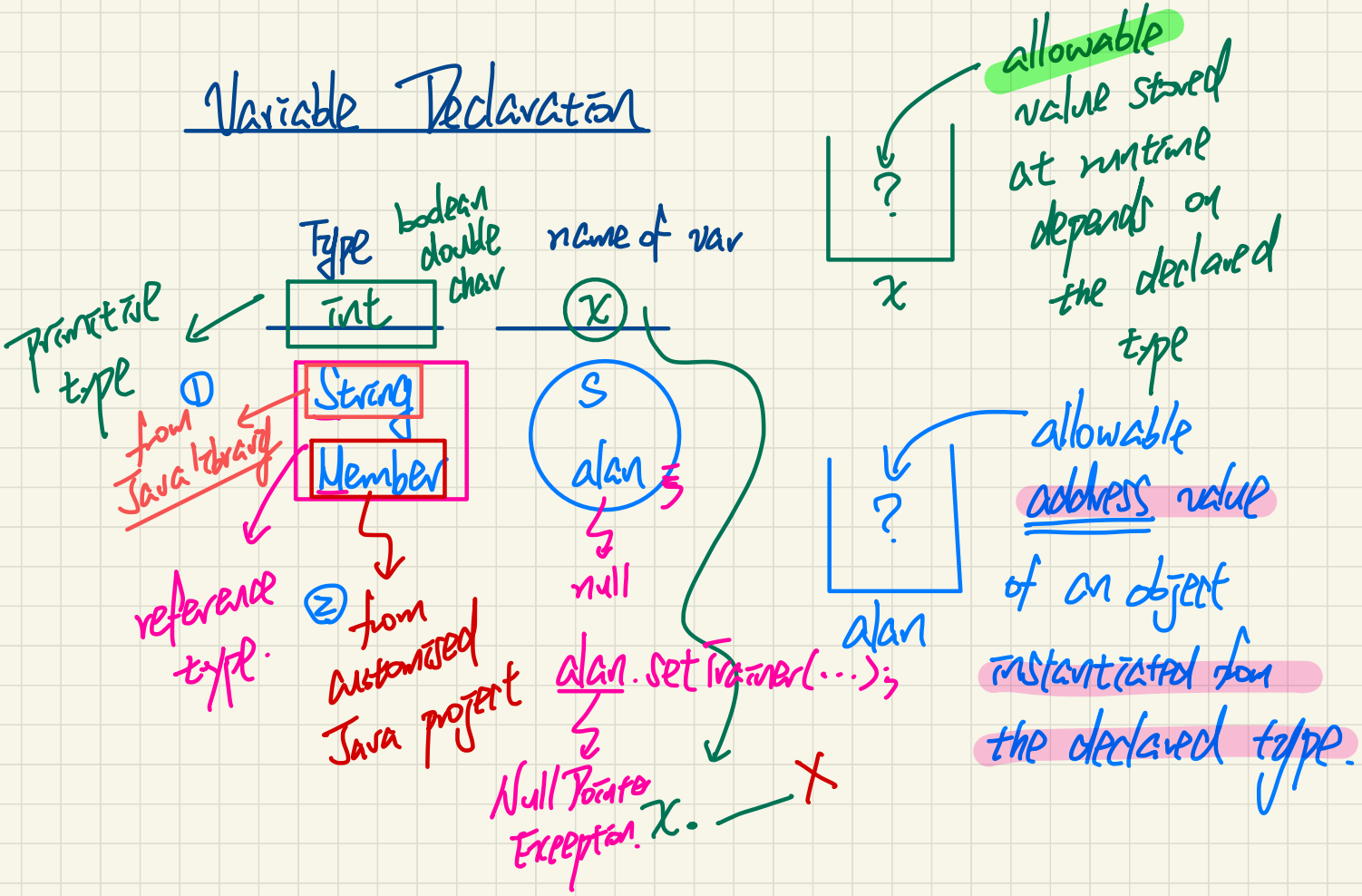
Lecture 4

Part C

Classes and Objects - Java Data Types, Anonymous Objects

Variable Declaration



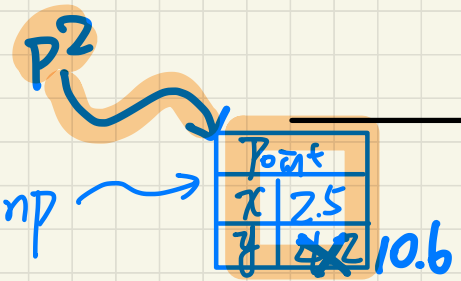
Reference-Typed Return Values

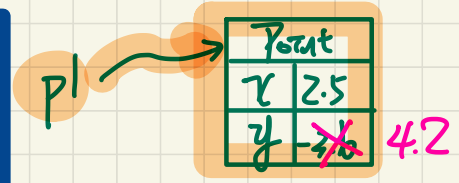
accessor (not modifying) — mutator (modifying) — context object

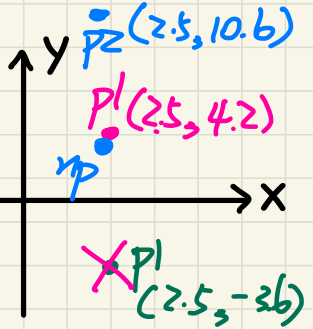
```
public class Point {
  public void moveUpBy(int i) { y = y + i; }
  Point movedUpBy(int i) {
    Point np = new Point(x, y);
    np.moveUpBy(i);
    return np;
  }
}
```

```
public class PointTester {
  public static void main(String[] args) {
    Point p1 = new Point(2.5, -3.6);
    p1.moveUp(7.8);
    Point p2 = p1.movedUpBy(6.4);
