Recall: Exception Handling

• When are exceptions thrown?
  Example:
  ```java
  if (internet is down) {
    throw new IOException("No network connection");
  }
  ```

• What happens when the exception is thrown?
• What does a “handler” look like?
  Example:
  ```java
  try {
    weather = downloadWeatherInfo();
  } catch (IOException e) {
    weather = lookOutWindowAndSeeIfItsRaining();
  }
  ```
Practical Examples

Have we seen examples where exception throwing would have made sense?

• Student class (spending a token when you have none)
  
  if(tokenLevel == 0) {
    throw new IllegalStateException("No tokens present");
  }

• FlagMaker (error flag is stupid)
  
  if(countryCode < 1 || countryCode > 12) {
    throw new IllegalArgumentException(countryCode + " is not a valid country.");
  }
Catching Multiple Types of Exceptions

You can catch more than one kind of exception:

```java
try {
    <put troublesome code here>
} catch (NullPointerException e) {
    <handler here>
} catch (ArithmeticException e) {
    <another handler here>
} catch (IOException e) {
    <another one here>
}
```
Finally block

Optional. Put code in finally block that is “mission critical”.

try {
    <put troublesome code here>
} catch (NullPointerException e) {
    <handler here>
} catch (ArithmeticException e) {
    <another handler here>
} catch (IOException e) {
    <another one here>
} finally {
    <put something here that should ALWAYS run>
}
Finally block ALWAYS runs

Once try block has begun, the finally block will run...
• When no exceptions are thrown
• When an exception is thrown and caught locally
• When an exception is thrown but NOT caught locally
• When method is terminated with return
Collections

Real world programs process huge quantities of data
How can we store a billion user names?

• Make a billion individual variables?

We need a way to use a single variable to store a
(theoretically) unbounded number of items.
1. Java Collections Framework (later)
2. Arrays (today)
Arrays of primitives

An array is a sequence of values stored contiguously.

Let’s explain and draw memory diagram:

```java
int[] a;
a = new int[4];
```

How do we access each value individually? Elements are indexed (0-based).

Examples of expressions using indexing.
Details

• Values in the array must all be the same type
• Arrays are objects, so they go on the heap.
• Arrays are always initialized with default values
• Indexing is 0-based
Processing Arrays

Arrays have a length field:

```java
arr.length
```

Standard idiom for processing array:

```java
for (int i = 0; i < arr.length; i++) {
    process a[i]
}
```
Examples

ArrayExamples.java
twoDArr.java