

Introduction to Programming

Chapter 2

Practice Program

- Write a program that prints “Hello, Name” to the user.

```
import javax.swing.JOptionPane; //necessary for the dialog box
public class HelloName //class declaration
{
    public static void main(String[] args) //main method declaration
    {
        //this line of code asks the user to type their name in a dialog box.
        String name= JOptionPane.showInputDialog("What is your name?");

        //print message to screen
        System.out.println("Hello " + name);
    } //end main
} //end class
```

Practice Program : with correct commenting

```
////////////////////////////////////  
//Main Class File: HelloName.java  
//File:      HelloName.java  
//Semester:   Fall 2012  
//  
//Author:     Alicia Maxwell  
////////////////////////////////////80 columns wide //////////////////////////////////  
import javax.swing.JOptionPane;  
  
/**  
 * Aquire the users's name with a dialog box, then print "Hello, Name!"  
 * @author Alicia Maxwell  
 */  
public class HelloName  
{  
    //continued on next slide
```

Practice Program : with correct commenting

```
/**
 * Get user's name and print a welcome message.
 * @param args - Command line arguments
 * @return void
 */
public static void main(String[] args)
{
    //ask the user for their name
    String name= JOptionPane.showInputDialog("What is your name?");

    //print out the message
    System.out.println("Hello " + name + "!");

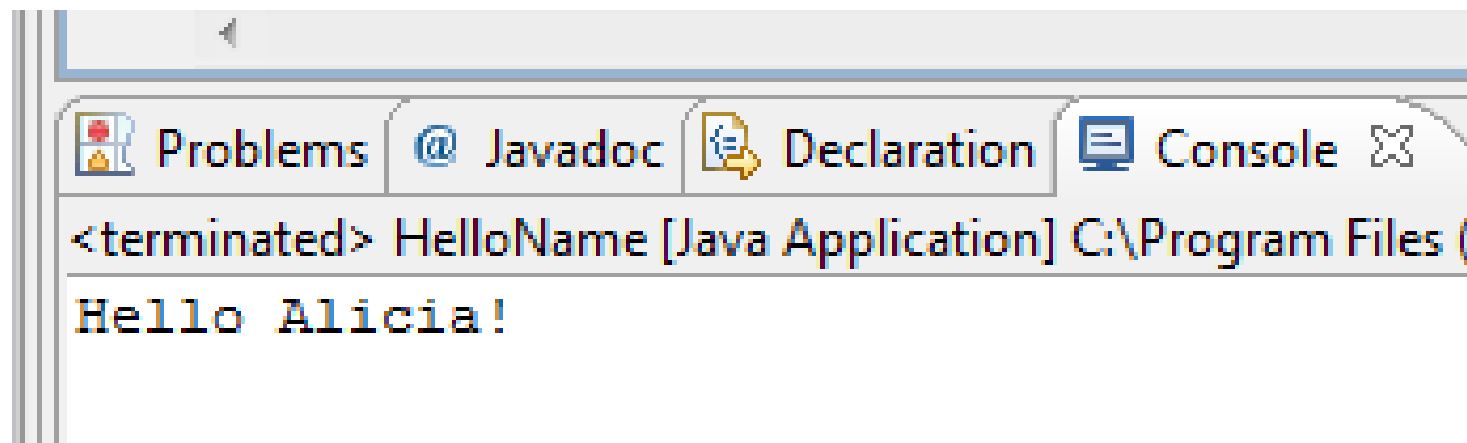
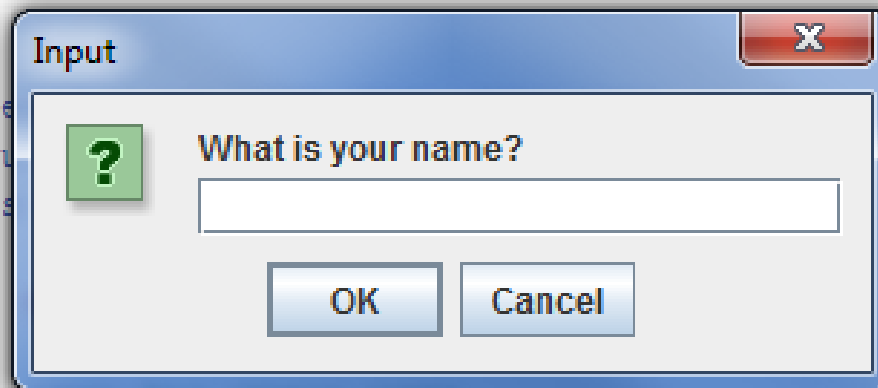
    //end the program when the last message box closes
    System.exit(0);

} //end main
} //end class
```

Practice Program: Run Program

```
import javax.swing.JOptionPane;

public class HelloName
{
    /**
     * Get the user's name
     * and say you love them
     * @param args
     */
    public static void main(String[] args)
    {
        //ask the user for their name
        String name= JOptionPane.showInputDialog("What is your name?");
    }
}
```



Fundamental Data Types

- Primitive Data Types

- int
- long
- short
- byte
- double
- Float
- boolean
- char

- String class

Primitive Data Types: integer numbers

- int
 - Integer values (12, -5, 0, ...)
 - Range from -2,147,483,648 to 2,147,483,647
 - Stored using a 2's complement (leading digit is 0 if positive, 1 if negative) with 32 bits
- long
 - Integer value
 - To store seven as a long, type: 7L
 - Range -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 with 64 bits.
- short
 - Integer value
 - Range -32,768 to 32,767 with 16 bits
- byte
 - Integer value
 - Range -128 to 127 with 8 bits

(4 bits)

