CS221
C and Systems Programming

Whirlwind introduction to C

C and Java have a lot in common

- C is the "parent" of many programming languages: Java, C++, C#, Objective-C, Rust, Go, …

Big difference #1: all code is not inside a class. There are no classes

- You just declare methods right in your C source file.

Big difference #2: Compiled C code is native machine language—no virtual machine

- Compiled C programs are not device-independent

Commonalities between C and Java

Curly braces

Variable declarations

Variable types: `char, short, int, long, float, double`

- No `byte`, no `boolean`
- `char` is just 1 byte long

Methods and method declarations

for(), while(), do-while()

if(), else if (), else

switch(), ternary operator

Commonalities between C and Java…

Arithmetic and Boolean operators

Variable scope rules

Casts

Comments

Ok, what are the differences?

`main()`

- Returns `int`, not `void`
- Has two inputs: `argc`, `argv`

No `new` operator

- We will talk about C's alternative in a few days
- Arrays are declared differently

Use `printf()` rather than `System.out.println()`

No `String` class

- C "strings" are arrays of `chars`, terminated with a 0 value

During course, we will talk about several more differences

Arrays in C

```
int idValues[40];
double scores[ idValues[0] ];
char thisIsAString[256];

char myName[13] = "Paul Haskell";
```

printf()

```
printf("Formatting string", var1, var2, …);
```

- %d: decimal integer
- %f: float
- %lf: double ("long float")
- %ld: long decimal integer
- %s: "string"
- %c: char

- \n newline
- \t tab
- \\ back-slash character
- %% percent character

sprintf()

```
char buf[256];
int var1;
double var2;
sprintf(buf, "Formatting string %d %lf", var1, var2);
```

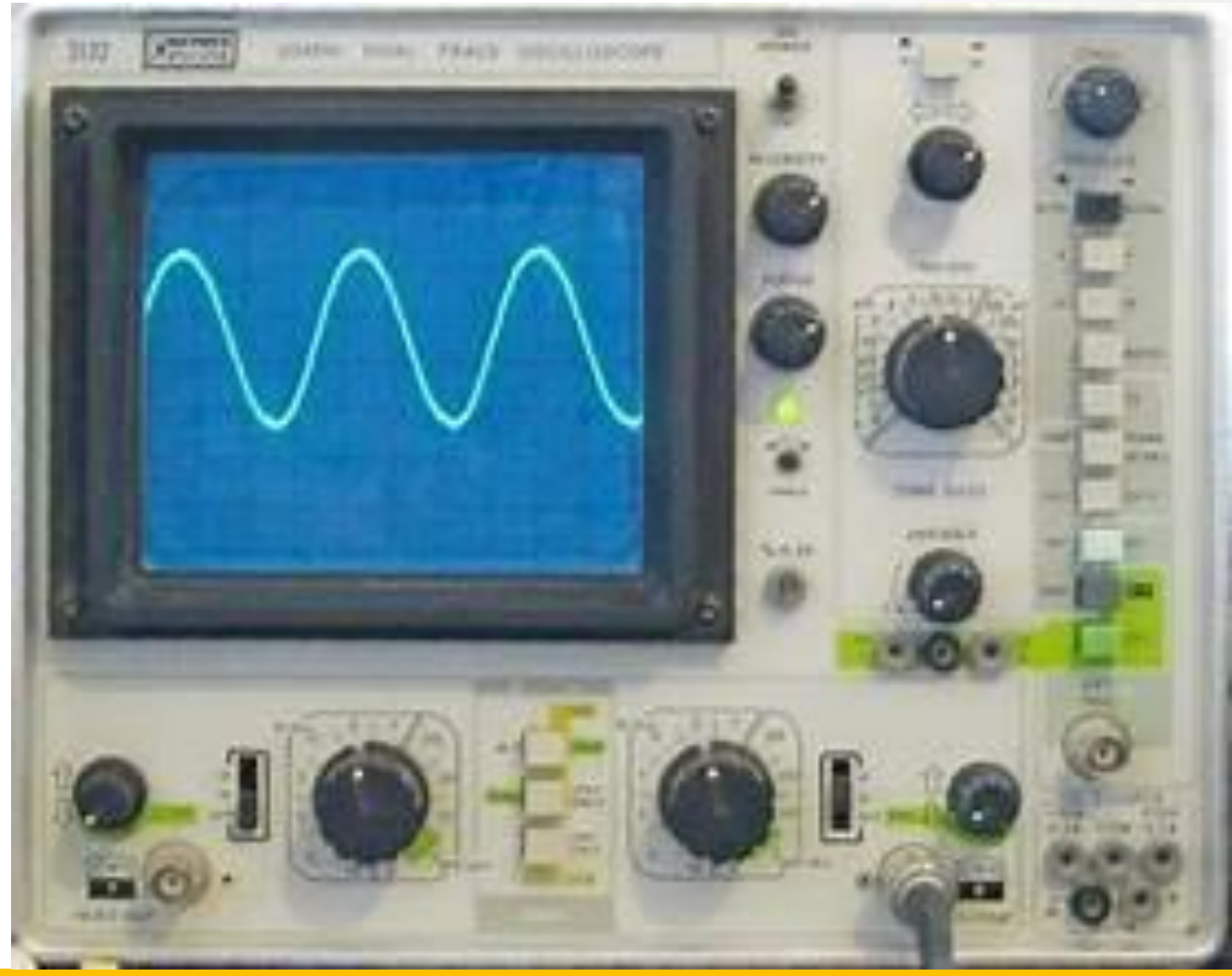
Why not use Java for everything?

C gives us tools that the Java designers thought were too dangerous…

…and sometimes we need those tools!

One more C program…

demo3.c



Tools for the class

gcc and 'make'

gcc - C compiler

make - program building assistant

- Talk about this next time

Libraries and #include files

What are they for?

Some common ones

- **math.h** **-lm** Mathematics operations
- **stdio.h** printing and input methods
- **stdlib.h** convert numbers to/from strings, random numbers, exit(), sort()
- **unistd.h** File I/O, usleep(), get/set system info

- We will work with several more this semester

Four stages of compilation

Preprocessing

- Handle #include files, macros (#define, #ifdef)
- Just text substitution
- `gcc -E`

Four stages of compilation

Preprocessing

Compilation

- Convert C code to assembly language
- `gcc -S`
- CS315 talks a lot more about assembly language

Four stages of compilation

Preprocessing

Compilation

Assembly

- Convert human-readable assembly language to machine code
- This is computer-platform-specific
- `gcc -c`

Four stages of compilation

Preprocessing

Compilation

Assembly

Linking

- Combine "object files" and "libraries" into an executable program
- `gcc`
- CS414 covers how compilers work and how to make them

gcc command-line options

Where does gcc output go?

-o	**o**utput file
-I	look for #**i**nclude files here
-L	look for **l**ibraries here
-l	include this **l**ibrary
-g	include debu**g** info
-O	**o**ptimize the compilation

Lab02