CSE 8B Today

Don’t forget: Fill out your CAPES, TA evaluations, and your survey

Just 2 more days of class!

Today: More Recursion and Exceptions
• Generics – quick example
• Thread example
• Exceptions
• Review midterm 4 – recursion examples

Thursday: Wrap up and review

How’s PSA8 going?
A. I haven’t started at all
B. I’ve read it and started thinking about it
C. I’ve started some coding
D. I’m done
FINAL EXAM – TUESDAY and THURSDAY in MANDEVILLE AUDITORIUM

- Seating assignment on your email on MONDAY – come early to find it (specially if you don’t get your email)
- The final exam will look a lot like the in-term exams, only longer. Designed to take about 2 hours (you will have 3)

The best things you can do to review or get help before the final:

- Study by RE-DOING clicker questions, in-class coding and PSAs (in paper??)!!

- Check website supplementary material, videos, ...

- Office hours!
  - LAST DAY I’ll be doing office hours: Monday morning.
  - Tutors and TA’s will also be available until Monday (double check calendar) After Monday, only Piazza.

- REVIEW SESSION: at discussion this week AND Thursday lecture.
Generics
Binary Search example

```
public static int binarySearch( ArrayList<Integer> theList, 
                            int toFind, int low, int high)
```

The method will return the position of the item if it is found

low will start at 0, high will start at theList.length-1

Assuming the array pictured above is named myA...

> binarySearch( myA, 4, 0, 8 )

> binarySearch( myA, 5, 0, 8 )
Threads ...
PanelAnimator is an inner class of ConnectFourAnimationDemo class

PanelAnimator implements Runnable {
  int panelNum;
  int delay;
  public PanelAnimator( int num, delayIn ) {
    panelNum = num;
    delay = delayIn;
  }
  public void run() {
    try {
      while( true ) {
        if ( running ) {
          if ( board[panelNum] == ' ' )
            board[panelNum] = turn;
          else
            board[panelNum] = ' '; // Repaint
          repaint();
        }
        Thread.sleep( delay );
      }
    }
    catch (InterruptedException ex) { } // End of PanelAnimator
  }
}

// In the ConnectFourAnimationDemoConstructor...
PanelAnimator p0 = new PanelAnimator( 0, 500 );
new Thread(p0).start();
} // End of C4AD constructor

// PanelAnimator is an inner class of ConnectFourAnimationDemo class

WHAT IF I WANT task object a separate class????

What will the GUI do when the user clicks “Start”? 
A. All three circles will change colors automatically, like they did before.
B. Only the first circle will change colors automatically
C. Nothing will happen
class PanelAnimator implements Runnable {
    int panelNum;
    int delay;
    public PanelAnimator( int num, delayIn ) {
        panelNum = num;
        delay = delayIn;
    }
    public void run() {
        try {
            while( true ) {
                if ( running ) {
                    if ( board[panelNum] == ' ' )
                        board[panelNum] = turn;
                    else
                        board[panelNum] = ' ';
                    repaint();
                }
                Thread.sleep( delay );
            }
            catch (InterruptedException ex) { }
        }
    }
}

// In the ConnectFourAnimationDemoConstructor...
PanelAnimator p0 = new PanelAnimator( 0, 500 );
new Thread(p0).start();  

DO IT AT HOME: Add code to make the three circles change color at different speeds
What is an IOException?

Essentially, an exception is just an error that can be passed around (thrown). If sourceFile does not exist, the Scanner’s constructor will throw an exception.
Handling exceptions

// In class StringPlay
public String readIt(String filename)
{
    File sourceFile = new File(filename);
    Scanner input = null;
    try {
        input = new Scanner(sourceFile);
    } catch (IOException e) {
        System.out.println(e.getMessage());
        return "";
    }
    String allText = "";
    while (input.hasNextLine()) {
        String s1 = input.nextLine();
        allText = allText.concat(s1);
    }
    System.out.println(allText);
    return allText;
}
public String readIt( String filename )
{
  File sourceFile = new File( filename );
  Scanner input = null;
  try {
    input = new Scanner( sourceFile );
  }
  catch ( IOException e ) {
    System.out.println( e.getMessage() );
    // return ""; NO MORE RETURN HERE
  }
  String allText = "";
  while ( input.hasNextLine() )
  {
    String s1 = input.nextLine();
    allText = allText.concat( s1 );
  }
  System.out.println( allText );
  return allText;
}
Practice PROBLEM: Complete the code to prompt the user to enter a new filename until the file exists.

```java
public String readIt( String filename ) {
    File sourceFile = new File( filename );
    Scanner input = null;
    Scanner userInput = new Scanner( System.in );
    while ( input == null ) {
        try {
            input = new Scanner( sourceFile );
        } catch ( IOException e ) {
            System.out.println( e.getMessage() );
        }
    }
    String allText = "";
    while ( input.hasNextLine() )
    {
        String s1 = input.nextLine();
        allText = allText.concat( s1 );
    }
    System.out.println( allText );
    return allText;
}
```
So EXCEPTIONS are ...

Object    Throwable
  |
Exception
  |
  |
  |
  |
ClassNotFoundException
  |
IOException
  |
RuntimeException
  |
  |
  |
  |
  |
  |
ArithmeticException
  |
NullPointerException
  |
IndexOutOfBoundsException
  |
IllegalArgumentException
  |
  |
  |
  |
  |
  |
  |
  |
  |
  |
VirtualMachineError
  |
  |
  |
  |
  |
  |
  |
  |
  |
  |
  |
Many more classes

Many more classes
Java Exceptions: Main Goal

• When you encounter a situation that won’t let your code act “normally”
  – Don’t “print an error” (printing to the screen isn’t the only way we interact with computers)
  – Throw an exception back to the location that called this method
  – It will then catch it and decide what to do

• Why throw it?
  – Scanner (written by Sun) doesn’t get to dictate the way YOU handle an exceptional circumstance
public class TurtleTester {
    public static void mosaic(Turtle t, int side) {
        if (side > 0 && side <= 300) {
            t.square(side);
            t.forward(20);
            mosaic(t, side*2);
            System.out.println("Square side: " + side);
        }
        return;
    } // end of mosaic method

    public static void main(String[] args) {
        World w = new World(400, 400);
        //Create a turtle in w at (x,y)
        Turtle jose = new Turtle(200, 400, w);
        TurtleTester.mosaic(jose, 40);
    } // end of main method

} // end of CLASS TurtleTester

How many stack frames do we create when we run this main?
A. 1
B. 3
C. 5
D. 7