30 September

Classes

either: - a program - a template for new types of objects

object is an "entity" that combines state and behavior

object-oriented program: programs that perform as interactions between objects. Keep state changes in well-defined objects

Classes contain: - fields: variable, part of object state. Each object has its own copy of each field. - access fields by *dereferencing* with the dot notation (ex: `s.length()`) - methods

```
public type name(parameters) {
  statements;
}
```

Methods can be:

- accessor
- mutator

(point class example) (how to access fields from inside same class)

Inheritance

Formalizes hierarchies of how data is structured

```
public class Animal {
  String name;
  int happiness;
  boolean newDay = true;

  public int getName() {
    return name;
  }

  public int getHappiness() {
    return happiness;
  }

  public void interact() {
    happiness = happiness + 1;
    newDay = false;

  public void sleep() {
    newDay = true;
  }
}
```

To create a *subclass* inheriting this *superclass*:

```
public class Cow extends Animal {
  boolean milked = false;

  public void milk() {
    milked = true;
  }
```

```
  public void interact() {
    happiness = happiness + 2;
  }

  public void grumpy_interact() {
    super.interact();
  }
}
```

- Multiple levels of inheritance are allowed; multi-inheritance is not.
- Constructors are not inherited: if the super-class has a constructor defined, so must the sub-classes

Linked List

- intro to new friend
- talk about what you know about linked lists
- assume we're going to implement a linked list with two classes: `MyLL` and `ElementLL`

```
public class MyLL {
  public ElementLL first;
  public ElementLL last;

  public MyLL(int[] inputList) {

  }
}
```

- what fields and/or methods does the `ElementLL` class need?
- what steps would you take to implement the constructor for `MyLL`?