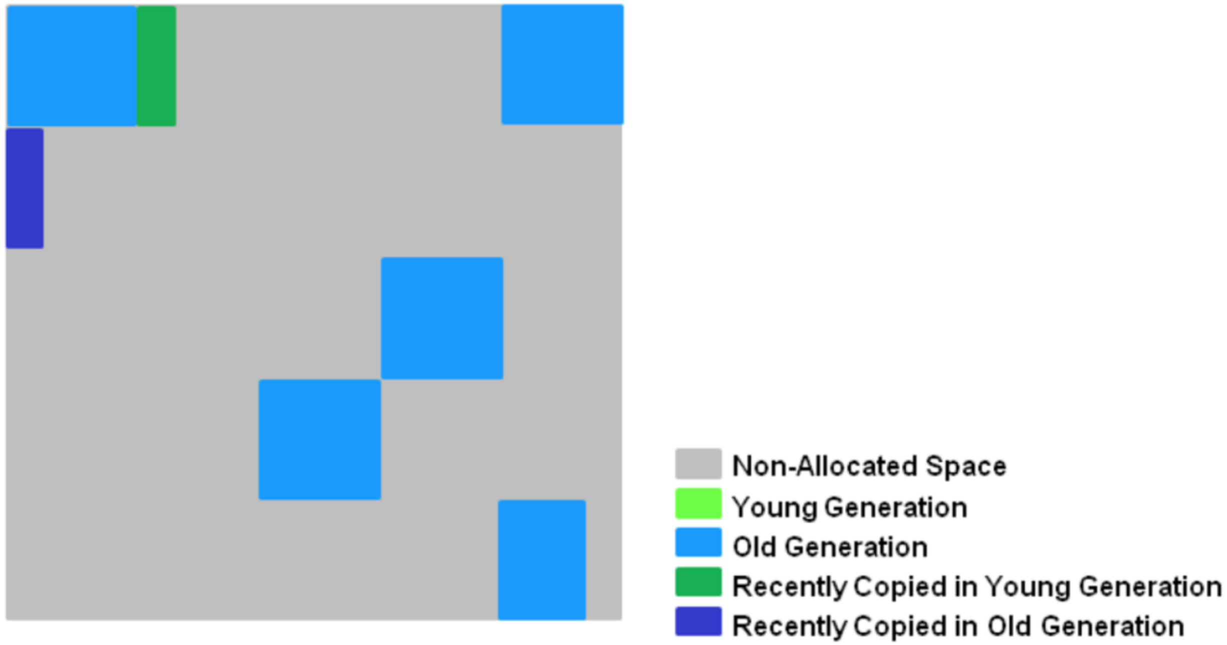
Major GC

Copying/Cleanup Phase

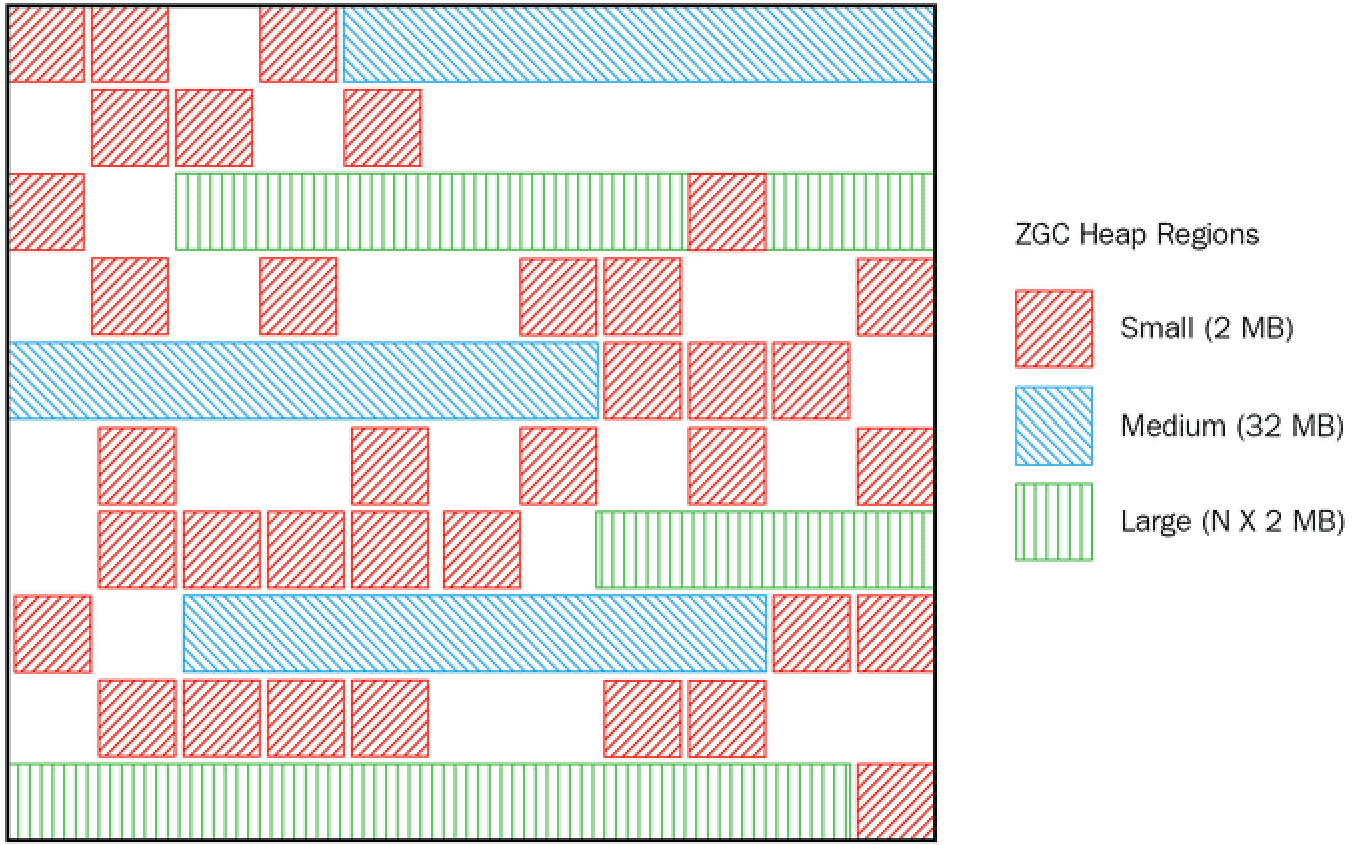

Non-Allocated Space
Young Generation
Old Generation
Recently Copied in Young Generation
Recently Copied in Old Generation

Major GC

After Copying/Cleanup Phase


Non-Allocated Space
Young Generation
Old Generation
Recently Copied in Young Generation
Recently Copied in Old Generation

Z Garbage Collectors (ZGC)


ZGC Heap Regions
Small (2 MB)
Medium (32 MB)
Large (N X 2 MB)