    System.out.println(p1 == p2);
  }
}
```

np.x = p1.x;
np.y = p1.y;



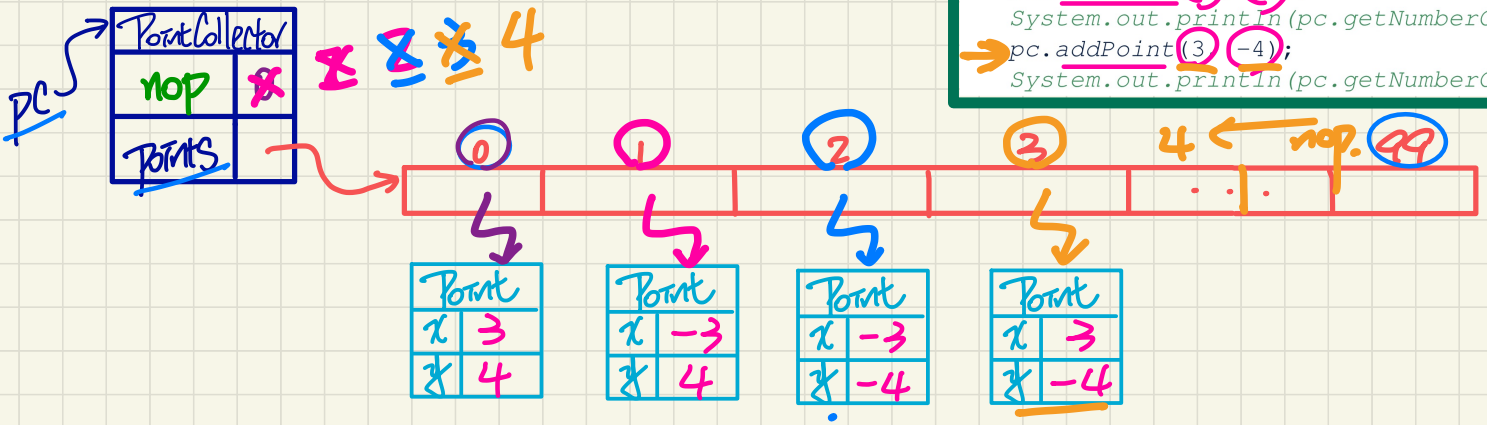




Programming Pattern: Mutator

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

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



Programming Pattern: Accessor

```
public Point[] getPointsInQuadrantI() {
  Point[] ps = new Point[this.nop];
  int count = 0; /* number of points in Quadrant I */
  for(int i = 0; i < this.nop; i++) {
    Point p = this.points[i];
    if(p.x > 0 && p.y > 0) { ps[count] = p; count++; } }
  Point[] q1Points = new Point[count];
  /* ps contains null if count < nop */
  for(int i = 0; i < count; i++) { q1Points[i] = ps[i] }
  return q1Points;
} }
```

