Reminders:

- Register on Piazza
- HW1 and P1 assigned.

Last class:

- Iterators
- Java Visibility Modifiers
- Primitives vs. References

Today:

- `ListIterator` to solve `hasDuplicates()`
- Primitives vs. References (finish)
- Command-Line Java
- Exceptions
- Week in Review
Using ListIterators: Example

Complete the method using listIterators to determine if the list contains duplicates.

    public boolean hasDups(ArrayList<String> list)
Primitives vs. References: Parameter Passing

Primitives

Suppose:

```c
void mod1(int x) {
    x = 7;
}
```

and in main():

```c
int x = 1;
int[] y = {1, 2, 3};
mod1(x); mod1(y[2]);
```

References:

Suppose:

```c
void mod2(int[] x) {
    x[0] = 7;
}
void mod3(int[] x) {
    x = new int[x.length]
    x[0] = 14;
}
```

and in main():

```c
int[] a = {1, 2, 3};
mod2(a);
mod3(a);
```
Command-Line Java Development

Editing

Compiling
-\texttt{d} option

Running
-\texttt{cp} option
Command-Line Arguments

Write a Java program that takes several command line arguments, puts them in a Sack, and counts the number of times the first argument appears in the Sack.
Exceptions

What are they?

Java Syntax

    throw exceptionObject

Example

    throw new NoSuchElementException();
Behind the Scenes
Handling Exceptions with try-catch blocks

Java syntax

```java
try {
    // try block

    ... // code that might cause an exception to be thrown

} catch (ExceptionType1 identifier1) {
    // catch block

    ... // code to handle exception type 1

} catch (ExceptionType2 identifier2) {
    // catch block

    ... // code to handle exception type 2

} ...
finally {
    // finally block

    ... // code executed no matter what happens in try
}
```

Example
Handling Exceptions with throws

Java syntax

... methodName (parameter list)
    throws ExceptionType1, ExceptionType2 , ... { 
        ...
    }

Example

public static void main(String[] args) throws IOException {
    ...
}
Defining Exceptions - Examples

Checked

    extends Exception

Unchecked

    extends RuntimeException
public class ExceptionTester {
    public static void main(String[] args) {
        System.out.print("main[");
        try {
            methodA(); System.out.print("after A,");
            methodE(); System.out.print("after E,");
        } catch (RedException exc) {
            System.out.print("red,");
        } catch (GreenException exc) {
            System.out.print("green,");
        }
        System.out.println("]main");
    }

    private static void methodA() {
        System.out.print("\nA[");
        try {
            methodB();
            System.out.print("after B,");
        } catch (BlueException exc) {
            System.out.print("blue,");
        }
        System.out.println("]A");
    }

    private static void methodB() {
        System.out.print("\nB[");
        try {
            methodC();
            System.out.print("after C,");
        } catch (YellowException exc) {
            System.out.print("yellow,");
            throw new GreenException();
        } catch (RedException exc) {
            System.out.print("red,");
        }
        methodD();
        System.out.print("after D,");
        System.out.println("]B");
    }
}
Exceptions – Example (cont'd)

What gets printed when:

• methodC throws a red exception?

    main[
    A[
    B[

• methodD throws a red exception? *

• methodE throws a red exception? *

• methodC throws a blue exception? *

• methodC throws a green exception? *

• methodD throws a green exception?

• methodC throws a yellow exception? *

• methodD throws a yellow exception? *

• methodE throws a yellow exception? *

• methodD throws an orange exception? *

* means it is left as an exercise for you. Testing code provided on website.
Week in Review

- Abstraction
- Interfaces, ADTs
- Objects, casting, autoboxing
- Generics
- Qualities of good software
- Sack ADT
- List ADT
- Iterators, List Iterators
- Primitives vs. References
- Misc Java stuff (Visibility, Command-Line)
- Exceptions