Welcome to CS108

Dr. Patrick Young

Before Objects

```
CHARACTER*8   NAME
INTEGER AGE
```

Fortran IV Code

Before Objects

```
CHARACTER*8 CSNAME
INTEGER CSAGE
```

Fortran IV Code

Before Objects

```
C This is Customer Data
   CHARACTER*8 CSNAME
   INTEGER CSAGE
```

Fortran IV Code

Records or Structs

```
01 CUSTOMER-RECORD
    05 NAME PIC A(10)
    05 AGE 999
```

Cobol
Code

Records or Structs

```
struct customer {
    char* name;
    int   age;
    ...
}
```

C Code

Records or Structs

```
struct customer {
    char* name;
    int   age;
    ...
}

void updateAddress(struct customer cust)
void printInfo(struct customer cust) ...
```

C Code

Classes

```
class Customer {
    private String name;
    private int   age;
    ...
    public void updateAddress() …
    public void printInfo() …
}
```

Java Code

What does this do?

```
void example() {
    Student st;
    …
}
```

What's the difference?

```
void example() {
    Student st;
    …
}

void example2() {
    Student* pSt = new Student;
    …
}
```

C++ Code

What's the difference?

```
void example3(Student &st) {
   …
}

void example4(Student st) {
   …
}
```

C++ Code

C++ Call-by-Reference vs. Call-by-Value

```
void incrementOne(int& a) {
  a++;
}

void incrementTwo(int a) {
  a++;
}
```

C++ Call-by-Reference vs. Call-by-Value

```
void incrementOne(int& a) {
  a++;
}

void incrementTwo(int a) {
  a++;
}
```

```
int x = 1;
incrementOne(x);
```

Vs.

```
int x = 1;
incrementTwo(x);
```

Java Call-By-Value with Primitive

```
public class CallByValueExample {

  public static void increment(int a) {
    a++;
  }

  public static void main(String[] args) {
    int x = 1;

    increment(x);
    System.out.println(x);
  }
}
```

Java Call-By-Value with Reference Type

```
public class CallByValueExample2 {

  public static void increment(Point a) {
    a.x++;
    a.y++;
  }

  public static void main(String[] args) {
    Point p = new Point(1,1);

    increment(p);
    System.out.println("x=" + p.x + ";y=" + p.y);
  }
}
```

What does this do?

```
public class CallByValueChange {

  public static void change(Point a) {
    a = new Point(5,5);
  }

  public static void main(String[] args) {
    Point p = new Point(1,1);

    change(p);
    System.out.println("x=" + p.x + ";y=" + p.y);
  }
}
```

Copying Objects

```
Foo x = new Foo(1);
Foo y = new Foo(2);

x = y;
```

What got copied?

Copy Constructors

```
Foo x = new Foo(1);
Foo y = new Foo(x);
```

Not the same as

```
x = y;
```

MyPoint Example

```
public class MyPoint {
   public int x;
   public int y;

   MyPoint(int x,int y) {
      this.x = x;
      this.y = y;
   }
}
```

Copying MyPoint

```
MyPoint p1 = new MyPoint(5,5);
MyPoint p2 = p1;

p2.x = 15;
```

What is the value of p1.x and p1.y now?

MyPoint Copy Constructor

```
public class MyPoint {
   …

   MyPoint(MyPoint p) {
      this.x = p.x;
      this.y = p.y;
   }
}
```

Copying MyPoint

```
MyPoint q1 = new MyPoint(5,5);
MyPoint q2 = new MyPoint(q1);

q2.x = 15;
```

What is the value of q1.x and q1.y now?

Comparing

```
MyPoint p1 = new MyPoint(5,5);
MyPoint p2 = p1;
```

Does p1 == p2?

```
MyPoint q1 = new MyPoint(5,5);
MyPoint q2 = new MyPoint(q1);
```

Does q1 == q2?

Writing an Equals Method

```
public class MyPoint {
   …
   public boolean equals(MyPoint p) {
      return (x == p.x) && (y == p.y);
   }
}

MyPoint q1 = new MyPoint(5,5);
MyPoint q2 = new MyPoint(q1);
```

Does what is q1.equals(q2)?

*** Depending on your planned use, you may want to write a more general version that takes an Object as a parameter not a MyPoint.**

String Comparison

```
String s1 = new String("Stanford");
String s2 = new String("Stanford");
```

Does s1 == s2?

Does s1.equals(s2)?

Multi-Dimensional Array

```
String[][] cartoons =
   {{"Homer", "Marge", "Bart", "Lisa", "Maggie"},
    {"Peter", "Lois", "Meg", "Chris", "Stewie", "Brian"},
    {"Cartman", "Kenny", "Stan", "Kyle"}};
```

*** Inspired by official Sun Java Example**