Introduction to Programming Concepts

Ryan Nixon

History of Programming

- Programming has ties all the way back to the 18th century and has undergone many transformations
- Here is a very brief history of computing, summarized through the people that contributed to the advancements

The Difference Engine

- Charles Babbage (1791-1871)
  - Making a difference...engine. Then designs for the analytical engine.
  - Developed the first "computer", more a simple calculator but signaled the beginnings of computing
- Ada Lovelace (1815-1852)
  - "The Enchantress of Numbers"
  - Collaborated with Babbage on his designs
  - Considered the first Programmer

The Founding of Computer Science

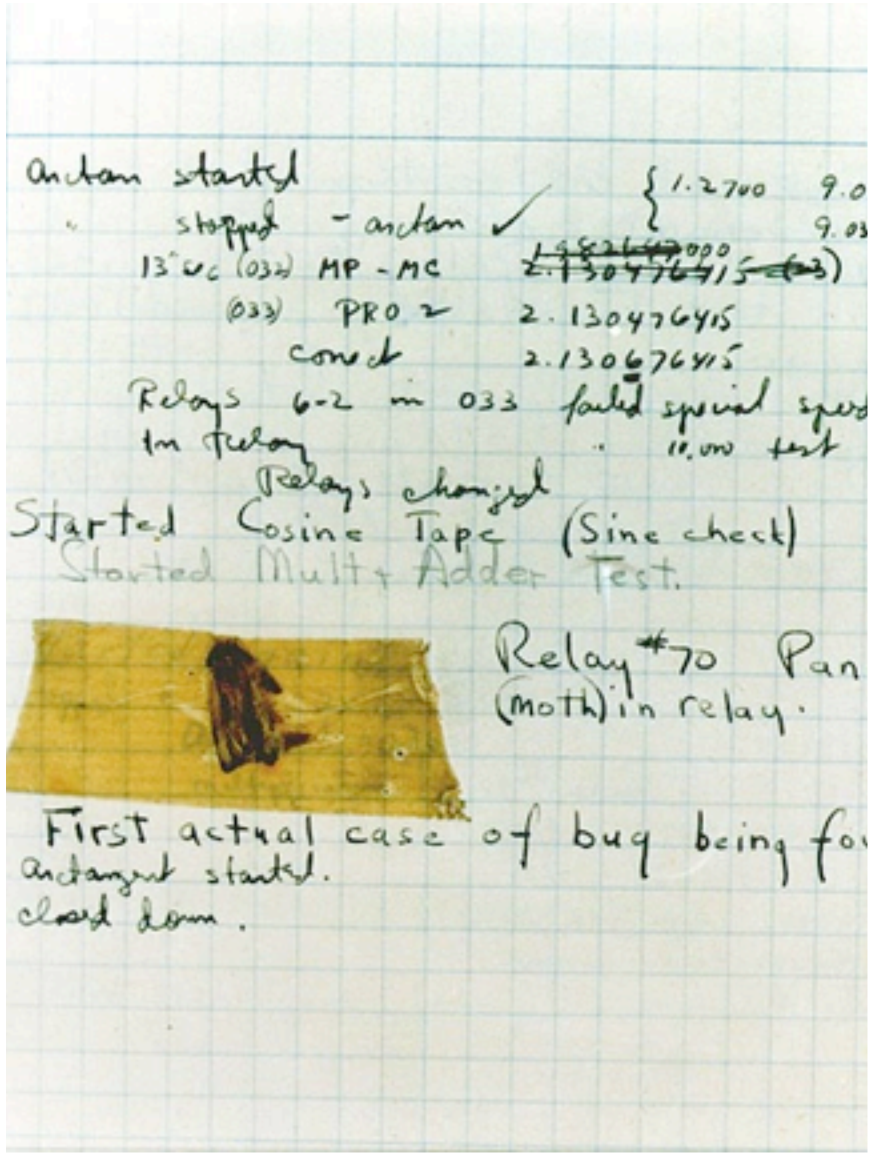
- Alan Turing (1912-1954)
  - Developed the ideas for computation & algorithms
  - Created the “Turing Machine”, a hypothetical device capable of executing any program
  - Advanced artificial intelligence forward with the “Turing Test”, a proctored exam between a machine and human to determine believability of the program

