Producing Production Quality Software
Lecture 6b: Error Handling with Java: Exceptions and Assertions
Prof. Arthur P. Goldberg
Fall, 2005

Java Exceptions, Review, Catch

\begin{verbatim}
try
  block
catch (exception_type1 identifier)
  block
catch (exception_type1 identifier)
  block
  ...
finally
  block
\end{verbatim}

Invoking a Method That Can Throw an Exception

Choices of action
  Catch the exception, handle it
  Catch the exception, and throw a new exception
    (which must be in your own throws clause)
  Declare the exception in your own throws clause,
    and let the exception pass through

Bad Use of Exceptions

\begin{verbatim}
try
  block // all your code
catch (Exception e)
  block
\end{verbatim}

This loses all the information in individual exceptions
Use Exceptions for Unexpected Error Conditions

- Catch most specific exception possible
- Catch clause design
  - If the error can be ‘fixed’, then ‘fix’ the error
  - Else, if error could be fixed reported by calling code, then throw exception
  - Else, report the error
    - If execution cannot continue, then exit
- Generally, error reporting should be centralized, so things like level of error detail, whether to log, language of error messages, etc., can be controlled from one place.

Java Exceptions, Review, Throw

```java
public class SomeException extends Exception {
    // ... } ...

    void method () throws SomeException {
        // ...
        if (AProblem)
            throw new SomeException( arg1, arg2);
    }
```

Tension

Good error handling, vs.
Code overwhelmed with exceptions

Try to find a balance

The End