First Programs

CSE 1310 – Introduction to Computers and Programming
Vassilis Athitsos
University of Texas at Arlington
Output

• System.out.println(...) prints out something.
  – System.out.println is the first piece of Java that we learn in this class.

• We will see in detail what kind of things can get printed.

• In the beginning, the things we care about printing are:
  – Numbers.
  – Strings (text).
Examples of System.out.println

Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println("Have a nice day.");
        System.out.println(6.3 + 12/7);
    }
}
```

Output:

```
run:
Have a nice day.
7.3
BUILD SUCCESSFUL (total time: 0 seconds)
```

In Netbeans, the program output always starts with "run:" and ends with "BUILD SUCCESSFUL ...".
Examples of System.out.println

Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println("Have a nice day.");
        System.out.println(6.3 + 12/7);
    }
}
```

Output:

```
Have a nice day.
7.3
```

In Netbeans, the program output always starts with "run:" and ends with "BUILD SUCCESSFUL ...".

From now on, we will not be showing those lines.

We will call "program output" what is between those lines.
Syntax of System.out.println

Program:

```java
public class hello1 {  
    public static void main(String[] args) {  
        System.out.println("Have a nice day.");  
        System.out.println(6.3 + 12/7);  
    }  
}
```

- What you want to print is called **the argument**.
- To use System.out.println, you write a line like this:
  - System.out.println(\textit{argument});
  - In other words, you write \texttt{System.out.println}, followed by a left parenthesis, followed by an argument, followed by a right parenthesis, followed by a semicolon.
Syntax of `System.out.println`

- If the argument is text (also called a **string**), then it must be enclosed in double quotes.
- If the argument is a numerical expression, then `System.out.println` prints the **result** of that expression.

Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println("Have a nice day.");
        System.out.println(6.3 + 12/7);
    }
}
```
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

  System.out.println("hello");

  System.out.println(hello);
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

```java
System.out.println("hello");
```
Correct, prints hello.

```java
System.out.println(hello);
```
Incorrect, missing double quotes. Will not run.
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

```java
System.out.println("6.3 + 12/7");
```

```java
System.out.println(6.3 + 12/7);
```
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

    System.out.println("6.3 + 12/7");
Correct, prints 6.3 + 12/7
Note that the argument here is **text**.

    System.out.println(6.3 + 12/7);
Correct, prints 7.3
Note that the argument here is a **numerical expression**.
Syntax of `System.out.println`

- Is each of these lines correct or not? If correct, what will it print?

```java
System.out.println("hello")
System.out.println(6.3 + 12/7)
```
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

`System.out.println("hello")`
Incorrect. Missing semicolon at the end. Will not run.

`System.out.println(6.3 + 12/7)`
Incorrect. Missing semicolon at the end. Will not run.
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

    System.out.println "hello";

    System.out.println 6.3 + 12/7;
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

    System.out.println "hello";
Incorrect. Missing parentheses. Will not run.

    System.out.println 6.3 + 12/7;
Incorrect. Missing parentheses. Will not run.
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

```java
System.out.println "hello" ();
```

```java
System.out.println 6.3 + 12/7 ();
```
Syntax of System.out.println

• Is each of these lines correct or not? If correct, what will it print?

    System.out.println "hello" ();
Incorrect. Misplaced parentheses. Will not run.

    System.out.println 6.3 + 12/7 ();
Incorrect. Misplaced parentheses. Will not run.
Syntax of System.out.println

• As we saw a few slides ago, to use System.out.println, you write a line like this:
  – System.out.println(argument);

• Java (like any programming language) is very strict.
• If you do not follow the syntax **EXACTLY**, it will refuse to execute that line.
• This is true not only for System.out.println, but for any syntax rules that we will see in this course.
Java as a Calculator.

- We can type in arbitrary numerical expressions, and Java evaluates them.
- This is still not that exciting.
- However, such calculations are a useful building block for real programs.

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println((23*3) + 12/4.5);
        System.out.println(6.3 + 12/7 - 4);
    }
}
```

Output:
71.66666666666667
3.3
More Math Calculations

public class hello1 {
    public static void main(String[] args) {
        System.out.println(Math.pow(2, 10));
        System.out.println(8 * Math.pow(2 + 3.5/7, 4));
        System.out.println(Math.sqrt(3));
        System.out.println(4 - Math.sqrt(3+5/7.2));
    }
}

