04: Keyboard Input

Standard Input and Output

Standard Input and Output

What I love about Java

- Java's philosophy: Compile once. Run anywhere.
- If I own a Mac and you own a PC, my Java programs will still work on your computer. Your Java programs will work on my computer.
- Java encourages sharing, even if people own different types of computers that don't normally place nice.
- Java is also a fast language. It's not the fastest language, but it's faster than most languages (including Python).

Java API

- Java has many built-in classes for the programmer to use. This list of classes numbers in the thousands.
- This collection is called the **Java Applications Programming Interface**.
- Shorter, this collection is called the Java API.
- A few classes are automatically imported into every program and do not require import lines.
- Such classes include **System** and **Math**.

System

Inside the **System** class, there are two objects which are available to all programs:

- **in**: This is the standard input. It is typically the keyboard.
- **out**: This is the standard output. It is typically the monitor.

Printing output to the monitor

We use **System.out** to print things to the screen:

- System.out.println("Hello, world!");
  - Prints "Hello, world!" to the screen with a end-of-line character at the end so that the next content printed will be on a new line.
- System.out.print("Hello, world!");
  - Prints "Hello, world!" to the screen without an end-of-line character. The next content printed will appear on the same line.

What will this print?

Work this out on paper. If you are unsure, write a small Java program to verify your answer. Be sure to put this content inside the **main** method.

```
System.out.print("A ");
System.out.println("B ");
System.out.print("C ");
System.out.println("D ");
```

Java Escape Sequences

All Java Escape Sequences begin with a backslash character. “\”. You may use them inside of strings to change how the string behaves.

- n: newline
- t: tab
- \: Causes a backslash to be printed.
- ': Causes a single quote to be printed.
- ": Causes a double quote to be printed.

Java Escape Sequence Example

```
// Print tabs between apples, oranges, and grapes
System.out.println("apples\toranges\tgrapes");

// Print new lines between apples, oranges, and grapes
System.out.println("apples\noranges\ngrapes");
```

Variable Definitions

- In Java (this is not true for every language) you must write a variable definition (also called a variable declaration).
- The Java compiler will provide an error if it encounters a variable in an expression which has not been defined.
- There are many different sizes and types of variables and we will cover these in more detail in this course.

Example of Variable output

```
int count = 5; // Example of Summation
int value = count + 1;
// Example of Concatenation
System.out.println("count is equal to "+count+"."); // 5
System.out.println("value is equal to "+value+"."); // 6
// Also prints 6
System.out.println("count+1 is equal to "+(count+1)+".");
```

The meaning of "+" operator in Java will change depending on the context. If both operands are numerical, then the two numbers will be added. If one or both of the operands is a String, Java will attempt to convert any non-String operands to a string and **concatenate** the String objects. If you wish to do addition inside of an output statement (not recommended), it must be wrapped in parentheses.

Rules for Naming Variables

- The first character must be one of the following:
  - upper or lower case letter
  - the underscore
  - the dollar sign
- Characters after the first can be any of the above and may include the digits 0 through 9.
- Variable names must not be reserved words (such as “class” or “int”).
- Variable names are case sensitive. “itemsOrdered” is different from “itemsordered”.

Soft Rules for Naming Variables

- "$$$", while legal, is a poor choice for a variable name.
- Begin your variables with a lower case letter.
- Begin your classes with an upper case letter (like String).
- Use nouns.
- Do not use jokes.
- Do not use abbreviations.
- Call things what they are. (This is a difficult life skill to learn.)
  - "thing" and "stuff" are poor choices for variable names.

Variable Declaration

- Variables must be declared.
- After they are declared, they must defined before they are used.
  - The defined value or expression must match the type of the variable.
  - Reading the value from a keyboard counts as defining a variable.
- The variable may then be printed to the screen.

What's wrong with the following code?

```
double pi = 3.14159;
double radius;
double area = pi * radius * radius;

System.out.println("The area is: "+area);
```

What's wrong with the following code?

```
double pi = 3.14159;
double radius;
double area = pi * radius * radius;

System.out.println("The area is: "+area);
```

The radius of the circle was never defined.

Input and Output

All programs from Microsoft Word to Facebook to Overwatch follow this general approach to processing data:

1. Get some data from an input device (such as the keyboard) or from a file.
2. Process that data using a combination of algorithm and secondary storage files.
3. Send the processed result to an output device (such as the monitor) or write the results to a file.

Reading from the Keyboard

- As we mentioned earlier, **System.in** points to your computer's keyboard.
- We still have to create an object which uses the keyboard and this object is called **Scanner**.

Code.

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

This will produce a compiler error because Java doesn't automatically import the Scanner class.

Importing the Scanner class

- The Scanner class is found in the **java.util.Scanner** class library.
  - We must put the statement "import java.util.Scanner;" near the top of our code.
- NetBeans has a tool called "Fix Imports" which will write this line for you.
  - On Windows, this is "Ctrl+Shift+I".
  - On Mac, this is "Cmd+Shift+I".
- We will need to use this tool frequently.

Using the Scanner class.

Once we have the class imported, we can use it. Here's an example. Because your age is a whole number, we must use the `int` type.

```
System.out.println("This program reports your age in five y
System.out.print("What is your age? ");

int age = keyboard.nextInt();
int agePlusFive = age + 5;
System.out.println("In five years, you will be " +
    agePlusFive + " years old.");
```

Program output

When we run this program, this is the output:

```
This program reports your age in five years.
What is your age? 16
In five years, you will be 21 years old.
```

Another example: Text

This program will request your name and say high back to you. Because we are reading in a name (which is text), we use the `String` type.

```
System.out.println("Howdy, camper.");
System.out.print("What is your name? ");

String name = keyboard.nextLine();
System.out.println("Hello, " + name +
    "! We are glad you could be here.");
```

Program Output

```
Howdy, camper.
What is your name? George Washington
Hello, George Washington! We are glad you could be here.
```

Another example: Double

This program will request your GPA and report it back to you. Because it uses a floating point number, we require a `double` data type.

```
System.out.println("Howdy, camper.");
System.out.print("What is your gpa? ");

double gpa = keyboard.nextDouble();
System.out.println("Hello! You have a " +
    gpa + " grade point average.");
```

Program Output

```
Howdy, camper.
What is your gpa? 3.85
Hello! You have a 3.85 grade point average.
```

Fill the rest of the hour.

Write a program which requests, in this order:

- Someone's name.
- Someone's age.
- Someone's GPA

The program then reports this information back to the screen on a single line.