Week 1 - Wednesday

CS 121
Last time

- What did we talk about last time?
- Syllabus
- Computers
Questions?
Back to computers
Hardware vs. software

- Hardware refers to physical parts of the computer
  - Processor
  - Memory
  - Hard disk
  - Monitor

- Software refers to the programs and data that run on it
  - Operating system (Windows, Mac OS, Linux, Unix)
  - Web browser (Internet Explorer, Safari, Firefox, Chrome)
  - Business applications (Word, PowerPoint)
  - Games
Von Neumann Architecture

- Basic layout of all modern computers
CPU

- The “brains” of the computer
- Fetches instructions and data from memory
- Performs computations on the data based on the instructions
- Can send results to I/O
- A modern CPU is made of electronic circuitry embedded in a small silicon chip
How fast are computers?

I typed this PowerPoint on a computer running at 2.6 GHz.

That’s 2,600,000,000 cycles per second.

Each cycle, your computer can do something like:

- Add
- Subtract
- Multiply
- (Usually not divide)
“The density of transistors on a CPU doubles every 18 months”
Historically, this has meant that CPU speeds have doubled every 18 months
We can’t make things much faster because of heat and power
We can still put more “stuff” into a CPU
What do we do with that extra stuff?
Modern laptops and desktops are now almost all **multicore**

Multicore means that each CPU actually has several independent processors called cores inside.

A CPU with 4 cores can actually be computing 4 different things at the same time.

Parallel processing
Parallel processing

- Works well for problems like washing loads of laundry in a laundromat

- But, if you have 3 loads of clothes, there is no way to wash them faster with 4 washers
Parallel limitations

- Parallel processing works very poorly when different processors have to work on the same data and conflicts can happen
- Brain surgery with 100 surgeons is not 20 times faster than brain surgery with 5
- It’s not safer, either
Memory

- Storage for all the data and instructions on your computer
- Modern computers store everything as binary digits (bits) which have a value of 0 or 1.

<table>
<thead>
<tr>
<th>Unit</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>8 bits</td>
</tr>
<tr>
<td>1 kilobyte (kb)</td>
<td>$2^{10}$ bytes</td>
</tr>
<tr>
<td>1 megabyte (mb)</td>
<td>$2^{20}$ bytes</td>
</tr>
<tr>
<td>1 gigabyte (gb)</td>
<td>$2^{30}$ bytes</td>
</tr>
<tr>
<td>1 terabyte (tb)</td>
<td>$2^{40}$ bytes</td>
</tr>
</tbody>
</table>
Cache
- Actually on the CPU
- Fast and expensive

RAM
- Primary memory for a desktop computer
- Pretty fast and relatively expensive

Flash Drive
- Faster than hard drives
- Seen on USB drives but SSDs are becoming common too

Hard Drive
- Secondary memory for a desktop computer
- Slow and cheap

Optical Drive
- Secondary memory that can usually only be written once
- Very slow and very cheap
Input/Output

- **Monitor**
  - Common visual output device

- **Speakers**
  - Common audio output device

- **Mouse**
  - Common input device

- **Keyboard**
  - Common input device
What is computer science?

- Now that we’ve (sort of) defined computers, what is computer science?
- The study of information, computation, and solving problems with programs
- Subfields:
  - Theoretical computer science
  - Programming languages and compilers
  - Operating systems and networking
  - Graphics
  - Numerical computing
  - Information storage, retrieval, and security
  - Architecture and hardware
  - Artificial intelligence and machine learning
What is programming?

- Computers are stupid, but fast
- Programming is the process of giving them very detailed instructions about what to do
- Usually, programming is done in a rigid, formalized language, like Java
- English is insufficient:
  - E.g., “Computer! Solve my relationship problems!”
  - Writing a program to solve your relationship problems in Java would require you to be more detailed and explicit
Layers

- Computer science is built out of layers (like a burrito)
- No one can understand everything
- People tend to focus on a particular level
Layers of Abstraction

User

Application

Operating System

Hardware

We will program here
Writing a Java Program
Writing the program

- Usually, this step takes the majority of the time
- A **Java** program is stored as an ordinary text file
- You can write a **Java** program using any text editor
  - Notepad
  - jEdit
  - UltraEdit
  - Notepad++
- We'll be editing, compiling, and executing all with Eclipse
Every language has syntax
Syntax is the set of rules that govern how you can make meaningful statements
In English, the basic syntax for a sentence says that you must have:
- Subject
- Predicate
Just like English, **Java** has many, many different rules you will need to learn
In Java, one of the most basic rules of syntax is that everything must be inside a class.

For now, just think of a class as a way to organize things.

We are going to create a Java program called **Hello**

We must create a file called **Hello.java** and put a class called **Hello** inside of it.
Hello.java

- Ignore the keyword `public` for now
- The keyword `class` announces that a new class is about to be named
- `Hello` is the name of the class
- The braces mark the beginning and end of the contents of class `Hello`

```java
public class Hello {
}
```
The previous empty class will compile, but it will not run.
We need to give our class a starting point.
The starting point for any Java program is a `main()` method.

```java
public class Hello {
    public static void main(String[] args) {
    }
}
```
The previous program will run, but it won’t do anything
Now, we add a print statement to our `main()`

```java
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}
```
Recap

- The full Hello World program
- Remember that everything is in a class
- The class name must match the file name (Hello.java)
- The `main()` method is where the program starts
- The print statement outputs information on the screen

```java
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}
```
Basic output is done with `System.out.println()`

You put what you want to print out inside the parentheses

You can print:

- Any text enclosed in quotes:
  
  ```java
  System.out.println("43 eggplants");
  ```

- Any number:

  ```java
  System.out.println(3.14159);
  ```

You can use `System.out.print()` instead if you don’t want a newline
Eclipse Tutorial
More Java Syntax
Semicolons

- In **Java**, like C, C++, and many other languages, we separate different statements with a semicolon (`;`).
- If we want to do a number of statements, we just type them in order, with a semicolon after each one.
Sequencing

- For example, instead of one print statement, we can have several:

```java
System.out.println("Hello, world!");
System.out.println("Hello, galaxy!");
System.out.println("Goodbye, world!");
```

- Each statement is an instruction to the computer
- They are printed in order, one by one
Java is a case sensitive language
Class is not the same as class
System.out.println("Word!");
prints correctly
system.Out.Println("Word!");
does not compile
**Whitespace**

- **Java** generally ignores whitespace (tabs, newlines, and spaces)

```
System.out.println("Hello, world!");
```

is the same as:

```
System.out.println("Hello, world!");
```

- You should use whitespace effectively to make your code readable
Comments

- Programs can be confusing
- Sometimes you want to leave notes for yourself or anyone else who is reading your code
- The standard way to do this is by using comments
- Although comments appear in the code, they do not affect the final program
Comments

- There are two kinds of comments (actually 3)
- Single line comments use //

```java
System.out.println("Hi!"); // this is a comment
```

- Multi-line comments start with a /* and end with a */

```java
System.out.println("Hi!"); /* this is a multi-line comment */
```
Java is a large, complex language
Even so, there are only a few tasks that you can ask it to do
You have already learned:
- Sequencing
- Basic output
Where we are headed

- There are not that many other things you can tell Java to do
  1. Storing numbers and text
  2. Basic mathematical operations
  3. Choosing between several options
  4. Doing a task repetitively
  5. Storing lists of things
  6. More complicated input and output
  7. Naming a task so that you can use it over and over again

- That’s basically it
Upcoming
Next time...

- We will finish talking about computers and software development
- Lab 1
Read Chapter 2 of the textbook