Binary	decimal
0000	0
0001	1
0010	2
0011	3

(32 bits)

0000 0000 0000 0000 0000 0000 0000 0000 = 0

0000 0000 0000 0000 0000 0000 0000 0001 = 1

1111 1111 1111 1111 1111 1111 1111 1111 = -1

Primitive Data Types: decimal numbers

- double

- A decimal number
- Stored using 64 bit IEEE double precision floating point (positive values, negative values, positive and negative 0, positive and negative infinity, and NaN-not a number).
- Type in regular notation: 1.8
- Type in scientific notation: 1E4
- Type in scientific notation: 5.3E-3

- float

- A decimal number
- 32 bit IEEE single precision floating point number
- To store 1.8 type: 1.8f

Primitive Data Types: non-numeric values

- char

- Characters including letters, numbers, and symbols
- To store c type: 'c'
- Stored in 16 bit Unicode (Appendix A)
- Range '\u0000' (0) to '\uffff'(65535)

- boolean

- Boolean notation (true or false values)
- Type: true
- Type: false

String class

- Not a primitive data type
- Used differently from other objects
- “String literal” –something written in quotes, which has a constant value
- Type: “Hello”

Reading Keyboard Input

- Use the Scanner class in the java.util package

```
import java.util.Scanner;
```

- Construct a Scanner object

```
Scanner in = new Scanner(System.in) ; //read from the keyboard
```

- Print out a prompt to tell the user what we want them to type

```
System.out.println("Type your name.");
```

- Use Scanner methods to read in what was typed

- nextInt() method to read an int

- Note: the return character is left sitting on the input line after nextInt() reads the int that was typed.

- nextDouble() for a double

- nextLine() for an entire line of text

- next() for a single String

- Use API documetation (see p 43) to discover more methods

```
String name = in.nextLine();
```

- Use the input

```
System.out.printf("Hello, %s", name);
```

Variables

- Definition: a storage location in a computer program with a name and a value.
- Holds only one value at a time.
- Value each variable holds may be changed.

- Variable declaration
 - Definition: creating the variable
 - Example: `int sum;`
- Variable initialization
 - Definition: setting an initial value for the variable
 - Example: `int sum = 0;`
 - Often combined with declaration of the variable.
 - Set to a default value (such as 0) and then change it later.
 - Must initialize to the declared type
 - `int sum = "hello"; //FAILS`

- You must declare and initialize a variable before you can use it.

```
int sum =0;
int a=5;
sum = a+b; //FAILS: b not declared yet
int b = 7;
```

Variables

- Variable assignment

- Definition: store a new value in a variable, replacing the previously stored value.
- Assignment is done with the “=” operator
 - Note that “==” defines mathematical equality while “=” is used for assignment.
- The left hand variable is given the right hand value.

```
int x = 12; //x is set to 12
int a = 11; //a is set to 11
x = a; //it is x whose value is changed
```

- assignment with the “++” operator
 - Increase the variable by one

```
int count = 0;
count ++; //same as count = count +1, count is now 1
```

- Likewise the “--” operator

- Other shorthand assignments

```
result += x; //result = result + x
```

```
result *=x; //result = result * x
```

Variables

- Variable Names

- Explains the purpose of the variable
- Easy to understand: do not use single letters
- Rules
 - Lower case first letter
 - Begin with a letter or the `_` underscore character
 - No other symbols are allowed
 - Cannot use Java reserved words
 - `class`, `double`, `true`... (see Appendix C)
 - If the name has two words, capitalize the first letter of the second word.
 - `studentName`

- Are these acceptable names?

- ~~String #1Job~~
- `double product` ✓
- ~~boolean correctValue?~~
- ~~String 5thGradeTeacher~~
- `int _myID` ✓
- ~~BaseballScore double~~

Variables

- Literals vs Variables

- Variables must be declared

```
String name = "Alicia";
```

```
int counter = 5;
```

```
int dozen = 10+2;
```

- Literals are actual Strings or numbers

- Constants

- A variable whose value can never change
- Written in all capital letters with _ between words
- Declared with the word "final"

```
final double MASS_OF_EARTH = 5.97E24; //kg
```

```
final double LITER_PER_OUNCE = 0.0296;
```

```
double numberOfEarthsInSun = MASS_OF_SUN / MASS_OF_EARTH;
```

- Avoid "magic numbers"

```
double canVolume = canOunces * LITER_PER_OUNCE;
```

```
double canVolume = canOunces * 0.0296;
```

- Clearer, easier to make changes

Arithmetic

Operation	Operator in Java	Example in Math	Example in Java
Addition	+	$f+7$	<code>f+7</code>
Subtraction	-	$p-c$	<code>p-c</code>
Multiplication	*	$bm, 5 \times 6$	<code>b*m</code>
Division	/	x/y	<code>x / y</code>
Remainder (Modulo)	%	$r \text{ mod } s$	<code>r % s</code>

Arithmetic: Expressions

Definition: a combination of variables, numbers (literals), operators, and/or method calls.

