Class Notes 2

eLearning has been updated with new material, including a link to the textbook in PDF format. This version has links in it so if you read it online, it will take you to examples and other parts of the text available from Dr. Eck's website.

The second topic on eLearning is a clickable link to the Class Google Drive.

Topics

Social and Ethical Impacts of Computing
CPU - central processing unit, transistors
Binary, Decimal, Hexadecimal
Natural Language, Machine Language
Data Types, Data Representation, Variables
Mathematical Operators

How a computer computes

Transistors are small devices that can store a single state, like 1 or 0 (expressed in programming as a **bit**).

Eight bits together are a unit of data called a **byte**.

Every concept that we express on a machine can be achieved using a combination of many 1's and 0's.

Translation

Machines take binary and convert them into two kinds of numbers, hexadecimal (base-16 number system) and decimal (which us humans prefer to use).

They also use something called the ASCII character set, an 8-bit expression of a single **character** on the computer system.

Programming Languages

In order to perform this kind of translation, we have a middle ground between natural and machine languages called *programming languages*. These programming languages allow us to convey instructions to a machine while eliminating the ambiguity of natural language.

Abstraction

x = 1

symbols can represent concrete values.

Java

The Java programming language has one major strength: it is extremely portable. When a Java program is compiled into *bytecode*, that bytecode can be shared with any computer and then compiled by that specific machine using the JVM, or Java Virtual Machine, which is installed by default to almost every single computer world-wide.

The drawback to Java programs is that they aren't as efficient as binaries compiled for specific processors. As a result, languages like C and C++ are better choices when processing speed and performance are a concern.

While there are hundreds of programming languages, they all in general share a lot of the same characteristics. Where they differ is in how they implement the different ideas of problem-solving, and the syntax that they use.

How your computers store data

Computers have two kinds of storage available to them.

The first is called **memory**. Memory, traditionally electronic chips with no moving parts, is *volatile* in nature. Information stored in memory is only present while the computer is powered on. The moment that power is lost or the machine is reset, that data is lost. Memory is the "working space" of your computer's processor.

The second is called **storage.** Storage used to include multiple platters of spinning magnetic discs, although they are rapidly being supplanted by solid state devices which are all memory chips with no moving parts. Storage is *persistent* in nature. Even when your computer is turned off, anything that you saved in the storage will be there the next time you turn it on.

Smartphones and tablets blur the line between memory and storage, because they largely function using memory only in order to keep response time very low. However, one can add regular storage to some devices in the form of microSD or SD cards.

--

Variables - a variable is like a *label*. It identifies a unique single piece of information in the computer's memory.

In Java, we can *declare* a variable of a specific type to store information. The eight *primitive data types* are:

integer types
byte - 8 bits
short - 16 bits
int - 32 bits
long - 64 bits

floating point types
float - 32 bits
double - 64 bits

others
boolean - true or false
char - single ASCII character

Beyond the primitive data types are the *abstract data types*. ADT's are more complex and take up more space than primitives, but can express more detailed concepts.

String is an ADT in Java that allows us to store "strings of characters."

Scanner is an ADT that we use to read input from the command line.

Random is an ADT that we will use later to generate pseudorandom numbers for simulations and games.

Mathematical Operators

Note that the order of operations (PEMDAS) still applies to all mathematical operations in our programming language.

+	addition
-	subtraction
*	multiplication
/	floating point division
%	modulus (remainder)

Programming Exercise No. 1

Write a program that prompts the user for variables a, b, and c. Implement and solve the quadratic formula using these variables.

```
x1 = ( -b + Math.sqrt( (b * b) - (4 * a * c ))) / ( 2 * a )
x2 = ( -b - Math.sqrt( (b * b) - (4 * a * c ))) / ( 2 * a )
```

Luckily, Math is a library that is important by default so you technically do not have to import java.lang.Math to use it.

The sqrt() function calculates the square root of whatever you give to it as a parameter.

From time to time, you may get a result that looks like *NaN*. This means "not a number" and can come up if you try to take the square root of a negative number.

Getting User Input

The Scanner ADT allows us to make an object that handles user input from the command line. The different functions it has allow us to accept and store different data types:

`nextInt()`	integers
`nextDouble()`	doubles
`next()`	Strings

Declaring and Instantiating

Declaring a variable means that you specify a variable's type as well as the name.

Instantiating means that you assign a specific value to that variable.

In programming we tend to do both together at the same time, but make sure you remember that they are actually separate things:

```
// separate

int a;
int b;

a = 5;
b = 6;

// together

int a = 5;
int b = 6;
```

Output

`System.out.print()` is used when you want to display a string but do not want to go to the next line after it is displayed.
`System.out.println()` is used when you want to display a string then have the cursor move to the next line (essentially the same as pressing the Enter/Return key on your keyboard).