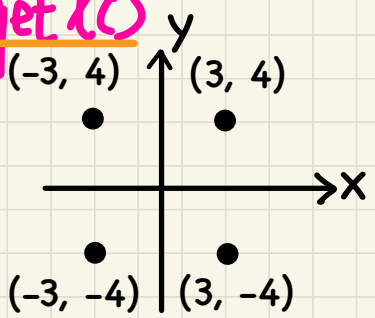
```
Point[] ps = pc.getPointsInQuadrantI();
System.out.println(ps.length); /* 1 */
System.out.println("(" +
  ps[0].getX() + ", " + ps[0].getY() + ")");
```

q1Points[i] = ps[0];

i: 0, 1, 2, 3

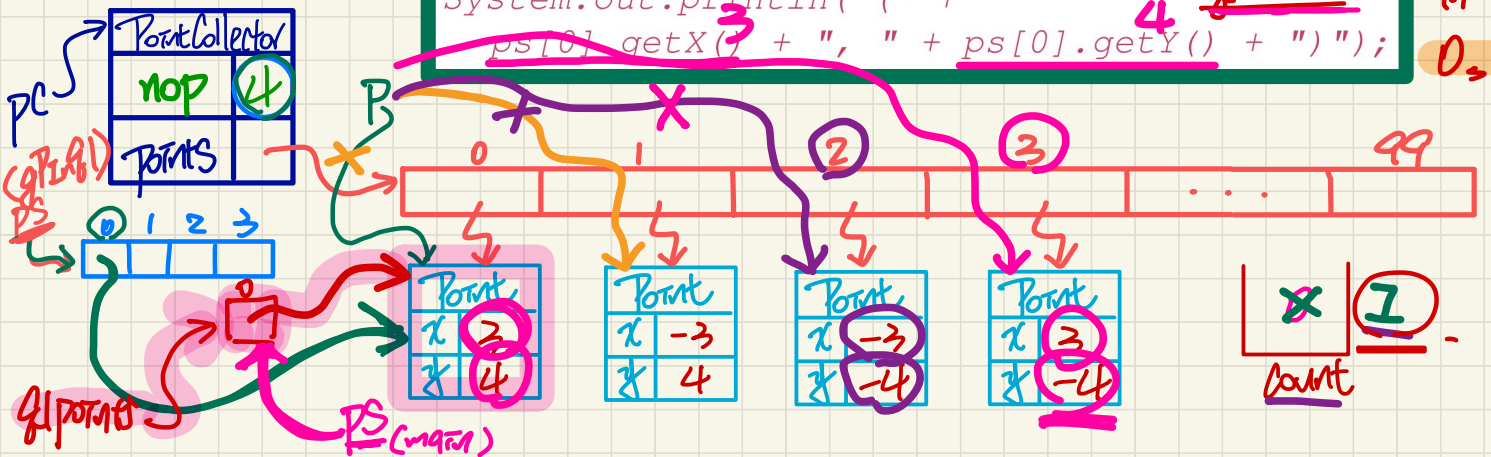
ps[0].getX()

c.o.



① there are "count" points in q1.

