CMSC 131
Object-Oriented Programming I

Exceptions II
Dept of Computer Science
University of Maryland College Park

This material is based on material provided by Ben Bederson, Bonnie Dorr, Fawzi Emad, David Mount, Jan Plane
Overview

- Exceptions
- Password Example
Exceptions

- You can have multiple catch clauses
  - Example: Multiple.java
  - How do we know the exceptions thrown by the methods? By using the Java API

- Throwing Exceptions
  - You can throw exceptions using throw
    - Notice you can define your own exceptions and throw them
    - You can throw exceptions defined by Java
    - An exception can be rethrown
  - Example: Throwing.java

- Important: exceptions should be used for handling errors and not for implementing solutions to problems

- Never leave the catch clause empty
  - If you don’t know what to place, the call printStackTrace
  - If you want code to compile, even though you have not implemented, then use:
    throw new UnsupportedOperationException("You must implement this method.");
There are two types of exceptions:

- **Checked**
  - We need to do something about them
  - We need to “catch” or “declare” them

- **Unchecked**
  - We don’t need to do anything about them

So far our examples have used unchecked

Let’s see an example of code that requires handling a checked exception

**Example:** Checked.java

- What happens if you remove the try/catch block?
- What alternative we have if we don’t want to provide a try/catch block

We can define our own exceptions

- Let’s define one
Examples

- Validating Integer
  - Let’s write a program that validates the user has entered a valid integer. Use `Integer.parseInt`, exceptions, and `JOptionPane`.
  - What happens when users select cancel?