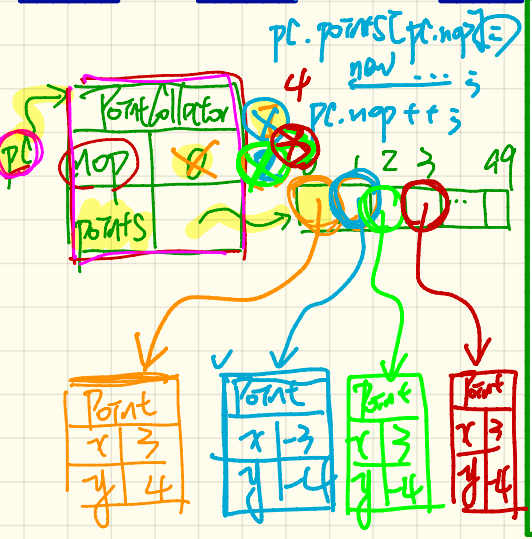
Monday March 18

Lecture 19

Programming Pattern: Mutator

```
class PointCollector {
  Point[] points; int nop; /* number of points */
  PointCollector() { points = new Point[100]; }
  void addPoint(double x, double y) {
    this.points[nop] = new Point(x, y); this.nop++; }
}
```

```
class PointCollectorTester {
  public static void main(String[] args) {
    PointCollector pc = new PointCollector();
    System.out.println(pc.nop);      /* 0 */
    pc.addPoint(3, 4);
    System.out.println(pc.nop);      /* 1 */
    pc.addPoint(-3, -4);
    System.out.println(pc.nop);      /* 2 */
    pc.addPoint(-3, -4);
    System.out.println(pc.nop);      /* 3 */
    pc.addPoint(3, -4);
    System.out.println(pc.nop);      /* 4 */
```



Circles.

the same

pc.add(Point(3,4)); Tutorial.

pc.ps[0].moveUp(3); null pointer exception

```
class PointCollector {
    int nop;
    Point[] ps;
    PointCollector() {
        this.ps = new Point[100];
    }
}
```

attribute nop is not initialized

default value: 0

pc → PC { nop: 1, ps: } → 0 1 2 .. 99

pc.ps[1].moveUp(3)

Point { x: 3, y: 4 }

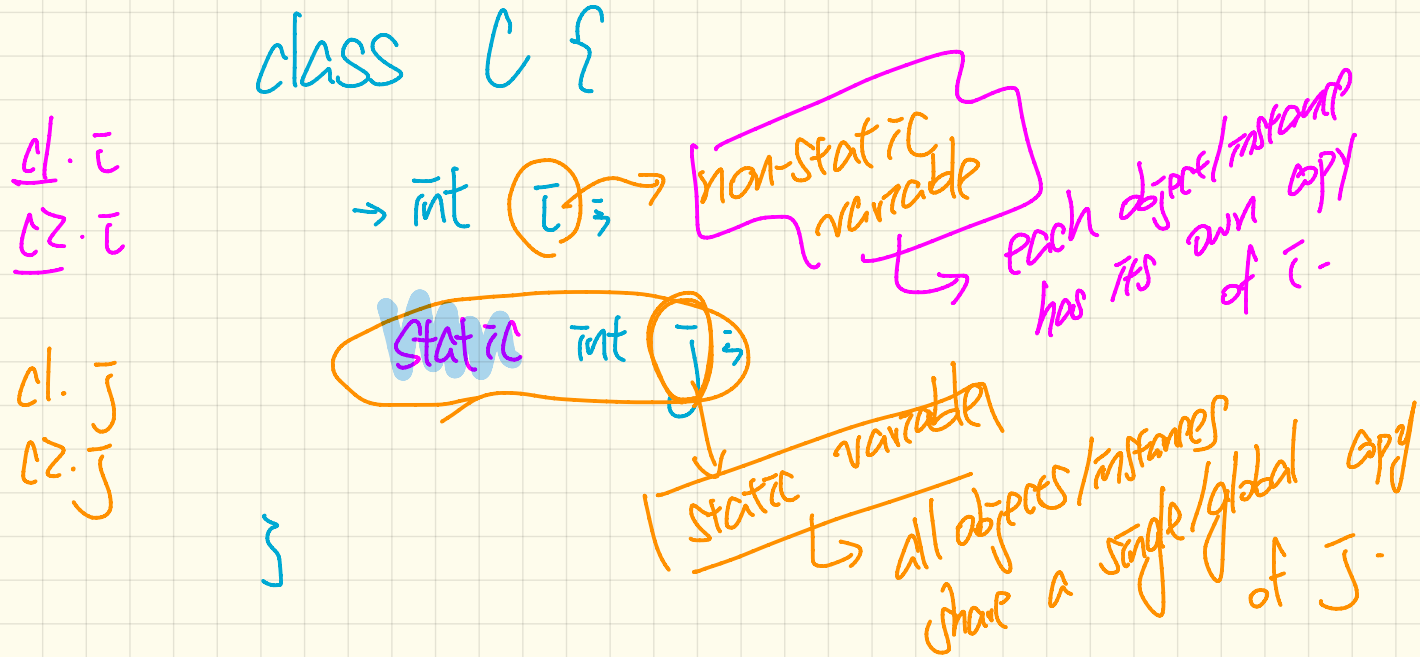
```
class PointCollector {
    int nop;
    Point[] ps;
    PointCollector() {
        this.ps = new Point[100];
        this.nop = 0;
    }
}
```