Modern Day Computers

- John von Neumann (1903-1957)
- Pioneered the "Von Neumann" model, a modern day computing architecture wherein the processing, arithmetic operations, memory and storage were separate
- This model allowed for machines to be built that stored programs on disk, rather than having to be redesigned for each task

Bugs

- Grace Hopper (1947)
- Removed a moth from one of the first computers (the Mark II), called it a “bug”
- Hopper then popularized the term, resulting in modern-day computer programs having bugs when something goes wrong in their programming
- “Debugging” would be the action of sleuthing out and removing these bugs



```
arctan started                                  { 1.2700     9.0
  "      stopped  - arctan ✓                    {            9.03
     13"00 (033)  MP - MC          ~~1.982147000~~
                                   ~~2.130476415~~ (-3))
           (033)  PRO 2             2.130476415
                  correct           2.130676415
        Relays 6-2 in 033 failed special spe
        in relay                  .   10,000 test
              Relays changed
Started Cosine Tape (Sine check)
Started Mult + Adder Test.

                               Relay #70 Pan
                               (moth) in relay.

First actual case of bug being fo
arctangent started.
closed down.
```

Personal Computers

- Steve Jobs & Steve Wozniak, Apple Computer (1976)
- Moved the computer era from mainframes and business machines to computers in every home
- Greatly expanded the software market, creating an explosion of programmer jobs