Precedence:

1. Multiplication, Division, Modulo
2. Addition and Subtraction

If multiple operations from one level, compute left to right

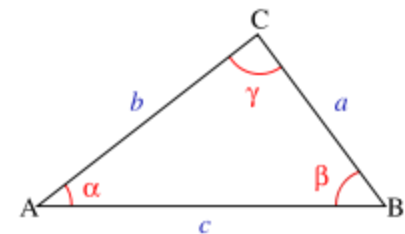
Write expressions in Java

- 1) Physics-motion under constant acceleration: $s = s_0 + v_0 t + \frac{1}{2} g t^2$

```
double s = s0 + v0*t + .5 * GRAVITY*Math.pow(t,2);
```

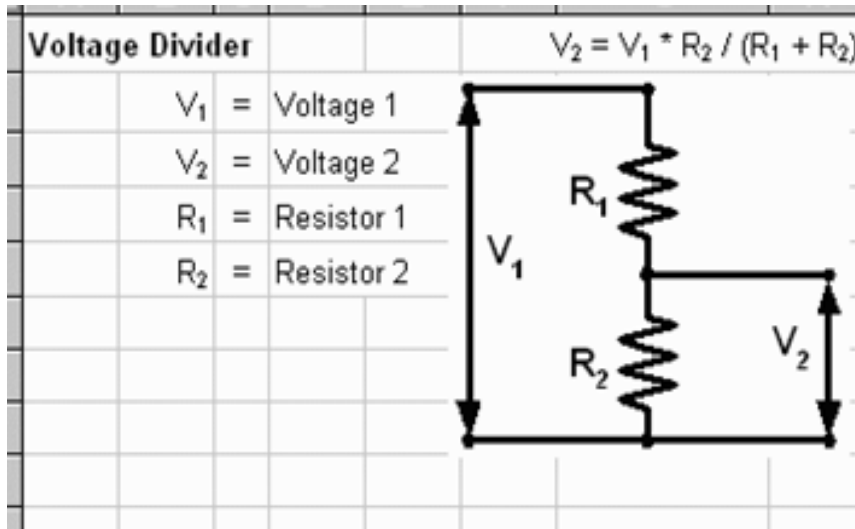
- 2) Math-Law of Cosines : $\sqrt{a^2 + b^2 - 2ab\cos(\gamma)}$

```
double c = Math.sqrt( Math.pow(a,2) + Math.pow(b,2)-  
2*a*b*Math.cos(gamma) );
```



Arithmetic: Expressions

3) From Electrical Engineering



```
double v2 = v1* (r2/ (r1+r2));
```

Arithmetic: Changing the type of a variable

- We cannot instantiate variables to a type different than they are declared

```
double angle = "something" ; //FAIL
```

- However, an int literal is automatically converted to a double when assigned to a double variable.

```
double angle = 15; //stores 15.0
```

- This also applies to math including both doubles and ints

```
7*2.0; // stores 14.0, not 14
```

- An int variable cannot be assigned a double literal

```
int angle 14.6; //FAIL
```

- Changing an int into a double can be performed with a “cast.” (The decimal portion of the number will be discarded and lost).

```
double balance = 25.67;
```

```
int dollars = (int) balance; //cast the balance to an int, result is 25
```

- Rounding is also possible

```
int dollars = (int) Math.round(balance); //now dollars is 26
```

Arithmetic: Integer Division

- Division including two integers
 - (If one number is a double, the result is stored as a double unless cast.)
- Remainders are discarded
- The result is NOT rounded
- Example: $8/2 = 4$
- Example: $7/2 = 3$ (the remainder of 1 is thrown away)

Arithmetic: Modulo and Remainder

- Java symbol is: %
- 5 mod 4 is written 5%4
- Essentially the remainder is saved.
 $7\%4 = 3$ (7 / 4 = 1 remainder 3)

- Example

```
int pennies = 1729;  
int dollars = pennies / 100; //result is 17  
int cents = pennies % 100; //result is 29
```

Arithmetic: Math Class

- Square root

```
Math.sqrt(n);
```

- Power

```
Math.pow(a,b); // a^b
```

- Constants

```
Math.PI;
```

Arithmetic: Comparisons

Algebraic operator	Java operator	Java example	What it means
=	=	x == y	x is equal to y
≠	!=	x != y	x is not equal to y
>	>	x > y	x is greater than y
<	<	x < y	x is less than y
≥	>=	x >= y	x is greater than or equal to y
≤	<=	x <= y	x is less than or equal to y

```
if (age >= 18)
    voteInElection();
if (age ! <18)
    voteInElection();
```

```
if (number % 2 == 0)
    isEven = true;
```

```
if (date != 25)
    todayIsChristmas = false;
```

```
while (hoursWorked < 8)
    dayOver = false;
```

```
for(int hour=0; hour<8; hour++)
    workAnotherHour();
```

Example Program

- Write a program that accepts the radius of a circle from the user and then prints the diameter (twice the radius), the circumference (2 times π times the radius), and the area (π times r squared) of the circle.
- Sample Solution 1:

```
import java.util.Scanner;
public class Circle
{
    public static void main(String[] args)
    {
        Scanner in = new Scanner(System.in);
        System.out.println("Please enter the radius of a circle: ");
        double radius = in.nextDouble(); //r

        double diam = radius*2;
        double circ = 2*Math.PI*radius;
        double area = Math.PI*Math.pow(radius,2);

        System.out.printf("Radius: %f, Diameter: %f, Circumference: %f, Area: %f",
            radius, diam, circ, area);
    }
}
```

Example Program

- Sample Solution 2:

```
import java.util.Scanner;
```

```
public class Circle
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        Scanner in = new Scanner(System.in);
```

```
        System.out.println("Please enter the radius of a circle: ");
```

```
        double radius = in.nextDouble(); //r
```

```
        System.out.printf("Radius: %f, Diameter: %f, Circumference: %f, Area: %f",  
                           radius, radius*2, 2*Math.PI*radius,Math.PI*radius*radius);
```

```
    }
```

```
}
```

Example

- Write a program that reads in the bill for dinner from a user, then computes appropriate tip.

```
import java.util.Scanner;

public class TipCalculator
{
    public static void main(String[] args)
    {
        Scanner in = new Scanner(System.in);

        System.out.println("Please enter bill: ");
        double bill = in.nextDouble();

        double tip = bill * 1.15; //assume 15% tip

        System.out.println("Correct tip: " + tip);
    }
}
```

