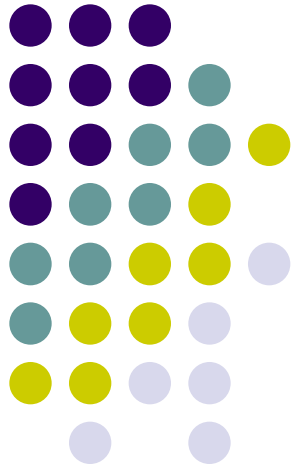
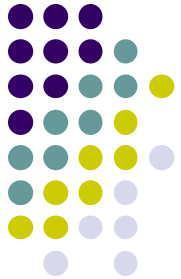
Instance Methods



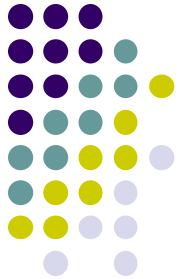
Recall:



- An *object* is a particular instance of a class.
  - Think of objects as the things being produced by your “factory” or class

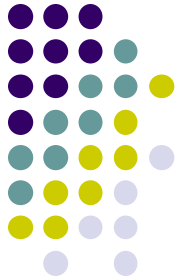
- **An object is more than simply a collection of data. Objects can also have functionality!**

Recall: the fraction class



- The class *fraction* is an expression composed of a numerator and a denominator in the form **n/d.**

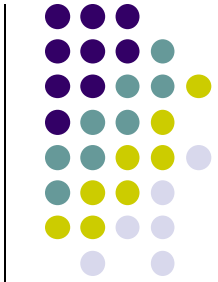
- In addition, there are mathematical properties that a fraction must possess.
  - For example:
    - A fraction is in the form n/d
    - The denominator can not be 0
- A computer program may not necessarily adhere to these rules.
  - That is:
    - We can access the num and den and multiply them together
    - We assign the den a value of 0.



- Therefore objects need to have functionality as well as fields.

- This functionality is controlled through 2 types of methods:
- Class methods and Instance methods

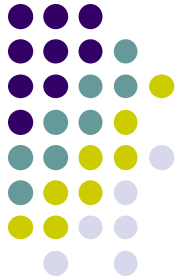
Instance Method



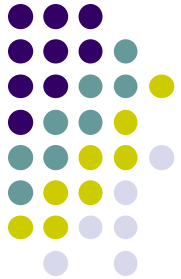
- Is a method contained within a class to control the functionality of the objects that are created.
  - Think of instance methods as the rules that dictate how your object is to function
  - If a class is a blueprint of an object. Instance methods are the instructions.

- We can now use instance methods to prevent the user of the program from multiplying the num and den or assigning the den a value of 0

Example 1:

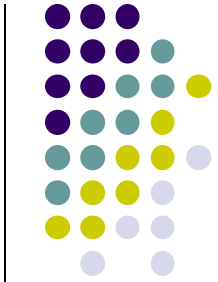


Create an instance method called *quotient* that will return num divided by den.



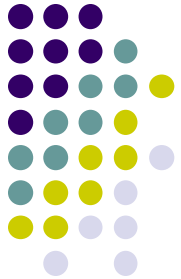
```
public class fraction {
    int num;
    int den;

        // quotient that will return num divided by den
        // notice that no parameters are necessary because the
    fields already belong to the object. That is, they are
   implicit

        public double quotient(){
            double  q;
            q = (double)num/(double)den;
            return q;
        }


}
```

Note:



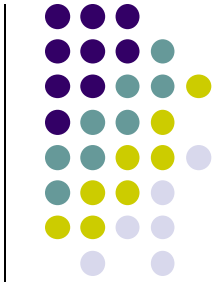
- In our instance method *quotient*, there is no modifier **static.**
  - Any method with the static modifier is a class method (The ones from the last unit)
  - Any method without the static modifier is an instance method.

Comments



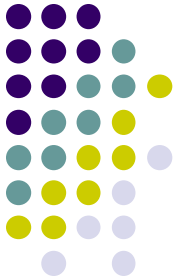
- All instance methods should contain the following comments:
  - A description of the parameters being received and their purpose
  - How the field(s) is/are manipulated. How does this operate.
  - what (if any) value is returned. Explain why that value is returned

Calling our instance method



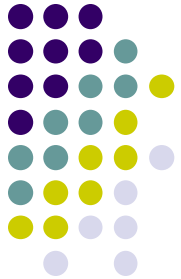
- To call an instance method, we first need to create an object in our main program and use that object to access the instance method.
  - Remember, an instance method is part of how an object behaves, so therefore we need to first create an object.
  - The instance method will then apply directly to the object that is calling it

Syntax



To call an instance method, we use the following syntax

<object identifier>.<instance method identifier>(<parameter list>);



For example:

```
public static void main(String[] args) {
        double result;

        fraction f = new fraction();
        f.num = 7;
        f.den = 9;

        result=f.quotient();
        System.out.println(result);
}
```

- // When calling an instance method:
  - we create an object called fraction
  - Using that object, we invoke the instance method.