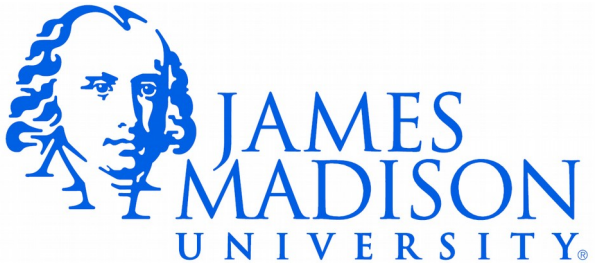
CS139 Exceptions



Style Question

A:

```
public BabyNames(ArrayList<String> nameLines) {
   String[] fields;

   girlNames = new HashMap<>();
   boyNames = new HashMap<>();

   for (String line : nameLines) {

      fields = line.split(",");

         if (fields[1].equals("F")) {
         girlNames.put(fields[0], Integer.parseInt(fields[2]));
      } else {
         boyNames.put(fields[0], Integer.parseInt(fields[2]));
      }
   }

}
```

B:

```
public BabyNames(ArrayList<String> nameLines)
{
   String[] fields;
   String name;
   String sex;
   int count;

   girlNames = new HashMap<>();
   boyNames = new HashMap<>();

   for (String line : nameLines) {

      fields = line.split(",");
      name = fields[0];
      sex = fields[1];
      count = Integer.parseInt(fields[2]);

      if (sex.equals("F")) {
         girlNames.put(name, count);
      } else {
         boyNames.put(name, count);
      }
   }

}
```

Which do you prefer? A or B?

Conflicting Considerations

- Concise code is good
  - In general: fewer lines of code means fewer places to make a mistake.
  - Lots of single-use intermediate variables can cause clutter

Conflicting Considerations

- Concise code is good
  - In general: fewer lines of code means fewer places to make a mistake.
  - Lots of single-use intermediate variables can cause clutter
- Explicit code is good
  - Code should be self documenting
  - Each line should solve one clear problem
  - Long, complex lines of code are hard to read and debug:

```
materialList.add(hatMap.get(bob.getName()).getMaterial());
```

  - In this case, I prefer B.

Null Pointer Exceptions

- Here is the error message:

```
Exception in thread "main" java.lang.NullPointerException
    at BabyNames.<init>(BabyNames.java:51)
    at NameApp.<init>(NameApp.java:22)
    at NameApp.main(NameApp.java:65)
```

- Here is line 51:

```
girlNames.put(name, count);
```

- What's wrong?
- How can we fix it?

Null Pointer Exceptions

- Here is the error message:

```
Exception in thread "main" java.lang.NullPointerException
    at BabyNames.<init>(BabyNames.java:51)
    at NameApp.<init>(NameApp.java:22)
    at NameApp.main(NameApp.java:65)
```

- Here is line 51:

```
girlNames.put(name, count);
```

- What's wrong?
  - `girlNames` is empty, (uninitialized, contains null)
- How can we fix it?
  - Make sure that we properly initialize `girlNames`

Exceptions

- Any problems with this code?

```
/**
 * Return the maximum value from the array numbers,
 * or -1 if the array has length 0.
 */
public static int max(int[] numbers) {

   if (numbers.length == 0) {
      return -1;
   }

   int curMax = numbers[0];

   for (int i = 1; i < numbers.length; i++) {
      if (numbers[i] > curMax) {
         curMax = numbers[i];
      }
   }

   return curMax;
}
```

Exceptions

- Any problems with this code?
  - What if the input is {-23, -7, -1, -4}?

```
/**
 * Return the maximum value from the array numbers,
 * or -1 if the array has length 0.
 */
public static int max(int[] numbers) {

   if (numbers.length == 0) {
      return -1;
   }

   int curMax = numbers[0];

   for (int i = 1; i < numbers.length; i++) {
      if (numbers[i] > curMax) {
         curMax = numbers[i];
      }
   }

   return curMax;
}
```

Solution: Exceptions

- Code may throw an exception to indicate that a problem has occurred.
- throw is a bit like return, but only used when something has gone wrong.
- throws keyword is used to indicate which exception types a method *could* throw. A bit like specifying a return type.