② these points are stored in indices: 0, ..., count-1



Lecture 4

Part D

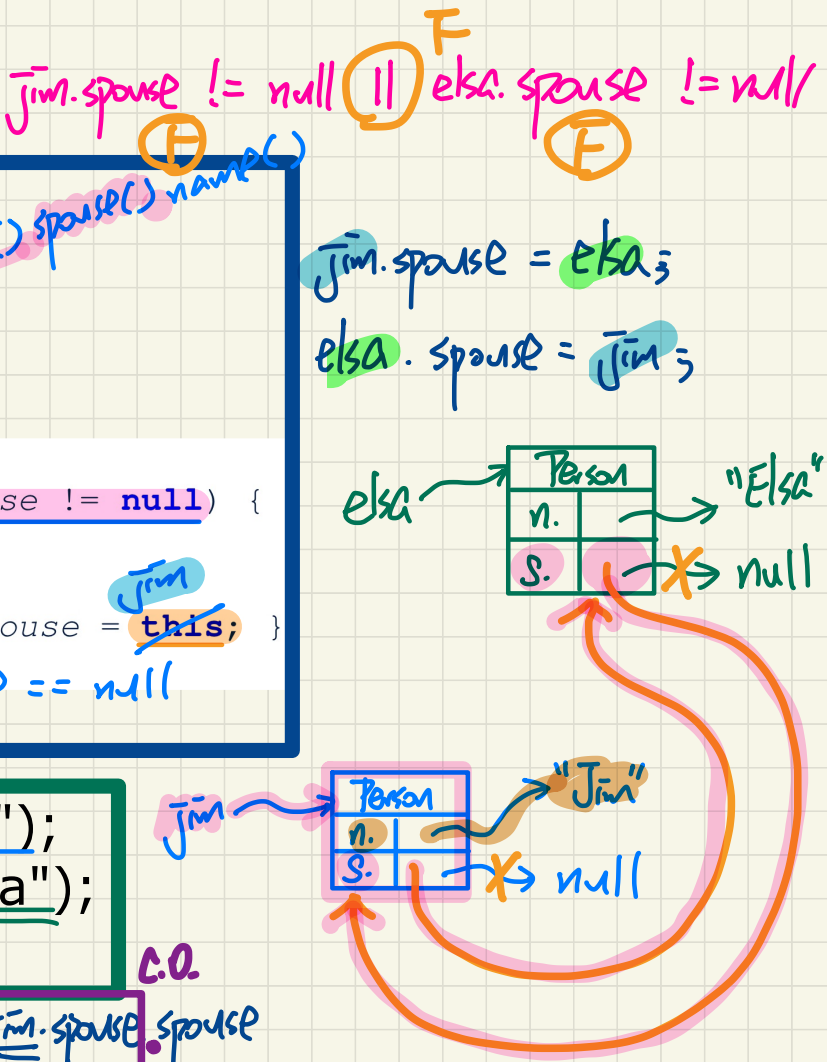
Classes and Objects - More Advanced Use of this

Example: Reference to this

```
public class Person {
  private String name;
  private Person spouse;
  public Person(String name) {
    this.name = name;
  }
  public void marry(Person other) {
    if (this.spouse != null || other.spouse != null) {
      /* Error: both must be single */
    }
    else { this.spouse = other; other.spouse = this; }
  }
}
```

this.spouse == null && other.spouse == null

```
Person jim = new Person("Jim");
Person elsa = new Person("Elsa");
jim.marry(elsa);
```

