# CS 213 : Software Methodology
# Spring 2017

*Sesh Venugopal*

Lecture 1: Jan 17
OOP – Constructors

# Constructor

A constructor <u>creates</u> an object. True or False?

FALSE. A constructor initializes an object.

In the statement `new X()`:

- the `new X` part creates an X object
- the `X()` part calls the no-arg constructor of X on behalf of the new object, to initialize it

When an object is <u>created</u> with `new`, its fields are initialized to their intrinsic default values (zero for int, null for object references, etc.). True or False?

TRUE.

# Constructor

```
public class Point { }
```

Will this class definition compile? Yes or No.

YES

So, how to create a new instance of `Point`?

```
new Point();
```

Really? But there's no constructor in the `Point` class ☹

Actually there is. The compiler throws in a default constructor that looks like this:

```
public class Point {
    public Point() { }
}
```

No arguments to constructor, nothing in the body

# Constructor

Given this definition of a Point class:

```
public class Point {
   int x,y;
   public Point(int x, int y) {
      this.x = x; this.y = y;
   }
}
```

Will this statement compile:

```
Point p = new Point();
```

NO. There isn’t a matching constructor in `Point`. (Default constructor is thrown in ONLY when there is no defined constructor.)

# Constructor

```
public class Point {
   int x,y;
   public Point(int x, int y) {
      this.x = x; this.y = y;
   }
   public Point(int x) {
      this(x,0);
   }
   public Point() {
      this(0,0);
   }
}
```

What do these statements do?

They call another matching (in argument sequence/types) constructor in the class – in this case the first constructor