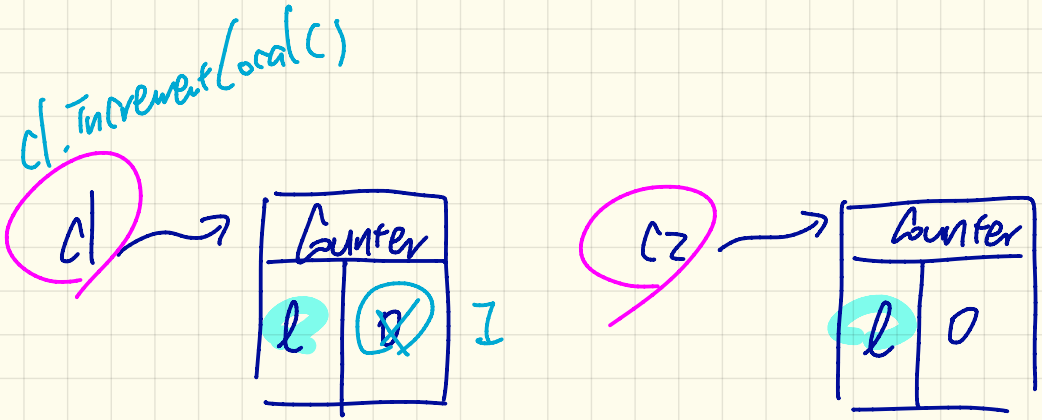
1.

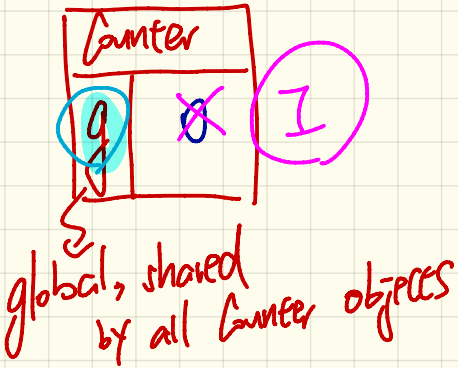

```
class PointCollector {
    int nop;
    Point[] ps;
    PointCollector() { ... }
    void addPoint(x, y) { ... }
}
```

```
pc this.ps[nop] = new Point(x, y);
this.nop++;
```

```
PointCollector pc =
    new PointCollector();
pc.addPoint(3, 4);
```


class C {
c1.i
c2.i
→ int i;
non-static variable
each object/instance has its own copy of i.
static int j;
c1.j
c2.j
static variable
all objects/instances share a single/global copy of j.
}


c1.Increment(local(c))
c1
Counter
l
x
1
c2
Counter
l
0


Counter
g
x
1
global, shared by all Counter objects