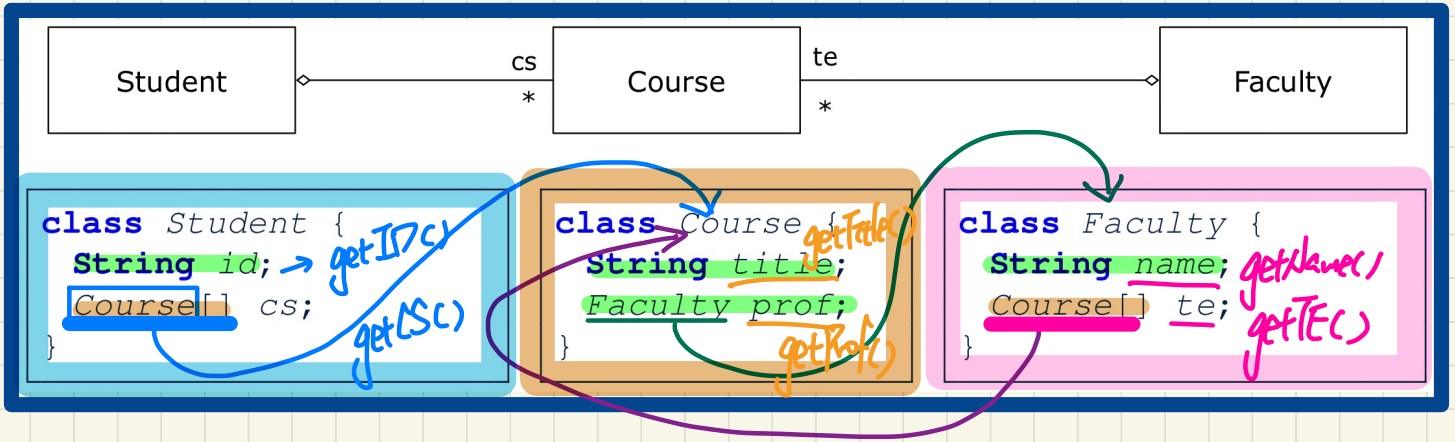


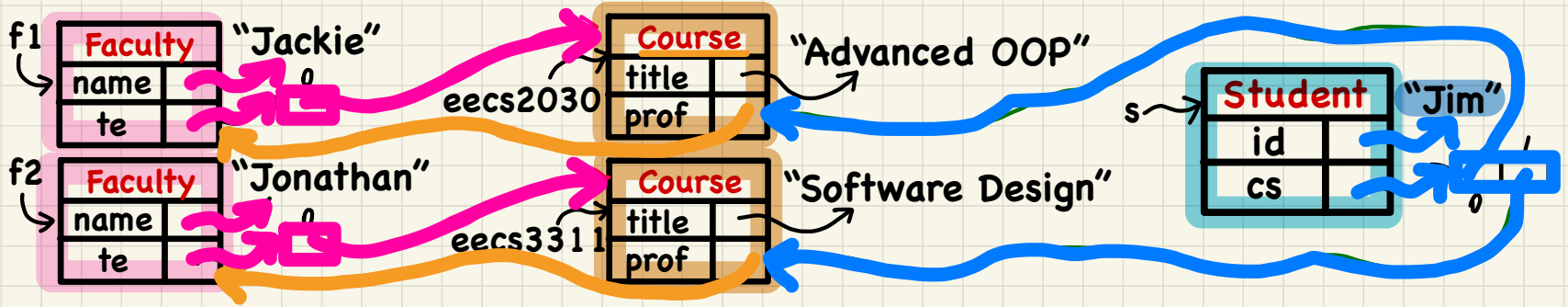
Lecture 4

Part E

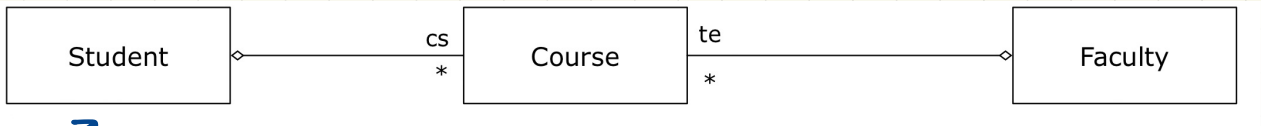
Classes and Objects – Navigating Classes via the Dot Notation

Object Structure: Student, Course, Faculty





Dot Notation for Navigating Classes (1)




```
class Student {
  String id;
  Course[] cs;
}
```

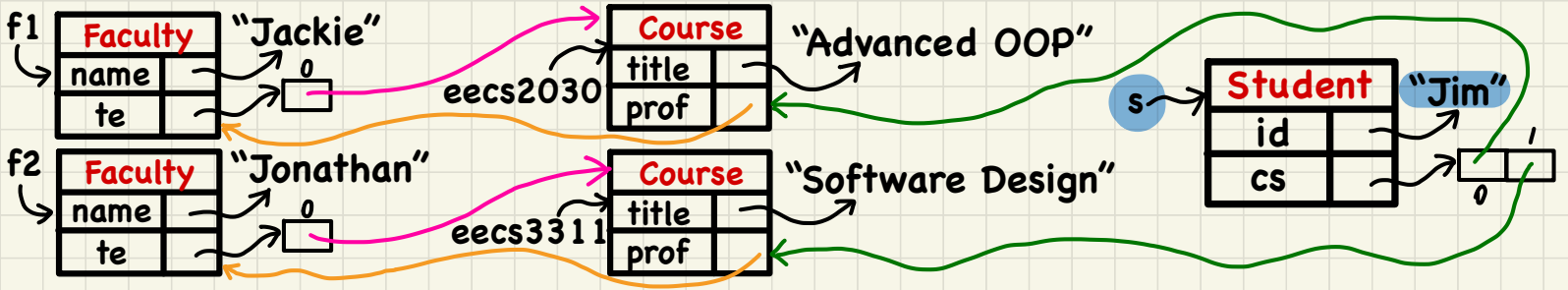
```
class Course {
  String title;
  Faculty prof;
}
```