Strings

- Definition: a sequence of characters (letters, numbers, punctuation, spaces, etc)
- Declaration:
`String myName = "Alicia";`
- Length of a String: the number of characters a String contains
`myName.length(); //returns 6`
- Empty String, length 0: `""`
- Concatenation
 - Combine two or more String's into a single string with the "+" sign
 - Example:
`int result = 12;`
`String message = "The result of this calculation is " + result;`

Strings: output to console

```
System.out.println("a message to print");
```

Escape Sequence	Description
<code>\n</code>	Newline. Position the screen cursor at the start of the next line
<code>\t</code>	Tab. Move the cursor to the next tab stop
<code>\r</code>	Carriage Return.
<code>\\</code>	Backslash. Print out the <code>\</code> character
<code>\"</code>	Double quote. Print out the <code>"</code> character

```
System.out.println("Then he said to me \"Hello.\"");
```

Output to console: Then he said to me "Hello."

```
System.out.println(System.out.println(" * \n *** \n*****"));
```

prints to console:

```
 *
 ***
*****
```

Strings: formatted output

```
System.out.printf( );
```

What you type	What kind of value you output
%s	String
%d	int
%f	double
%.2f	double with two decimal points like 3.45 or 12.34. Note that if you change 2 to a different number, you get that many digits after the decimal point.
%c	char

```
int a=5; int b= 7; int sum = a+b;  
System.out.printf(" a= %d, b=%d, a+b=%d" , a, b, sum);
```

Output: a=5, b=7, a+b=12

```
double balance = 14.566289533; String name= Joe;  
System.out.printf("Account Holder %s has %.3f dollars to spend.", name, balance);
```

Output: Account Holder Joe has 14.566 dollars to spend.

Strings: turning into numbers

- Calling convention:

```
Double.parseDouble(String);
```

```
Integer.parseInt(String);
```

- Example 1

```
String bal = "14.56";
```

```
double balance = Double.parseDouble(bal); //returns 14.56
```

- Example 2

```
String stringAge = "23";
```

```
int age = Integer.parseInt(stringAge);
```

- Example 3

```
String animal = "cat";
```

```
int cat = Integer.parseInt(animal); //FAIL "cat" is not an integer
```

Strings: made of characters

- Declared using single quotes
char letter = 'a';
- Encoded as numbers by the computer
 - See appendix B
 - System.out.println('H' + 1) = 73 because 'H' is 72
 - Note that 'h' and 'H' are different chars and encoded as different numbers
- Finding the ith character of a String using charAt()
 - Counting begins with character 0 in the String.

A	l	i	c	i	a
0	1	2	3	4	5

- "Alicia" has length of 6, characters in position 0 to 5

```
String myName = "Alicia";  
char myFirstLetter = myName.charAt(0); //result is 'A'  
char something = myName.charAt(6); //FAILS no char6
```

```
String classroom = "Room 5, CS building";  
char roomNu = classroom.charAt(5);
```

Strings: substring

- Calling convention:

```
stringVar.substring(int startOfSubstring, int afterEndOfSubstring);
```

```
stringVar.substring(int startOfSubstring); //collects all remaining characters
```

- Example 1

```
String thisClass = "CS 302 Introduction to Programming";
```

```
String courseNu = thisClass.substring(3,6); //returns "302"
```

```
//includes char3='3' to char5='2', but NOT char6=' '
```

```
int courseNumber = Integer.parseInt(courseNu); //returns 302
```

- Example 2

```
String myName = "Alicia Maxwell";
```

```
String myLastName = myName.substring(7); //returns "Maxwell"
```

- Example 3

```
String partner1 = "Rob";
```

```
String partner2 = "Sue"
```

```
String carveOnATree = partner1.charAt(0) + " & " + partner2.charAt(0);
```

```
//chars cannot be concatenated, so pull substrings of length 1
```

```
import java.util.Scanner; //Scanner lets us read input from the keyboard
```

```
public class Product{
```

```
    public static void main(String[] args) {  
        Scanner input = new Scanner(System.in);
```

```
        int x; // values given by user
```

```
        int y;
```

```
        int z;
```

```
        int result; //the product
```

```
        System.out.println( "Enter first integer: "); //prompt for input  
        x = input.nextInt(); //read first integer from keyboard
```

```
        System.out.println( "Enter second integer: "); //prompt for input  
        y = input.nextInt(); //read second integer from keyboard
```

```
        System.out.println( "Enter third integer: "); //prompt for input  
        z = input.nextInt(); //read third integer from keyboard
```

```
        result = x*y*z; //calculate the product of the three input numbers  
        System.out.printf("Product is %d\n", result); //print the result
```

```
    } //end main method
```

```
} //end class Product
```

- Employee who earns commission as well as an hourly wage. Write a method to calculate their monthly payment

Example

```
public class Employee
{
    private String name; //variables
    private double hourlyPay;
    private double percentCommission; //between 0 and 1, bonus per item sold

    //constructor
    public Employee(String firstName, double payRate, double commission) {
        name = firstName;
        hourlyPay = payRate;
        percentCommission = commission;
    }

    //payment, percentCommission, hourlyPay already set
    public double getWeeklyPayment(int hoursWorked, int itemsSold)
    {
        double payment = hoursWorked*hourlyPay;
        double addedCommission = itemsSold*percentCommission;
        payment += addedCommission;
        return payment;
    }
}
```

```
public class TestEmployee
{
    public static void main(String[] args)
    {
        Employee joe = new Employee("Joe", 8.50, .21);
        Employee ann = new Employee("Ann", 9.25, .16);
        Employee jamie = new Employee("Jamie", 7.98, .75);

        System.out.println("Payment to Joe: $" + joe.getWeeklyPayment(40, 15));
        System.out.println("Payment to Ann: $" + ann.getWeeklyPayment(35, 33));
        System.out.println("Payment to Jamie: $" + jamie.getWeeklyPayment(34, 74));

        //results:
        //Payment to Joe: $343.15
        //Payment to Ann: $329.03
        //Payment to Jamie: $326.82
    }
}
```