Output:
1024.0
312.5
1.7320508075688772
2.077906234221534

• Powers:
  – $2^{10}$ becomes Math.pow(2, 10)
  – $8 \left(2 + \frac{3.5}{7}\right)^4$ becomes 8 * Math.pow(2 + 3.5/7, 4)

• Roots
  – $\sqrt{3}$ becomes Math.sqrt(3)
  – $4 - \sqrt{3 + \frac{5}{7.2}}$ becomes 4 - Math.sqrt(3+5/7.2)
public class hello1 {
    public static void main(String[] args) {
        System.out.println(Math.PI);
        System.out.println(Math.sin(Math.PI / 2));
        System.out.println(Math.cos(Math.PI / 2));
        System.out.println(Math.tan(Math.PI / 2));
        System.out.println(Math.log(12.5));
    }
}

• The pi constant: Math.PI
• The sine of x: Math.sin(x)
• The cosine of x: Math.cos(x)
• The tangent of x: Math.tan(x)
• The natural logarithm of x: Math.log(x)
**Division: Floating Point and Integer**

- **Floating point division:**
  - `7.0 / 4.0`
  - `7 / 4.0`
  - `7.0 / 4`
  - They all evaluate to 1.75

- **Integer division:**
  - `7 / 4` evaluates to 1
  - `7 % 4` produces the **remainder** of 7/4, so it evaluates to 3.

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println(7.0 / 4.0);
        System.out.println(7 / 4.0);
        System.out.println(7.0 / 4);
        System.out.println(7 / 4);
        System.out.println(7 % 4);
    }
}
Output:
1.75
1.75
1.75
1
3```
Circumference and Area of Circle

• We want to write a program to compute the circumference and area of a circle.
• What do the the circumference and area of a circle depend on?
Circumference and Area of Circle

• We want to write a program to compute the circumference and area of a circle.
• What do the circumference and area of a circle depend on?
  – The radius of the circle.
• circumference = 2 * pi * radius
• area = pi * radius²
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.
• Computing the circumference of the circle:
  – Circumference = $2 \times \pi \times \text{radius}$
  – Code?
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.
• Computing the circumference of the circle:
  – Circumference = 2 * pi * radius

```
System.out.println(2 * Math.PI * 20.231);
```
Output: 127.11512194955021
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.
• Computing the circumference of the circle:
  – Circumference = 2 * pi * radius
  
  ```java
  System.out.println(2 * Math.PI * 20.231);
  ```

  Output: 127.11512194955021

• Computing the area of the circle:
  – area = pi * radius²
  – Code?
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.

• Computing the circumference of the circle:
  – Circumference = 2 * pi * radius

```
System.out.println(2 * Math.PI * 20.231);
```
Output: 127.11512194955021

• Computing the area of the circle:
  – area = pi * radius²

```
System.out.println(Math.PI * Math.pow(20.231, 2));
```
Output: 1285.8330160806754
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.

• Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println(2 * Math.PI * 20.231);
        System.out.println(Math.PI * Math.pow(20.231, 2));
    }
}
```

• Is this a good program to sell to a user?
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.

• Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println(2 * Math.PI * 20.231);
        System.out.println(Math.PI * Math.pow(20.231, 2));
    }
}
```

• Is this a good program to sell to a user?
• No: the only way for the user to use this program is to modify the code every time, to specify the radius.
• That is bad. Users should not need to be programmers.
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.

• Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println(2 * Math.PI * 20.231);
        System.out.println(Math.PI * Math.pow(20.231, 2));
    }
}
```

• Any other issues/problems with this program?
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.

• Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println(2 * Math.PI * 20.231);
        System.out.println(Math.PI * Math.pow(20.231, 2));
    }
}
```

• Any other issues/problems with this program?
  – The radius is specified TWICE.
  – This is bad practice, introduces the risk of errors.
  – Also, more painful to change the radius, we must change it in two places.
Circumference and Area of Circle

• Suppose we have a circle with radius = 20.231.