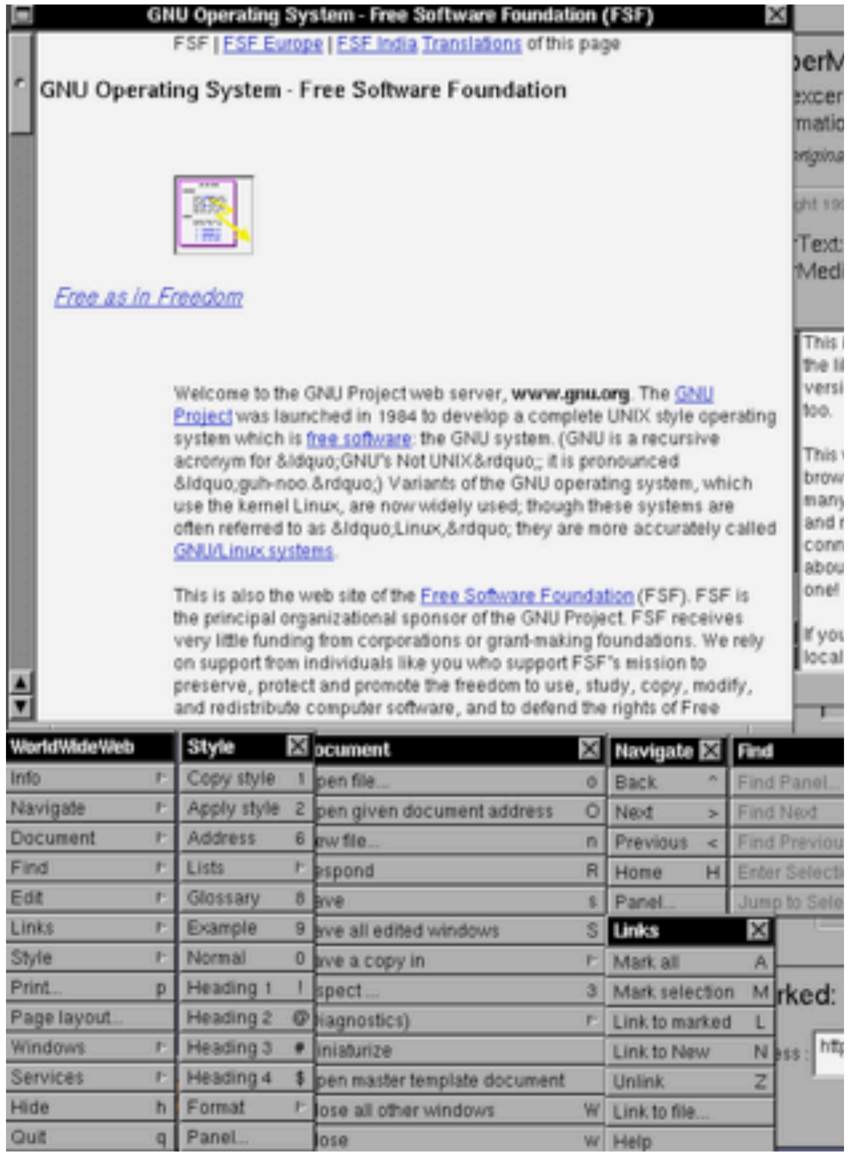
Modern Operating Systems

- Bill Gates & Paul Allen, Microsoft (1975)
- Developed the Windows platform and freely licensed it to PC manufacturers
- The first real software powerhouse
- Continues to advance the software industry 30 years later with the .NET platform

MICROSOFT®
WINDOWS™

World Wide Web

- Tim Berners-Lee (1990)
- Created the WorldWideWeb browser and HTTPd web server which spawned the world wide web that we now take for granted
- Single-handledly created the web development industry that uses web servers and the HTTP protocol rather than building specifically against the computer


GNU Operating System - Free Software Foundation (FSF)

FSF | FSF Europe | FSF India Translations of this page

GNU Operating System - Free Software Foundation

Free as in Freedom

Welcome to the GNU Project web server, **www.gnu.org**. The GNU Project was launched in 1984 to develop a complete UNIX style operating system which is free software: the GNU system. (GNU is a recursive acronym for &ldquo;GNU's Not UNIX&rdquo;; it is pronounced &ldquo;guh-noo.&rdquo;) Variants of the GNU operating system, which use the kernel Linux, are now widely used; though these systems are often referred to as &ldquo;Linux,&rdquo; they are more accurately called GNU/Linux systems.

This is also the web site of the Free Software Foundation (FSF). FSF is the principal organizational sponsor of the GNU Project. FSF receives very little funding from corporations or grant-making foundations. We rely on support from individuals like you who support FSF's mission to preserve, protect and promote the freedom to use, study, copy, modify, and redistribute computer software, and to defend the rights of Free

The Virtual Machine

- James Gosling, Sun Microsystems (1991)
- Developed the Java programming language as a “write-once, run anywhere” system
- Created the concept of the Java Virtual Machine, an abstract machine that sits between the program and the real hardware, to achieve this mobility
- Led to the Microsoft .NET platform which also runs on a virtual machine

Programmers

Q: How many programmers does it take to plug in a light bulb?

Programmers

A: Sorry, that’s a hardware problem.

Programmers

- Programmers use a machine language to instruct a computer what to do. This may include writing a program that tells another program what to do
- They need to be aware of how a machine works and how to work with peripherals, but the magic happens only while the machine is running
- Note that programmers come in all types, scripters, web developers, kernel junkies. To each your own

Machine Language

- The real programming language, used by computers to process instructions and to communicate with other computers
- Consists of binary numbers - a base 2 numbering system used by computers
- Bit - 0 or 1. This may also be considered on/off, true/false, yes/no, etc.
- Byte - 8 bits (01100100)
- These machine languages are specific to the architecture that is in use. AMD, Intel, Atom, etc. all have different instruction sets
- Near impossible to program. Programming languages are developed to assist with this

Low Level Languages

- Symbolic languages that directly represent instructions on the machine

-
  ```
  LW   $t0, 4($gp)
  ADD  $t1, $t0,  3
  MULT $t1, $t1,  $t0
  SW   $t1, 0($gp)
  ```
- Programmers still need to know how to work with registers, the specifics of the processor implementations, and each vendor-specific instruction set
- Assembly (ASM) is the most common low-level language with many architecture-specific derivatives