(See java code posted on website for complete program)

Random Class

- Pseudo-random numbers are generated
 - Numbers drawn from a sequence of numbers that does not repeat for a very long time, thus appearing random.
 - Random Number Generators which use the same seed will always generate the same random numbers, in the same order.
- Construction of Random Object
 - Zero argument constructor
 - `Random rng = new Random();`
 - Computer uses timestamp as the seed
 - Seeding
 - `Random rand = new Random (SEED VALUE);` //SEED VALUE is of type long
 - Seed is used to compute the pseudo-random number

Random

- Methods

- nextInt(n)

- Returns a random int between 0 (inclusive) and n (exclusive)

- int x=nextInt(n) means $0 \leq x < n$

- nextInt()

- Returns a random int in the range of allowed integers in Java (-2.1 billion to 2.1billion)

- nextDouble()

- Returns a random double between 0 (inclusive) and 1 (exclusive)

- Other methods exist

Random

- Generating numbers in a specific range

- Generate a number that is 8,9,10, or 11

- nextInt(4) gives $0 \leq x < 4$

- nextInt(4) + 8 gives $8 \leq x < 12$ which is $8 \leq x \leq 11$

- Solution:

- ```
Random rng = new Random();
```

- ```
int n = rng.nextInt(4) + 8;
```

- Generate a number that is in the range of 1-6

- nextInt(6) gives $0 \leq x < 6$

- nextInt(6) + 1 gives $1 \leq x \leq 6$

- Solution:

- ```
Random random = new Random();
```

- ```
int dice = random.nextInt(6) + 1;
```

- Generate a floating point between 0 and 2

- nextDouble() returns $0 \leq x < 1$

- nextDouble() * 2 returns $0 \leq x < 2$

- Solution:

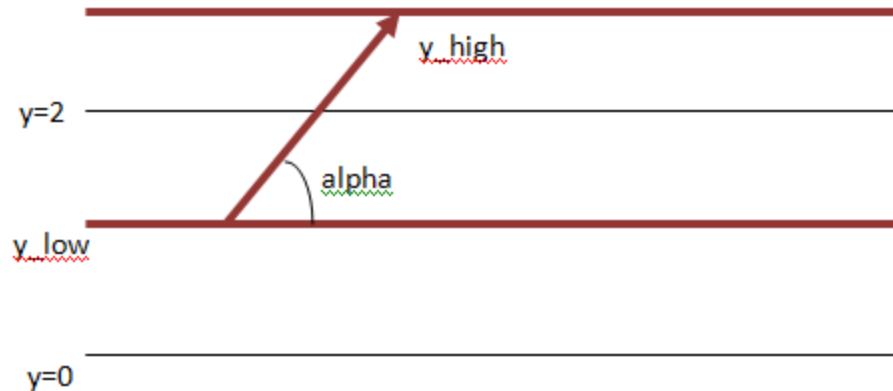
- ```
Random rand = new Random();
```

- ```
double r = rng.nextDouble()*2;
```

Random

- Simulate the Buffon needle experiment (1700s)

- Drop a needle onto a paper with lines drawn two inches apart. If the needle lands on a line, it counts as a hit.



- Call the line below to the bottom of the needle $y=0$. Then y_{low} will be between 0 and 2. The angle must be between 0 and 180 degrees.

- Geometry: $y_{high} = y_{low} + \sin(\alpha)$

- Java to drop the needle:

```
Random rng = new Random();  
double yLow = 2* rng.nextDouble();  
double angle = 180*rng.nextDouble();  
double yHigh = yLow + Math.sin(Math.toRadians(angle)) ;
```

- If $y_{high} > 2$, then we count this as a hit

- Repeat this multiple times and count the number of hits (inside a loop). Tries/Hits will approximate pi.

Code for the experiment:

Random

```
int nuTries = 10000;
int hits = 0;
Random rng = new Random();

for (int i=0; i<nuTries; i++)
{
    double yLow = 2*rng.nextDouble();
    double alpha = 180*nextDouble();
    double yHigh = yLow + Math.sin(Math.toRadians(alpha)) ;

    if (yHigh >= 2)
        hits++;
}

System.out.printf("Tries/Hits = %8.5f\n", nuTries/hits);
```

For 10,000, we get 3.08928

For 100,000, we get 3.14204

Example

- Write a program that simulates flipping a coin each time the user asks. Keep track of the number of heads and tails thrown. (Parts of the Code we will need below)

```
double nuHeads = 0;
```

```
double nuTails = 0;
```

```
System.out.println("Do you wish to flip a coin? Enter 1 for yes, -1 to quit");
```

```
choice = in.nextInt();
```

```
int result = rng.nextInt(2); //0 or 1
```

```
if (result == 0)
```

```
{
```

```
    System.out.println("Tails");
```

```
    nuTails++;
```

```
}
```

```
else
```

```
{
```

```
    System.out.println("Heads");
```

```
    nuHeads++;
```

```
}
```

```
System.out.println("Number of Heads: " + nuHeads +  
    "\nNumber of Tails: " + nuTails + "\nFraction Heads: " +  
    nuHeads / (nuHeads + nuTails));
```

