Announcements

• Reading lecture notes?
  • Some homework submissions suggest that some of you are NOT!
  • Please review lecture notes before you try homework

• Problem Set 1 (PS 1) is due Tuesday, 9/13, 11:55pm KST

• See the updated tutoring schedule and see if it would work for you

• Labs from now on – go to your lab section as in schedule
  • Bring your laptop

• Reading assignment for this week: Chapter 2 of Liang

• Lecture slide format: 1x or 4x?
Operator precedence

• **precedence**: order in which operators are evaluated

• Generally, operators evaluate left-to-right
  
  4 – 2 – 5 is (4 – 2) – 5 which is –3

• But *, /, and % have a higher level of precedence than + and –
  
  3 + 4 * 2 is 11
  
  4 – 5 / 2 * 5 is –6

• Parentheses can force a certain order of evaluation:
  
  • (4 + 3) * 2 is 14

• Spacing does not affect order of evaluation
  
  • 5+3 * 4-2 is 15
Real numbers (double)

- Examples: 5.036, -32.0, 2.54e12
  - Placing .0 or . after an integer makes it a double

- The operators +, -, *, /, %, (, ) all still work with double
  - / produces an exact answer: 13.0 / 2.0 is 6.5
  - Precedence is the same: ( ) before * / % before + -
Mixing types

- When `int` and `double` are mixed, the result is a double
  
  \[3.4 \times 2\] is 6.8

- The conversion is per-operator, affecting only its operands

\[
8 \div 3 \times 1.3 + 4 \div 2
\]

- \(->\) \[2 \times 1.3 + 4 \div 2\]
- \(->\) \[2.6 + 4 \div 2\]
- \(->\) \[2.6 + 2\]
- \(->\) \[4.6\]
Examples: order of operations

- $4 + 2 \times 3 / 2$
- $4 + 16 \% 3 - 2$
- $7 - 3 \times 4 / 5$
- $2.3 \times 3 \times 4 / 5 + 3.5$
- $68 \% 10 + 3 \% 8 \% 3$

Answers: 7, 3, 5, 58.7, 8
String concatenation

- **string concatenation**: using + between a string and another value to make a longer string
  - “hello” + 24 is “hello24”
  - 10 + “xyz” + 32 is “10xyz32”
  - 2 + 3 + “apple” is “5apple”
  - “apple” + 3 * 4 is “apple12”
  - “3” + 1 + 1 is “311”
  - 5 – 2 + “sunyk” is “3sunyk”

- Use + to print a string and an expression’s value together
  - System.out.println(“Interest: “ + (balance * rate));
  - Output: Interest: 22.6 if balance * rate is 22.6
Printing a variable’s value

- Use + to print a string and a variable’s current value on one line
  ```java
double grade = 94.5;
System.out.println("Your grade was " + grade);

int students = 32;
System.out.println("There are " + students + " students in CSE 114.");
```

- Output:
  ```
  Your grade was 94.5
  There are 32 students in CSE 114.
  ```

- Underneath the hood, Java converts a variable’s value to a string and then concatenates it to the other string
Example: area of circle

- Let’s see how to write a simple program to calculate the area of a circle given its radius

```java
public class CircleStuff {
    public static void main(String[] args) {
        double radius = 5.4;
        System.out.println("radius = \" + radius);
        double area = Math.PI * radius * radius;
        System.out.println("area = \" + area);
    }
}
```

- Can we read the radius from the user (via key board)?
Reading console input

• We can get user input by reading a value from the keyboard

• We will use a **Scanner** object to do this
  • An *object* is a collection of data along with operations (methods) we can perform on that data
  • More about objects later in the semester

• **Scanner input = new Scanner(System.in);**
  • *input* is the name of our **Scanner** object
  • **System.in** refers to **standard input device**, the keyboard
Packages

- Java provides many names (types, classes, e.g., *Scanner*)
- We put related names into **packages**
- The Java API contains dozens of packages
- The *java.lang* package contains basic, core classes in the language
  - This is the default package in Java
- The *java.util* package contains useful utility classes, such as *Scanner*
Packages

- To use a class in a non-default package, you must do one of the following:

  1. Refer to the member by its fully qualified name:
     ```java
     java.util.Scanner input = new java.util.Scanner(System.in);
     ```

  2. Import the class from the package:
     ```java
     import java.util.Scanner;
     ```

  3. Import the class’ entire package:
     ```java
     import java.util.*;  // This is called a wildcard import
     ```
import java.util.Scanner;

public class CircleStuff {
    public static void main(String[] args) {
        double radius;
        Scanner input = new Scanner(System.in);

        // Prompt the user to enter a radius
        System.out.print("Enter a radius: ");
        radius = input.nextDouble();
        System.out.println("radius = "+radius);

        double area = Math.PI * radius * radius;
        System.out.println("area = "+area);
    }
}

Area of circle with console input
Other types of console input

- We can read other types of input using other methods in `Scanner`.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>nextInt()</code></td>
<td>reads an <code>int</code> from the user and returns it</td>
</tr>
<tr>
<td><code>nextDouble()</code></td>
<td>reads a <code>double</code> from the user</td>
</tr>
<tr>
<td><code>next()</code></td>
<td>reads a one-word <code>String</code> from the user</td>
</tr>
<tr>
<td><code>nextLine()</code></td>
<td>reads a one-<code>line</code> <code>String</code> from the user</td>
</tr>
</tbody>
</table>

- See `Read.java`
Augmented operators

- Syntactically simpler forms (syntactic sugar)

- \( i++ \) for \( i = i + 1 \)
- \( i-- \) for \( i = i - 1 \)
- \( i += 3 \) for \( i = i + 3 \)
- \( i -= 3 \) for \( i = i - 3 \)
- \( i *= 3 \) for \( i = i * 3 \)
- \( i /= 3 \) for \( i = i / 3 \)
- \( i %= 3 \) for \( i = i % 3 \)
Next time

- Will start Chapter 3

Happy Chu Suk!