

CMSC 433 – Programming Language Technologies and Paradigms Spring 2007

Abstraction
Feb. 27, 2007

Data Abstraction

- Data abstraction = objects + operations
 - List + { addFirst, addLast, removeFirst, ... }
 - Set + { add, contains, ... }
- Categories of operations
 - Constructors (creators/producers)
 - Mutators/Modifiers
 - Observers

Abstraction Function

- Specification for data structure is abstract
- Implementation of data structure is concrete
- How do you know if implementation meets the spec?
- Abstraction function : concrete \rightarrow abstract
 - Specifies how the representation of an abstract data type is interpreted as an abstract value.

Example

```
class IntSet { int[] elts; ... }
```

– $AF(s) = \{ s.elts[i] \mid 0 \leq i < elts.length \}$

- You always need an abstraction function when you build a data abstraction
 - Often it's implicit

Representation Invariant(s)

- A constraint that characterizes whether an instance of an abstract data type is well formed,
- Constraint must hold
 - After the constructor has finished
 - Before and after each operation

```
class IntSet {  
    // rep inv: elts contains no duplicates  
    int[] elts; ...  
}
```

- Part of the (internal) specification

Implementing the Rep Invariant

- Interesting idea: Write a function to check the rep

```
public boolean repOK() {  
    ...check for duplicates in elts...  
}
```

- Where can you use this?
 - Can add wherever you expect rep to hold
 - Can call during unit testing
- Cost?