Sample Results:

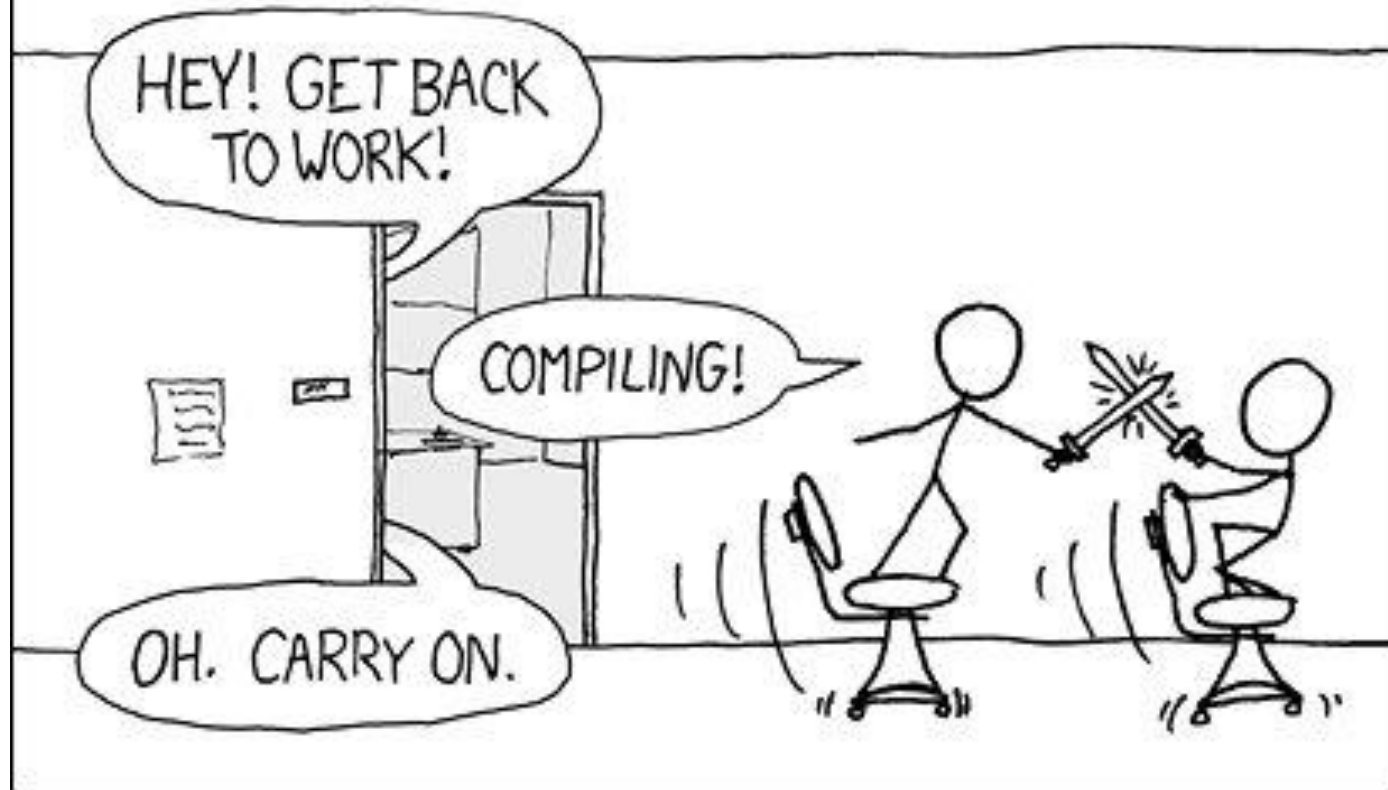
Number of Heads: 23.0

Number of Tails: 15.0

Fraction Heads: 0.6052631578947368

See [CoinToss.java](#) on our website for complete code

THE #1 PROGRAMMER EXCUSE
FOR LEGITIMATELY SLACKING OFF:
"MY CODE'S COMPILING."



Example 1

- Write program that asks the user for the width and height of a rectangle. Then generate three random numbers in the range 0 to 255 (then we will use these to make a color for our rectangle).

```
//ask user for size of rectangle
```

```
Scanner in = new Scanner(System.in);
```

```
System.out.println("Enter height of rectangle: ");
```

```
int height = in.nextInt();
```

```
System.out.println("Enter width of rectangle: ");
```

```
int width = in.nextInt();
```

```
//generate a random color by mixing Red, Green, and Blue
```

```
Random rng = new Random();
```

```
int red = rng.nextInt(256);
```

```
int blue = rng.nextInt(256);
```

```
int green = rng.nextInt(256);
```

Example 1 (cont)

```
Scanner in = new Scanner(System.in);
System.out.println("Enter height of rectangle: "); int height = in.nextInt();
System.out.println("Enter width of rectangle: "); int width = in.nextInt();

Random rng = new Random();
Color myColor = new Color(rng.nextInt(256), rng.nextInt(256), rng.nextInt(256));

Rectangle box = new Rectangle (5,10, width, height); //make rectangle
RectangleComponent rectToDraw = new RectangleComponent(box, myColor); //drawable box

//make a frame for the drawing to occur in
JFrame frame = new JFrame();
frame.setSize(300,400);
frame.setTitle("Rectangle");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); //end the program when the
    window is closed

//put the drawable rectangle in the frame we made and make it able to be seen
frame.add(rectToDraw);
frame.setVisible(true);
```

Example 1 (cont)

Note: we need the following imports for the previous slide to work.

```
import java.awt.Color;
import java.awt.Rectangle;
import java.util.Random;
import java.util.Scanner;
import javax.swing.JFrame;
```

We need the following imports for the code on the next slide to work:

```
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Rectangle;
import javax.swing.JComponent;
```

(They are present in the online version of this code. Also Eclipse will remind you.)

