Homework h2 available, due 10 pm Friday, Sept 23th
Program p1 assigned. GET STARTED NOW (incremental development is key)

Assignment questions? Post it on Piazza or Consult with a TA during scheduled hours.

Report any exam conflicts or McBurney exam accommodations before Oct 7th.
Updated Module 3 with links for Exams and instructions for reporting Exam Conflicts

Email your instructor by this Friday, 9/23, if you participate in religious observances
that might interfere with course requirements. Include your name, UW ID#, date and explanation.

Last Time
Iterators
• iterators and the Java API
• using iterators
• options for implementing iterators
• making a class iterable

Today
Hand in all required files for each Program submission. We only see final submission.
Exceptions Review
• throwing
• handling
• execution
• practice with exception handling
• throws and checked vs. unchecked
• defining

Next Time
Read: finish Exceptions, start Linked Lists
Java Primitives vs. References Review
Chains of Linked Nodes
• Listnode class
• practice with chains of nodes
Exception Throwing – Signaling a Problem

Java Syntax

    throw exceptionObject;

Example
Exception Handling – Resolving a Problem

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
}
... more catch blocks

finally {
    // finally block – optional
    code always executed when try block is entered
}
```

Example
Exception Execution

Normal Execution
- Start: top of main()
  - Execute:
  - Skip:
  - Switch to Exception Handling Execution

Exception Handling Execution
- Skip:
  - Execute:
  - Switch back to Normal Execution

Searching for a Matching Catch
1. Locally
  2. Remotely

Checking a Match
1. Match Found
  2. No Match Found
ExceptionTester Example

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("main-red,");
        } catch (GreenException exc) {
            System.out.print("main-green,");
        } finally {
            System.out.print("main-finally,");
        }
        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("A-blue,");
        }
        System.out.println("]A");
    }

    private static void methodB() {
        System.out.print("\nB[");
        methodC();
        System.out.print("after C,");
        try {
            methodD();
            System.out.print("after D,");
        } catch (YellowException exc) {
            System.out.print("B-yellow,");
            throw new GreenException();
        } catch (RedException exc) {
            System.out.print("B-red,");
        } finally {
            System.out.print("B-finally,");
        }
        System.out.println("]B");
    }
}
What is Output When:

1. no exception is thrown

\[
\text{main[A[B[}
\]

2. \text{methodE throws a YellowException?}

\[
\text{main[A[B[}
\]

3. \text{methodC throws a GreenException?}

\[
\text{main[A[B[}
\]

4. \text{methodD throws a GreenException?}

\[
\text{main[A[B[}
\]
What is Output When:

5. `methodC throws a RedException`?

```
main[
 A[
 B[
```

6. `methodD throws a RedException`?

```
main[
 A[
 B[
```

7. `methodD throws a YellowException`?

```
main[
 A[
 B[
```

8. `methodD throws a OrangeException`?

```
main[
 A[
 B[
```
What is Output When:

9. methodC throws a YellowException?

main[
A[
B[

10. methodC throws a BlueException?

main[
A[
B[

11. methodE throws a RedException?

main[
A[
B[
throws clause – Passing the Buck

Checked vs. Unchecked

Java Syntax

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

Example

```java
public static void main(String[] args) throws IOException {
    ...
```
Defining a New Exception Class

Checked

    public class MyException extends ________________ {

    }

Unchecked

    public class MyException extends ________________ {

    }

Example

    public class EmptyBagException extends Exception {

        public EmptyBagException() {
            super();
        }

        public EmptyBagException(String msg) {
            super(msg);
        }

    }
What is Output When SOLUTION:

1. no exception is thrown

main[
 A[
 B[after C,after D,B-finally,]B
 after B,]A
 after A,after E,main-finally,]main

2. methodE throws a YellowException?

main[
 A[
 B[after C,after D,B-finally,]B
 after B,]A
 after A,main-finally,]main
Exception in thread "main" YellowException
 at ExceptionTester.methodE(ExceptionTester.java:145)
 at ExceptionTester.main(ExceptionTester.java:37)

3. methodC throws a GreenException?

main[
 A[
 B[main-green,main-finally,]main

4. methodD throws a GreenException?

main[
 A[
 B[after C,B-finally,main-green,main-finally,]main
What is Output When SOLUTION:

5. methodC throws a RedException?

```java
main[
A[
B[main-red,main-finally,]main
```

6. methodD throws a RedException?

```java
main[
A[
B[after C,B-red,B-finally,]B
after B,]A
after A,after E,main-finally,]main
```

7. methodD throws a YellowException?

```java
main[
A[
B[after C,B-yellow,B-finally,main-green,main-finally,]main
```

8. methodD throws a OrangeException?

```java
main[
A[
B[after C,B-finally,main-finally,Exception in thread "main"
OrangeException
at ExceptionTester.methodD(ExceptionTester.java:129)
at ExceptionTester.methodB(ExceptionTester.java:80)
at ExceptionTester.methodA(ExceptionTester.java:60)
at ExceptionTester.main(ExceptionTester.java:34)
```
What is Output When SOLUTION:

9. methodC throws a YellowException?

```java
main[
A[
B[main-finally,Exception in thread "main" YellowException
  at ExceptionTester.methodC(ExceptionTester.java:109)
  at ExceptionTester.methodB(ExceptionTester.java:76)
  at ExceptionTester.methodA(ExceptionTester.java:60)
  at ExceptionTester.main(ExceptionTester.java:34)
```

10. methodC throws a BlueException?

```java
main[
A[
B[A-blue,]A
after A,after E,main-finally,]main
```

11. methodE throws a RedException?

```java
main[
A[
B[after C,after D,B-finally,]B
after B,]A
after A,main-red,main-finally,]main