Exceptions

```java
/**
 * Return the maximum value from the array numbers, throw an
 * IllegalArgumentException if the array has length 0.
 */
public static int max(int[] numbers) throws IllegalArgumentException {

   if (numbers.length == 0) {
      throw new IllegalArgumentException("Array has length 0.");
   }

   int curMax = numbers[0];

   for (int i = 1; i < numbers.length; i++) {
      if (numbers[i] > curMax) {
         curMax = numbers[i];
      }
   }

   return curMax;
}
```

Exceptions

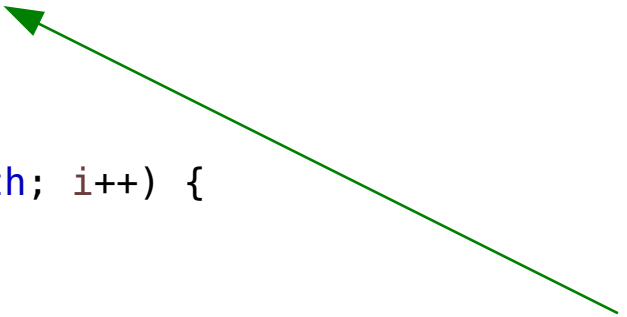
```java
/**
 * Return the maximum value from the array numbers, throw an
 * IllegalArgumentException if the array has length 0.
 */
public static int max(int[] numbers) throws IllegalArgumentException {

   if (numbers.length == 0) {
      throw new IllegalArgumentException("Array has length 0.");
   }

   int curMax = numbers[0];

   for (int i = 1; i < numbers.length; i++) {
      if (numbers[i] > curMax) {
         curMax = numbers[i];
      }
   }

   return curMax;
}
```



Exceptions are objects!

Exceptions

Only required for checked exceptions. (More in a minute)

```
/**
 * Return the maximum value from the array numbers, throw an
 * IllegalArgumentException if the array has length 0.
 */
public static int max(int[] numbers) throws IllegalArgumentException {

   if (numbers.length == 0) {
      throw new IllegalArgumentException("Array has length 0.");
   }

   int curMax = numbers[0];

   for (int i = 1; i < numbers.length; i++) {
      if (numbers[i] > curMax) {
         curMax = numbers[i];
      }
   }

   return curMax;
}
```

Try/Catch Example

```java
/**
 * Read some ints from the terminal and show the user the maximum.
 */
public static void showUserMax() {
   int[] inputArray;
   int maxValue;
   String message = null;

   while (message == null) {
      try {
         System.out.println("Please enter some numbers.");
         inputArray = getArrayFromUser();
         maxValue = max(inputArray);
         message = String.format("Your max is: %d", maxValue);

      } catch (IllegalArgumentException e) {
         System.out.println("Error! You entered 0 numbers!");
      }
   }
   System.out.println(message);
}
```

Try/Catch Example

```
/**
 * Read some ints from the terminal and show the user the maximum.
 */
public static void showUserMax() {
   int[] inputArray;
   int maxValue;
   String message = null;

   while (message == null) {
      try {
         System.out.println("Please enter some numbers.");
         inputArray = getArrayFromUser();
         maxValue = max(inputArray);
         message = String.format("Your max is: %d", maxValue);

      } catch (IllegalArgumentException e) {
         System.out.println("Error! You entered 0 numbers!");
      }
   }
   System.out.println(message);
}
```

If an exception happens here, the remaining code in the try block will be skipped.

Execution will jump to the catch block with the matching catch clause

Unchecked Exceptions

- You've seen several exception types this semester:
  - ArrayIndexOutOfBoundException
  - NullPointerException
  - Etc.
- These have all been unchecked exceptions
  - Unchecked exceptions are usually the result of programmer error.
  - The solution is NOT to create a try catch block
  - The solution is to fix the error in the code.

Checked Exceptions

- Other exception classes are checked exceptions
  - FileNotFoundException
  - EOFException
- Typically, these may result from problems outside of the programmers control
- Java requires us to write code to deal with checked exceptions or it won't compile
- (This is why I've been providing all of the file I/O code.)

Nice Things About Exceptions

- https://docs.oracle.com/javase/tutorial/essential/exceptions/advantages.html
- Separate error handling code from "normal" code.
  - try block contains the normal code.
- Flexibility in communicating what went wrong:
  - There are many exception classes, and we can create our own
- Flexibility in deciding where to handle the error
  - Exceptions just keep working up the call stack until they get caught or they get past main.

Quiz

- What will be printed by the following code?

```
fileName = "NONEXISTENTFILE.txt";
System.out.print("A ");

try {

   System.out.print("B ");
   file = new File(fileName);
   scanner = new Scanner(file); // Will throw FileNotFoundException!
   System.out.print("C ");

} catch (FileNotFoundException e) {
   System.out.print("D ");
}
System.out.print("E ");
```

Quiz

- What will be printed by the following code?

```
fileName = "NONEXISTENTFILE.txt";
System.out.print("A ");

try {

   System.out.print("B ");
   file = new File(fileName);
   scanner = new Scanner(file); // Will throw FileNotFoundException!
   System.out.print("C ");

} catch (FileNotFoundException e) {
   System.out.print("D ");
}
System.out.print("E ");
```

A B D E