Example 1 (cont)

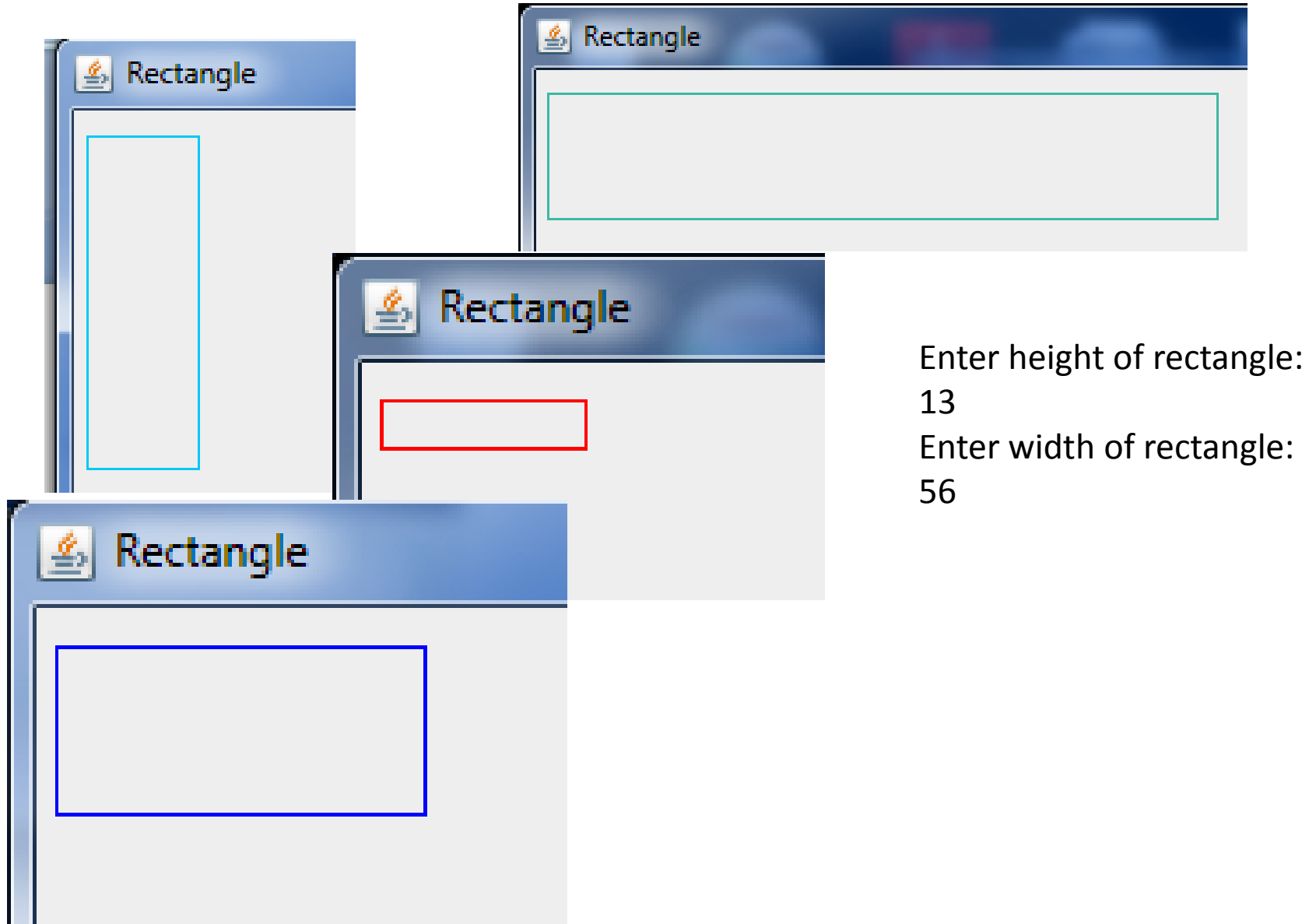
//components are used to draw. “extends” tells the computer to treat this class like it is also
//a component.

```
public class RectangleComponent extends JComponent
{
    private Rectangle rectangle; //the box we made
    private Color color;

    //construct the RectangleComponent
    public RectangleComponent(Rectangle box, Color whatColor)
    {
        rectangle = box; //set the variables we need for the drawing method
        color = whatColor;
    }

    //necessary to draw
    public void paintComponent(Graphics g)
    {
        Graphics2D g2 = (Graphics2D) g; //necessary due to changes in Java
        g2.setColor(color); //what color should I draw in
        g2.draw(rectangle); //draw the box that we made
    }
}
```

Example 1 (sample output)



Example 2

- We are given a string that contains the location of a meeting we must attend. Assume the first words are the name of the building, and the last three characters are the room number. Create separate variables to hold the building, floor number, and room number on that floor.

```
String location; //provided
```

```
String number = location.substring( location.length() -3);
```

Suppose location is "Mills Hall 128"

M	i	l	l	s		H	a	l	l		1	2	8
0	1	2	3	4	5	6	7	8	9	10	11	12	13

`location.length() == 14`

`location.length() - 1 == 13`



Example 2 (cont)

- We are given a string that contains the location of a meeting we must attend. Assume the first words are the name of the building, and the last three characters are the room number. Create separate variables to hold the building, floor number, and room number on that floor.

```
String location; //provided
```

```
String number = location.substring( location.length() -3);
```

```
String building = location.substring(0, location.length()-3);
```

```
String buildingName = building.trim(); //consume whitespace at front or back
```

```
String floor = number.substring(0,1); //only the first of the numbers
```

```
String room = number.substring(1); //all the other numbers
```

