COMP 110/L Lecture 22

Kyle Dewey

Outline

- Exceptions

Exceptions

Recall

```
int[] array = new int[3];
int result = array[27];
```

-What happens if this code snippet is run?

Recall

```
int[] array = new int[3];
int result = array[27];
```

```
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException
```

-What happens if this code snippet is run?

Recall

```
int[] array = new int[3];
int result = array[27];
```

```
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException
```

```
int result = Integer.parseInt(“hello”);
```

-What happens if this code snippet is run?

Recall

```
int[] array = new int[3];
int result = array[27];
```

```
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException
```

```
int result = Integer.parseInt(“hello”);
```

```
Exception in thread "main"
java.lang.NumberFormatException
```

–What happens if this code snippet is run?

Exceptions

- Intended to signal events which happen infrequently but cannot be ignored
    - “Exceptional”
    - Errors are common examples
- Can define different kinds of exceptions for different conditions

Exceptions

- Intended to signal events which happen infrequently but cannot be ignored
  - “Exceptional”
  - Errors are common examples
- Can define different kinds of exceptions for different conditions

```
java.lang.ArrayIndexOutOfBoundsException
java.lang.NumberFormatException
```

-For example, we can define exceptions for an array index being out of bounds (one kind of error condition), and exceptions indicating that a number was of an unexpected format / we couldn’t parse it (another kind of error condition)

Defining Exceptions

Inherit from the `Exception` class.

Has a constructor that takes a `String`.

–The passed String indicates a message which can encode more details (e.g., “57 is not negative”)

Defining Exceptions

Inherit from the `Exception` class.

Has a constructor that takes a `String`.

```
public class MyException
  extends Exception {
  public MyException(String message) {
    super(message);
  }
}
```

–The passed String indicates a message which can encode more details (e.g., “57 is not negative”)

Example:
`MyException.java`

Throwing Exceptions

Methods must state which exceptions they throw, using the `throws` reserved word

Throwing Exceptions

Methods must state which exceptions they throw, using the `throws` reserved word

```
public static void myMethod()
   throws MyException {
   ...
}
```

–Declaring that myMethod throws MyException

Throwing Exceptions

Methods must state which exceptions they throw, using the `throws` reserved word

```
public static void myMethod()
   throws MyException {
   ...
}
```

```
public static void myMethod()
   throws MyException, OtherException {
   ...
}
```

-Declaring that myMethod throws MyException or OtherException

Throwing Exceptions

Exceptions can be thrown with the `throw` reserved word

Throwing Exceptions

Exceptions can be thrown with the `throw` reserved word

```
public static void myMethod()
   throws MyException {
   if (...) {
      throw new MyException("message");
   }
}
```

Example

- `MyException.java`
- `ThrowMyException.java`

-Key point in the example: thrown exceptions can traverse method boundaries. Main can also throw MyException even though it doesn't explicitly use throw, since it calls something that says it throws MyException

Catching Exceptions

Exceptions can be caught with `try`...`catch`, stopping them from moving up

Catching Exceptions

Exceptions can be caught with `try`...`catch`, stopping them from moving up

```
try {
  myMethod();
} catch (MyException e) {
  System.out.println(e.toString());
}
System.out.println(“GETS HERE”);
```

Example: `CatchException.java`