Lecture 8
Java I/O
Testing & Debugging 101
What is an interface?

- In its most common form, an interface is a group of related methods with empty bodies.
- Similar to an abstract class where everything is abstract.
- A *contract* that binds the interface and any class implementing it.

```java
public interface SocialEntity {
    public String getName ();
    public long getId ();
}
```
Assignment 7 Review

- Classes
  - Always write the constructor first
  - Do not expose underlying structure; define methods to manipulate member variables

```java
ArrayList<Network> networks;
...
public void addNetwork (Network n) {
    networks.add (n);
}
```
Assignment 7 Review

- All objects must have a type & be created using `new()`

```java
ArrayList<Network> networks;

....

public MyClass () {
    Networks = new ArrayList<Network>();
}
```
• *this* still confusing for some

  – *this* is a reference to the *current object*
  – Use it in constructors

```java
String name;
long iid;

public Person (String name, long iid) {
    this.name = name;
    this.iid = iid;
}
```
Refresher

Intro/Overview
– compilation, execution
– Java Basics:
  – Structure & Syntax, Variables, Types, & Operators

Control Flow
– Methods & Conditionals, Loops & Arrays

Object-oriented Programming (OOP):
– Objects & Classes
– Inheritance & Abstraction:

Collections

Exceptions

Brief Intro to Software Design
Outline

• Java I/O
• Testing & Debugging 101
  – Assertions
  – Eclipse debugger
Java I/O

- Package for input / output operations
- Focuses on *streams* of data
- You can Read from (input) and Write to (output) a stream
Java I/O

• Ways to access data
  – Streams
  – Network
  – File
  – Etc.

• Output
  – System.out
  – System.err
  – Network
Java I/O

- Different ways to access data
- Streams
  ```java
  int nextInt = myStream.read()
  ```

- Readers / Writers
  - Special classes to read / write `char[]`

- These can also be `buffered`
Java I/O: A Tour

Java I/O: Example

- Reading text from a file

```java
try {
    BufferedReader in =
        new BufferedReader(new FileReader("infilename"));
    String str;
    while ((str = in.readLine()) != null) {
        process(str);
    }
    in.close();
} catch (IOException e) {
    // handle the potential exception
    e.printStackTrace();
}
```
Testing & Debugging 101

• Include simple tests to check that your program is running as it is supposed to run

• Debugging tools allow to run your code step by step, check the state of the variables etc.
Java Assertions

- Assertions allow you to test your assumptions about your program
- Each assertion contains a boolean expression that you believe will be true when the assertion executes. If it is not true, the system will throw an error

```java
assert booleanExpression;
OR
assert booleanExpression : messageValue;
```
public static double fahrToCelsius(double fahr) {
    assert fahr >= -459.67d : "temp < abs zero";
    double celsius = (fahr -32) / 9 * 5;
    return celsius;
}
Java Assertions

- Normally not intended for end-users
- By default, assertions are disabled at runtime
- Command-line switch to enable assertions:

  `-enableassertions` or `-ea`
Debugging with Eclipse

Courtesy of The Eclipse Foundation. Used with permission.
Assignment 8

- Magic squares!

- Read two files
- Check that all rows sum to the same constant!
Grades

- Please verify that your assignment grades match what you expect

- Click on *Gradebook* on course webpage
Course Evaluation

- Please evaluate the course to help us make it better in the future:


- Feedback from people who dropped the course very useful too

Thanks for attending!