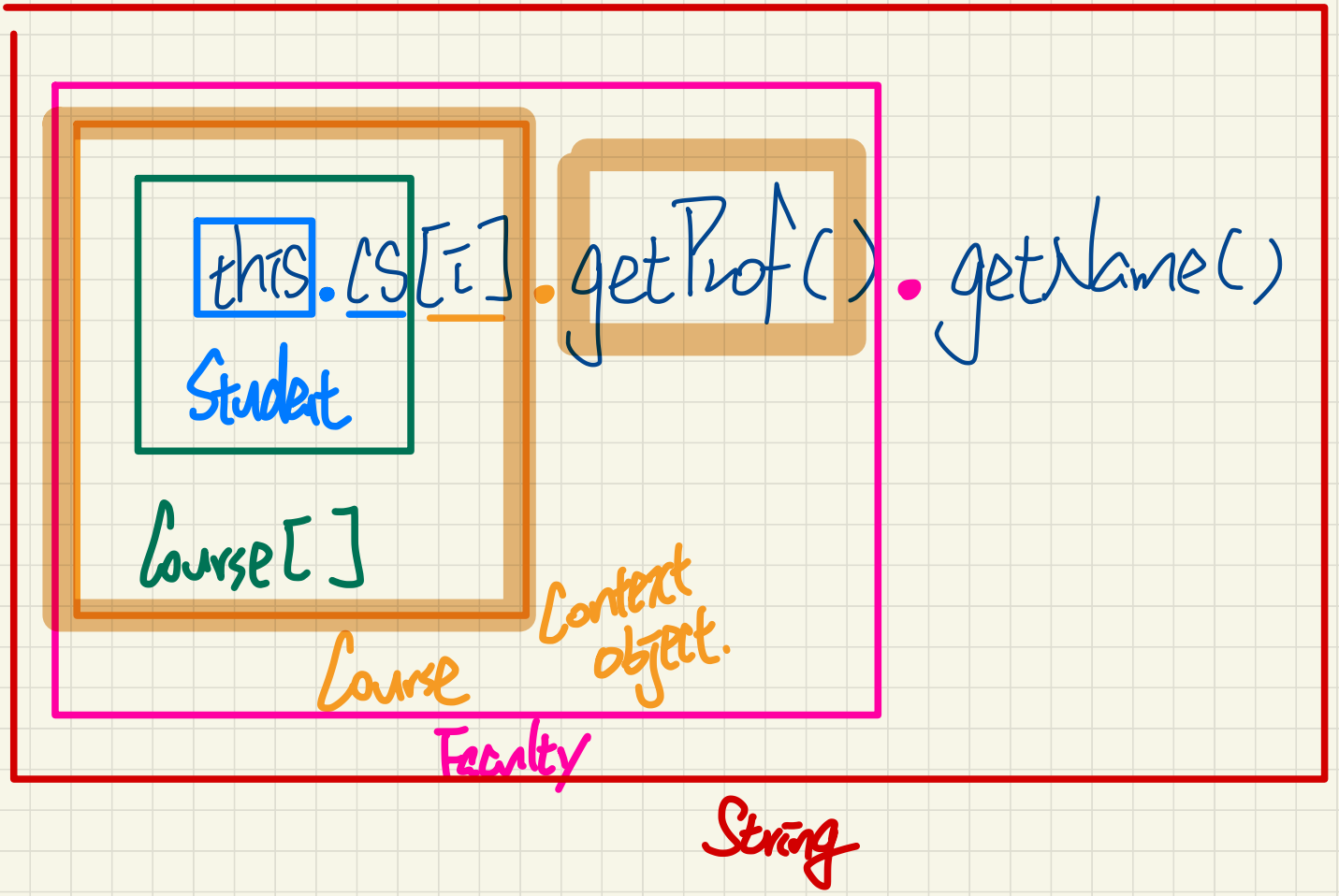
```
class Faculty {
  String name;
  Course[] te;
}
```

```
/* Get the student's id.
 */
String getID() {
    return this.id;
}
```

