

6.170

Clarifying Overloaded Language



MIT EECS

(Subjective) Remarks on David Ziegler's Lecture



- **Beyond-the-course software: it's up to you**
 - Having people use your software is the most important coding experience that you can have after 6.170
 - UROP, open-ended labs, SIPB
 - But you have to choose carefully to gain such experience
- **Press-the-button system**
 - David's conclusion: throw away small hacks
 - Another possible conclusion: systems are built without a specification and design because their birth goes unnoticed
 - Similar effect if carefully designed for a narrow application but used more widely

Cleanup of Overloaded English



- Abstract what?
- Abstract state
- Abstraction function
- Abstract methods
- Abstract classes

Abstract State



- Not a Java concept

```
class Rat {  
    private int num;  
    private int denom;  
    ...  
}
```

[num=30, denom=10]

Abstract state is the rational number 3

Abstraction Function



- Not a Java concept
- AF: `Rat` \rightarrow mathematical rational numbers
- The domain: objects of class `Rat` that satisfy the representation invariant (e.g., `denom != 0`)
- $\text{AF}(\text{num}, \text{denom}) = \text{num}/\text{denom}$
- $\text{AF}(\text{representation}) = \text{abstract state}$

- The AF is a function in the *mathematical* sense that maps the representation of an object to its abstract state

Classes



```
class point extends AbstractPoint {  
    private int x;  
    private int y;  
    public int x() { return x; }  
    ...  
}
```

A class can be instantiated:

```
Point p = new P(3,4);
```

Abstract Classes



```
abstract class AbstractPoint {  
    abstract public int x(); // just a promise  
    abstract public int y(); // another  
    public double maxNorm() {  
        return Math.sqrt(x()*x() + y()*y());  
    }  
}
```

Cannot be instantiated:

```
AbstractPoint p;  
p = new P(3,4);
```

Interfaces = Purely Abstract Classes



```
Interface AbstractPoint {  
    abstract public int x(); // just a promise  
    abstract public int y(); // another  
    public double maxNorm();  
}
```

Cannot be instantiated; multiple supertypes:

```
class Rat implements Number, Comparable  
interface RatInt extends Number, Comparable
```