Example 2 (cont)

```
String location; //provided
```

```
String number = location.substring( location.length() -3);
```

```
String building = location.substring(0, location.length()-3);
```

```
String buildingName = building.trim(); //consume whitespace at front or back
```

```
String floor = number.substring(0,1); //only the first of the numbers
```

```
String room = number.substring(1); //all the other numbers
```

```
int floorNumber = Integer.parseInt( floor.trim() );
```

```
int roomNumber = Integer.parseInt(room.trim() );
```

Writing a Complete Java Program

- Goal for this chapter: write a complete java program from memory.
- Incrementally develop programs
- Problem: Write a program to help elementary students practice their math skills.

- Start with only multiplication.

```
Random rng = new Random();  
Scanner in = new Scanner(System.in);
```

```
for (int i=0; i<10; i++)  
{  
    int num1 = rng.nextInt(10);  
    int num2 = rng.nextInt(10);  
  
    int answer;  
    int product = num1*num2;  
  
    do  
    {  
        System.out.printf("%d times %d = ", num1, num2);  
        if( in.hasNextInt() )  
            answer = in.nextInt();  
        else  
            in.nextLine();  
    }  
    while(answer != product);  
}
```

- Extension: allow the user to choose what type of math to practice.

```
int MULT = 1; //create constants
```

```
int DIVIDE = 2;
```

```
int ADD = 3;
```

```
int SUB = 4;
```

```
int questionType = 0;
```

```
while (questionType < MULT || questionType > SUB)
```

```
{
```

```
    System.out.println("Choose type of problem: 1 for multiplication, 2 for division, 3 for  
        addition, 4 for subtraction");
```

```
    if (in.hasNextInt())
```

```
        questionType = in.nextInt();
```

```
    else
```

```
        in.nextLine();
```

```
}
```

- Addition and Subtraction are easy to code

```
int sum = num1 + num2;
```

```
int diff = num1 - num2;
```

- In elementary school, division either always results in a whole number, or we write the quotient and the remainder both.

```
int quotient = num1/num2;
```

```
int remainder = num1 % num2;
```

- Include addition and subtraction as well as multiplication in the correct result

```
int correctAnswer;
```

```
if (questionType == MULT)
```

```
    correctAnswer = num1 * num2;
```

```
else if(questionType == ADD)
```

```
    correctAnswer = num1 + num2;
```

```
else //must be subtraction
```

```
    correctAnswer = num1 - num2;
```

```
int studentAnswer = correctAnswer - 1;//initialize to wrong answer
```

```
int correctAnswer;
if (questionType == MULT) {
    correctAnswer = num1 * num2;
    System.out.printf("%d times %d = ", num1, num2);
}
else if(questionType == ADD) {
    correctAnswer = num1 + num2;
    System.out.printf("%d plus %d = ", num1, num2);
}
else //must be subtraction {
    correctAnswer = num1 - num2;
    System.out.printf("%d minus %d = ", num1, num2);
}
int studentAnswer = correctAnswer - 1;//initialize to wrong answer

do {
if( in.hasNextInt() )
    studentAnswer = in.nextInt();
else
    in.nextLine();
if (studentAnswer != correctAnswer)
    System.out.printf(" Incorrect. Please try again: " );
}

}while(studentAnswer != correctAnswer);
```

- Include division as well

- We need both a quotient and a remainder, unlike the other three operands
- We could add checks so that the two random numbers generated resulted in a remainder of zero.
- We could ask the user for two integers, first the quotient, then the remainder
- We could use Strings and have them input the result as the following:

$$10/4 = 2 \text{ R } 2$$

```
if( questionType == DIVIDE)
{

}
else
{
    //Insert code from last slide here
}
```

- Add instructions for division

```
System.out.println("Please give your answer as quotient R remainder. " +  
    "For example if the question is 13/5, enter: 2 R 3 ");
```

- If the question is a division problem

```
if (num2 == 0)//avoid divideByZero errors
```

```
    num2++;
```

```
int quotient = num1 / num2; //integer division
```

```
int remainder = num1 % num2;
```

```
String answer = "";
```

```
System.out.printf("%d / %d = ", num1, num2);
```

```
do
```

```
{
```

```
    answer = in.nextLine();
```

```
}while (answer.length() == 0);
```

```
String studentQ = answer.trim().substring(0,1); //first character as a string;
```

```
int studentQuotient = Integer.parseInt(studentQ);
```

```
int studentRemainder = Integer.parseInt(answer.substring(answer.length()-1));
```

```
if (studentQuotient == quotient && studentRemainder == remainder)
```

```
    System.out.println("Correct");
```

```
else
```

```
    System.out.println("I'm sorry, that is incorrect.");
```

- Our program is now complete, and we could make further modifications if desired.
- (Full program is posted on our webpage)
- From a few sample runs:

0 times 4 = 0

Correct.

6 times 9 = 5

Incorrect. Please try again: 54

Correct.

1 plus 7 = 8

Correct.

8 plus 4 = 12

Correct.

1 plus 4 = 5

Correct.

1 minus 8 = -7

Correct.

6 minus 2 = 4

Correct.

7 / 6 = 1 R 1

Correct

5 / 3 = 1 R 2

Correct

8 / 3 = 2 R 2

Correct

0 / 4 = 0 R 0

Correct

2 / 3 = 0 R 2

Correct



DEPARTMENT OF
Computer Sciences
UNIVERSITY OF WISCONSIN-MADISON