Coding Conventions

Coding Conventions are the unwritten rules that programmers follow to generate readable code that can be easily shared with others.

For variables in Java, they are:
1. nouns
2. always start with lower case
3. follow camelNotation for multiple word names
4. cannot start with a number

Decisions and Conditions

In order for our programs to be much more useful, we have to find a way for our computer to be able to make *decisions* based on *conditions*. For example, you might be familiar with this system in some homes:

The Nest Thermostat is an example of an artificial intelligence system. The software behaves as an *intelligent agent* to regulate the temperature within a home based on data that it collects as it is being used. That data is processed, analyzed, and used by the system to make **decisions** about when to turn itself on or off, and what specific temperatures it is trying to stay within.

Conditions

Conditions on a computer are expressed using booleans (true or false). We create *boolean expressions* that resolve to either true or false:

```
a < b          a is less than b
a > b          a is greater than b
a <= b         a is less than or equal to b
a >= b         a is greater than or equal to b
a == b         a is equal to b
a != b         a is not equal to b
```

Please note that in boolean expressions we have to use the double equals sign for boolean equality because in Java the single equals sign is already used as the *assignment operator*.

Conditional Control Statements
Decision Statements

Our computers can ask questions that resolve to true or false using **if** and **else.** There are three different ways for a computer to ask questions:

if

```
if( condition )
{
    // code block if true
}
```

The standard if statement checks to see if a condition is true. If it is true, it will run the block of code enclosed in the curly braces. If the condition is false, it does nothing.

if-else

```
if( condition )
{
    // code block if true
}
else
{
    // code block if false
}
```

The if-else is a kind of *branching statement* where if the condition is true, the first code block will execute, and if the condition is false, the second code block will execute. At no time will both of them run, since they are mutually exclusive.

if-else-if

```
if( condition )
{
    // code block if true
}
else if( condition )
{
    // code block if true
}
else if( condition )
{
    // code block if true
}
else if( condition )
{
    // code block if true
```

```
}
.
.
else
{
     // code block if false
}
```
The if-else-if allows us to check multiple conditions sequentially, where *order matters*. Only one block can be executed as a result, and the statement stops after one of the conditions is true and that code block executes.

Programming Exercise No. 2

Theme Park pricing application. Write a program that prompts a user for their age. If they are age 3 or younger, their admission is free. If they are under age 10 but older than 3, they pay a $200 child admission. If they are over age 10, they pay $500 standard admission. If they are 65 or older, they pay $250 senior admission, and they are 100 or older, they get a free super senior admission. Your program should function as follows:

```
Theme Park Prices!
Enter your age: 24
Standard admission of $500

Theme Park Prices!
Enter your age: 3
Free admission!
```

Random Number Generation

For simulations and games, it is very useful to have our computer generate random numbers. However, computers are very logical and concrete, and thus are unable to generate truly random numbers. Using an algorithm, computers can generate what we call *pseudorandom numbers*. They rely on a **seed value**, which is a number on the computer which constantly changes. The most common number used is the System time.

Time on most computers is stored as a very big integer which records the number of seconds that has elapsed since January 1, 1980. This time is referred to as the *epoch*. The Chinese calendar has a different epoch, and is currently at 4716, because their recorded history begins with the written records of the dynasty that began 4716 years ago. Many other civilizations document the date based on a significant event in their past that represents their epoch time.

To generate random numbers we use a library:

```
import java.util.Random;
```

From there, we have to instantiate an object (just like we do Scanner) to use it:

```
Random gen = new Random();
```

We are going to use one specific function in the Random library called `nextInt( int value ).`

```
int randomNum = gen.nextInt( 10 );
```

This function, when passed an integer value, will give us a random integer between 0 and one less than the integer specified. So the line of code above here will generate a random integer from 0 to 9 and store it in the variable randomNum.

To generate a random number in a specific range, we apply the following algorithm:

```
// generating an integer within a specific range
int min = 3;
int max = 7;
int randomNum = gen.nextInt((max-min) + 1 ) + min;
```

Programming Exercise No. 3

Write a program that simulates rolling two six-sided number cubes and displays the total.

Be sure to import java.util.Random, then declare and instantiate the RNG before you use the nextInt() function.

WARNING: DO NOT NAME YOUR JAVA CLASSES AFTER EXISTING JAVA ADT'S LIKE RANDOM OR STRING. IF YOU DO, THEY WILL OVERRIDE THE ONES INCLUDED WITH THE PROGRAM AND THEY WILL NOT WORK AS EXPECTED.