• Program:

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println(2 * Math.PI * 20.231);
        System.out.println(Math.PI * Math.pow(20.231, 2));
    }
}
```

• Any other issues/problems with this program?

• The program is hard to read and understand.
  – If you show it to a programmer, is it clear what the program is supposed to be doing?
  – The output is just numbers, not very user-friendly.
Using Variables

public class hello1 {  
    public static void main(String[] args) {  
        double radius = 20.231;  
        double circumference = 2 * Math.PI * radius;  
        double area = Math.PI * Math.pow(radius, 2);  
        System.out.println(circumference);  
        System.out.println(area);  
    }  
}  

• This code has the same output as the previous version.

• However:
  – The radius is specified only once (better than specifying twice).
  – If you show this program to any programmer, it is fairly obvious what it does (easy to read).
Declaring a Variable

• At any point, you can create a variable, by doing a **variable declaration**.

• Syntax for variable declaration:

  
  \[
  \text{type variable\_name} = \text{initial\_value};
  \]

• For example:

  ```
  int x = 123;
  int number\_of\_fingers = 5;
  double radius = 20.231;
  ```
Types

• There are many different types in Java.
• However, initially, you just need to know these two:
  – double
  – int
• You need to think carefully, and use the correct type for your variable.
• For integers (positive and negative), use int.
• For floating point numbers, use double.
Variable Names

• The textbook describes the rules for variable names.

• Here is a simplified version:
  – variable names should start with a letter (upper or lower case).
  – variable names should only include letters, numbers, and underscores.
  – variable names are case-sensitive.
  – variable names cannot be equal to reserved words, such as double, class, int, public, ...
Using Variables

• After you declare a variable, you can use it in the rest of the code:
  – You can use its value.
  – You can change its value. This is called **assignment.**

```java
public class hello1 {
    public static void main(String[] args) {
        int candies = 5;
        System.out.println(candies);
        candies = 7;
        System.out.println(candies);
        candies = candies + 10;
        System.out.println(candies);
    }
}
```

Output:
Using Variables

• After you declare a variable, you can use it in the rest of the code:
  – You can use its value.
  – You can change its value. This is called assignment.

```java
public class hello1 {
    public static void main(String[] args) {
        int candies = 5;
        System.out.println(candies);
        candies = 7;
        System.out.println(candies);
        candies = candies + 10;
        System.out.println(candies);
    }
}
```

Output:
5
7
17
Declarations and Assignments

• In this program:
  – Which lines of code are declarations?
  – Which lines of code are assignments?

```java
public class hello1 {
    public static void main(String[] args) {
        int candies = 5;
        System.out.println(candies);
        candies = 7;
        System.out.println(candies);
        candies = candies + 10;
        System.out.println(candies);
    }
}
```

Output:

```
5
7
17
```
Declarations and Assignments

• In this program:
  – Which lines of code are declarations?
    ```java
    int candies = 5;
    ```
  – Which lines of code are assignments?
    ```java
    candies = 7;
    candies = candies + 10;
    ```

```java
public class hello1 {
    public static void main(String[] args) {
        int candies = 5;
        System.out.println(candies);
        candies = 7;
        System.out.println(candies);
        candies = candies + 10;
        System.out.println(candies);
    }
}
```

Output:

```
5
7
17
```
Returning to the Circles Program

Version with variables:

```java
public class hello1 {
    public static void main(String[] args) {
        double radius = 20.231;
        double circumference = 2 * Math.PI * radius;
        double area = Math.PI * Math.pow(radius, 2);
        System.out.println(circumference);
        System.out.println(area);
    }
}
```

• Which lines are declarations?
• Which lines are assignments?
Returning to the Circles Program

Version with variables:

```java
public class hello1 {
    public static void main(String[] args) {
        double radius = 20.231;
        double circumference = 2 * Math.PI * radius;
        double area = Math.PI * Math.pow(radius, 2);
        System.out.println(circumference);
        System.out.println(area);
    }
}
```

- Which lines are declarations? Shown in red.
- Which lines are assignments? None.
Returning to the Circles Program

Version with variables:

```java
public class hello1 {
    public static void main(String[] args) {
        double radius = 20.231;
        double circumference = 2 * Math.PI * radius;
        double area = Math.PI * Math.pow(radius, 2);
        System.out.println(circumference);
        System.out.println(area);
    }
}
```

• Problem: the radius is hardcoded.
  – Why is this a problem?
Problem: Radius is Hardcoded

• Why is this a problem?
• Biggest reason: the user needs to be a programmer.
  – You cannot use this program without changing the program.
Solution

- Allow the user to enter the radius value as input.
import java.util.Scanner;

public class hello1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        System.out.printf("Please enter the radius: ");
        double radius = in.nextDouble();
        double circumference = 2 * Math.PI * radius;
        double area = Math.PI * Math.pow(radius, 2);
        System.out.println(circumference);
        System.out.println(area);
    }
}

Revised Program with User Input

• There are several new things here:
  – the import line.
  – The Scanner object.
  – The System.out.printf method.
import java.util.Scanner;

public class hello1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        System.out.printf("Please enter the radius: ");
        double radius = in.nextDouble();
        double circumference = 2 * Math.PI * radius;
        double area = Math.PI * Math.pow(radius, 2);
        System.out.println(circumference);
        System.out.println(area);
    }
}

• The **Scanner** object allows us to obtain user input.
• To create a **Scanner** object, we need to:
  – Put the import statement at the top of the Java file.
  – Create a Scanner object, as shown in the first line of the main method:
    ```java
    Scanner in = new Scanner(System.in);
    ```
import java.util.Scanner;

public class hello1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        System.out.printf("Please enter the radius: ");
        double radius = in.nextDouble();
        double circumference = 2 * Math.PI * radius;
        double area = Math.PI * Math.pow(radius, 2);
        System.out.println(circumference);
        System.out.println(area);
    }
}

• The `System.out.printf` method is a more powerful version of the `System.out.println` method.
• We will see more details in a few days.
• One difference is that `System.out.println` always prints a new line at the end, whereas `System.out.printf` does not.
println and printf

```java
public class hello1 {
    public static void main(String[] args) {
        System.out.println("hello");
        System.out.printf("hello
");
    }
}
```

• These two lines do the exact same thing:

```java
System.out.println("hello");
System.out.printf("hello\n");
```
import java.util.Scanner;

public class example1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.printf("Please enter number of weeks: ");
        int weeks = in.nextInt();
        int days = weeks * 7;
        System.out.printf("There are %d days in %d weeks\n", days, weeks);
    }
}
import java.util.Scanner;

public class example1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.printf("Please enter number of weeks: ");
        int weeks = in.nextInt();
        int days = weeks * 7;
        System.out.printf("There are %d days in %d weeks\n", days, weeks);
    }
}
Another Example: Computing the Average of Three Numbers

```java
import java.util.Scanner;

public class example1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.printf("Please enter the first number: ");
        double n1 = in.nextDouble();
        System.out.printf("Please enter the second number: ");
        double n2 = in.nextDouble();
        System.out.printf("Please enter the third number: ");
        double n3 = in.nextDouble();

        double average = (n1 + n2 + n3) / 3.0;
        System.out.printf("The average is %.2f\n", average);
    }
}
```
Another Example: Computing Gravity

```java
import java.util.Scanner;

public class example1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.printf("Please enter the first mass: ");
        double m1 = in.nextDouble();
        System.out.printf("Please enter the second mass: ");
        double m2 = in.nextDouble();
        System.out.printf("Please enter the radius: ");
        double r = in.nextDouble();

        double G = 6.694E-11;
        double gravity = G * m1 * m2 / (r * r);
        System.out.printf("The gravity force is \%fn", gravity);
    }
}
```
Comments

/* A program that converts weeks into days.
   Written on 7/15/2015. */

import java.util.Scanner;

public class example1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.printf("Enter number of weeks: ");
        int weeks = in.nextInt();

        // Here is where we convert weeks into days.
        int days = weeks * 7;
        System.out.printf("Result: %d days
", days);
    }
}

• Comments allow you to make notes on the program for yourself, and for other people reading your code.
• Comments are ignored by Java.
• Single line comments: they start with // (see line in green above)
• Multiple-line comments: they start with /*, end with */ (see lines in red)
import java.util.Scanner;

public class example1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);  // Create scanner object.
        System.out.printf("Enter number of weeks: ");
        int weeks = in.nextInt();  // Get user input

        int days = weeks * 7;  // Converting weeks into days.

        System.out.printf("Result: %d days\n", days);
    }
}

• Comments starting with // can be placed at the end of a line (see code marked in red)
Some Guidelines

• To learn how to code, you need PRACTICE.
  – What will usually not work:
    • Listen to the lectures.
    • Go and try to do the assignments.
  – What will usually work:
    • Listen to the lectures and KEEP NOTES.
    • Actually run every piece of code that we do in class.
    • Understand every line of every piece of code we do in class.
    • Think of variations of what we do in class, and try them.
      – Predict what the variation will do, and verify by running it.
    • Then try the assignments.
Some Guidelines

• You need to understand the terminology:
  – method, string, double, int, main class name,
    numerical expression, variable, declaration,
    assignment, newline character

• You will encounter many terms in this course. YOU NEED TO LEARN EXACTLY WHAT THEY MEAN.

• DO NOT RELY ON ENGLISH. These terms have meanings in conversational English that are only vaguely related with their meaning in programming.