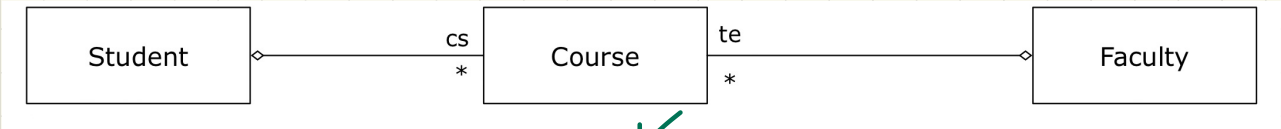
```
/* Title of ith course
 */
String getTitle(int i) {
    return this.cs[i].getTitle();
}
```

```
/* Name of
 * ith course's instructor
 */
String getName(int i) {
    return this.cs[i].getProf().getName();
}
```




this.CS[i].getProf().getName()
Student
Course[]
Course
Context object.
Faculty
String

Dot Notation for Navigating Classes (2)




```
class Student {
  String id;
  Course[] cs;
}
```

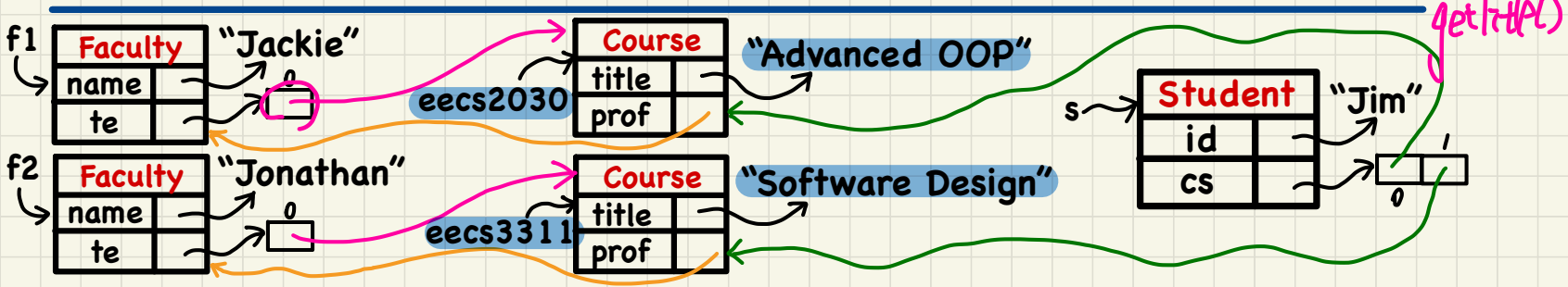
```
class Course {
  String title;
  Faculty prof;
}
```

```
class Faculty {
  String name;
  Course[] te;
}
```

```
/* Get course's title.
 */
String getTitle() {
  return this.getTitle();
}
```

```
/* Name of instructor
 */
String getName() {
  return this.prof.getName()
}
```

```
/* Title of instructor's
 * ith teaching course
 */
String getTitle(int i) {
  return this.getProf().getTE()[i].
}
```



Dot Notation for Navigating Classes (3)


```
class Student {
  String id;
  Course[] cs;
}
```

```
class Course {
  String title;
  Faculty prof;
}
```

```
class Faculty {
  String name;
  Course[] te;
}
```

```
/* Name of instructor
 */
String getName() {
  return this.name;
}
```

```
/* Title of instructor's
 * ith teaching course
 */
String getTitle(int i) {
  return this.te[i].getTitle();
}
```