Object/Class Design

Slides adapted from Craig Zilles

Design is hard

- Design is an art, not a science
- Large/infinite design space, not enumerable
- Requirements, constraints, trade-offs, and priorities

- You get better with practice / experience / seeing good design / hearing critiques of others designs

Virtues of a good design (software)

- Manages complexity
- Loose coupling
- Reusability
- Ease of maintenance
- Standard techniques
- Extensibility
- Lots of Uses
- Few Dependencies

Good Design Manages Complexity

- **“Seven plus or minus two” (Miller’s Law)**
- **The goal of all software-design techniques**
  - **Break complicated problems into simple problems**
- **Separation of concerns**
  - Focus on one at a time

Keep Coupling Loose

- small interfaces (few methods, few arguments/method)
- obvious (interactions through parameter passing)
- flexible

How hard was third code review assignment?

A) Easy

B) Moderate

C) Challenging

D) Unreasonable

How long did third assignment take?

A) Less than 3 hours

B) 3 to 6 hours

C) 6 to 9 hours

D) 9 to 12 hours

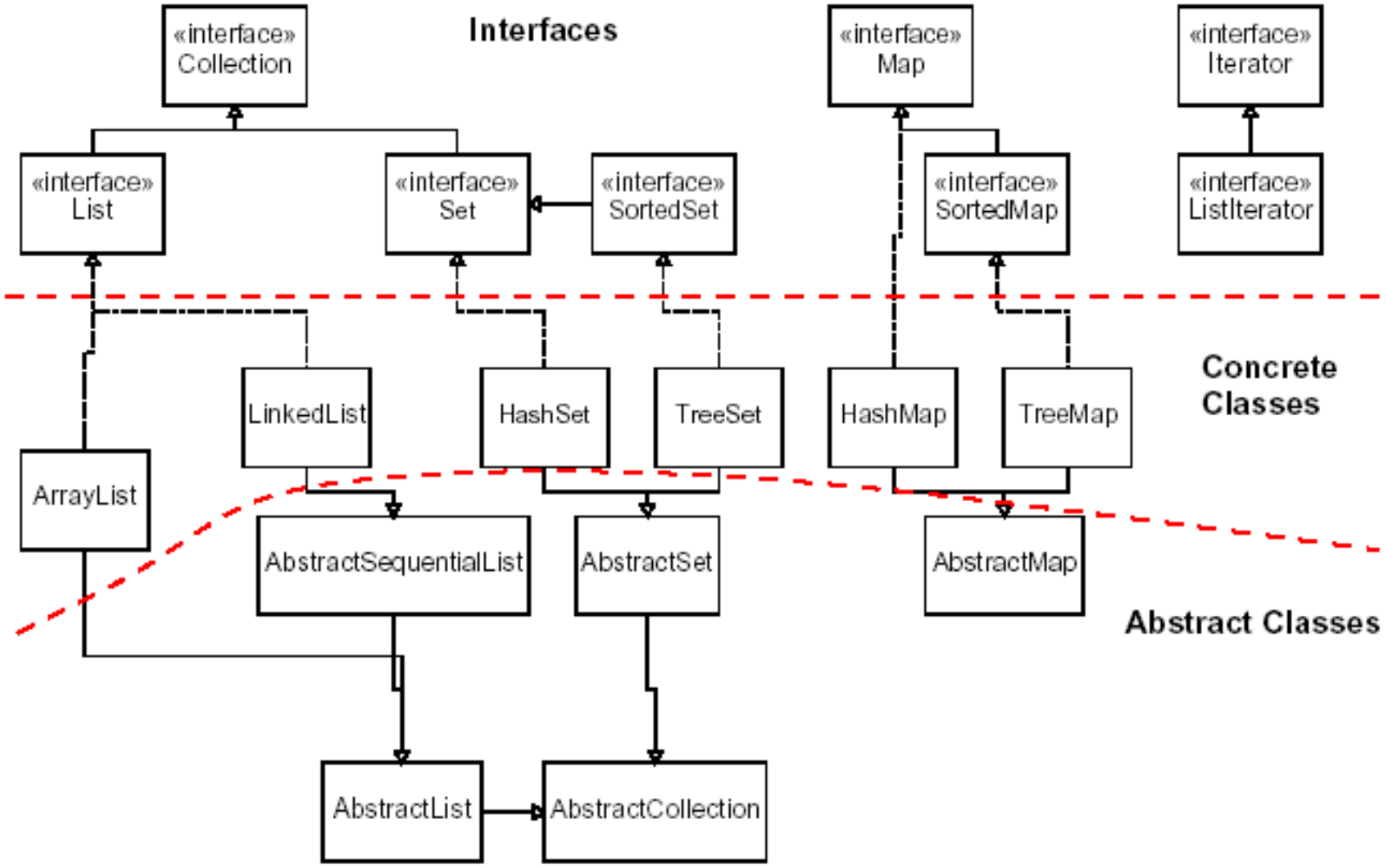
E) More than 12 hours

Abstract Data Types

- **Define a class based around conceptual structures**
  - Encapsulation / information-hiding
  - Make interfaces more informative (self-documenting)
  - Easier to reason about correctness

- **Treat even simple items as ADTs**
  - Good for extensibility

Java Collection Framework



Map

- Allows lookups from one kind of object to find another object

```
Map<KeyType, ValueType> myMap =
        HashMap<KeyType, ValueType>();

KeyType key = …;
ValueType value = …;
myMap.put(key, value);
ValueType lookup = myMap.get(key);
assert lookup == value;
```

Map Interface

`put(k,v)`	Associate `v` with `k`
`get(k)`	The value associated with `k`
`size()`	The number of pairs
`isEmpty()`	Whether it is empty
`remove(k)`	Remove the mapping for `k`
`clear()`	Remove all mappings
`containsKey(k)`	Whether contains a mapping for `k`
`containsValue(v)`	Whether contains a mapping to `v`
`keySet()`	Returns the Set view of keys
`entySet()`	Returns the Set view of keys and values

Inheritance can provides 2 things

- **Shared interface:**
  - Public methods
  - Ensure methods have consistent meaning in derived class
    - Liskov Substitution Principle

- **Shared implementation**
  - Code in shared super class, not replicated in each derived
  - Could be private data/methods

hasA vs. isA relationship

Which is a candidate for inheritance?

A) hasA relationship

B) isA relationship

C) both hasA and isA relationships

D) neither hasA and isA relationships

Inheritance vs. Interfaces

- Inheritance should be a isA relationship

- Interfaces are for capabilities (“mixin”s)

Designing Good Interfaces

- Sufficiently Expressive
- General
- Minimal