High Level Languages

- Uses plain English to represent large groups of instructions

-
  ```
  int t0;
  cin >> t0;
  cout << t0 * (t0 + 3);
  ```
- Enables programmers to organize code in much more understandable ways
- Cannot be run directly on the machine. High level languages use compiler programs to convert understandable code to lower levels
- Interpreters are a type of compiler that compiles chunks of code as they are encountered while running. This is known as Just In Time (JIT) compiling

High Level Languages

- A third type of compiler does not compile to the machine language but to an intermediate language, (sometimes known as bytecode)
- Java uses this type. Java code is compiled to bytecode to be executed on the Java Virtual Machine (JVM)
- This allows Java code to be compiled once and run anywhere, on anything...phones, websites (using Applets), even your microwave if it has the JVM

High Level Languages

- Compiled languages: C, C++, BASIC
- Interpreted JIT languages: Python, PHP, Ruby
- Bytecode languages: Java, C# (.NET)

Programming Paradigms

- **Functional** - A language that emphasizes consistency between executions. The same input should always produce the same output
- **Imperative/Procedural** - A collection of variables and functions (not the same as Functional) which runs in a top-down manner from beginning to end
- **Object-oriented** - Mimics real life, using objects that contain their own attributes and methods and communicate with other objects
- **Logic** - Flips the programming model on its head. Programmers define the solution constraints, and the program attempts to find all possible outputs for the input within those constraints

General Programming Process

- Design, Mock up, Pseudocode
- Write code, iterate design as needs arise
- Compile, test, debug, fix
- Refactor remaining code if time allows
- Documentation & training

The Sad News

- About 30% of programming involves actual coding. That's the best part! (Well, for me)
- The rest is interface/algorithm design, testing, debugging, documentation, meetings with fellow developers, etc.

Your First Program

```
class Hello {

    public static void main(String[] args) {

        System.out.println("Hello World");

    }

}
```

Your First Program

- `class Hello` - The name of the “class” containing your program logic. The name of the file (Hello.java) is often the same name as the class
- `{ }` - Braces contain application logic. They are used to define “scope” which will be covered later. For now think of them as the beginning and end of what you are writing

Your First Program

- `public static void main(String[] args)`
- This line is written in *all* Java programs. It defines the entry point to the program, the first code to be executed.
- `public` - Accessible by other parts of a program
- `static` - A class-level method. More on this later
- `void` - Nothing is 'returned' by this method. If a + b = c, then c would be what is normally returned.
- `main` (Case-sensitive) - The name of the method
- `String[] args` - The first parameter to the program. Often used when executing a program from the command-prompt

Your First Program

- `System.out.println(“Hello World”);`
- `System.out.println` - The name of *another* method stored in the System package
- `( )` - Placing parentheses after a method name will invoke that method
- `“Hello World”` - A string (sequence of characters) being sent as the first argument to the println method.
- `;` - All commands in Java end with a semicolon. Note that {} brackets are excluded from this since they are not commands

Your First Program

- In the end, the Hello World program should print the words “Hello World” to the screen
- This program is particularly famous. It is a common practice for old and new programmers to try creating a “Hello World” program when introduced to a new language
- Of course since Java is a high-level language, it must be compiled before we can run it

Compiling A Java Program

1. Open a terminal or command prompt. On the classroom machines you can use the Windows Powershell

2. Navigate to the directory of your stored "`Hello.java`" file using cd and ls/dir

3. Run the Java compiler on the file using "`javac Hello.java`" This will generate a compiled class file, "Hello.class"

4. Run this class file using "`java Hello`" (note the missing file extension)

5. You should see the words "Hello World" output to the command prompt

Compiling A Java Program

- When we get to the point of working with larger programs, this command-line compiling method will become a burden
- You are expected to know how to compile from the command-line, but soon I we will switch to an Integrated Development Environment (IDE) to ease this compilation process
- If you prefer, you may use any Java IDE that you like provided that you ask me first. The class will be taught using NetBeans but Eclipse is another very popular IDE

Questions?

- Please remember to look over the course syllabus and schedule
- A link will be on Blackboard as soon as I have access but in the meantime you may use: http://uaa.ryannixon.com/cs201