C Programming

What is C?

- a general purpose, compiled, procedural computer language
  - structured programming constructs
    - loops, conditional statements
  - static type system
    - variable types are known at compile time
  - recursion
- often referred to as a high level language that is as "close to the machine" as possible while still being architecture independent

A very brief history of C

- for the definitive history of C see
  - https://www.bell-labs.com/usr/dmr/www/chist.html

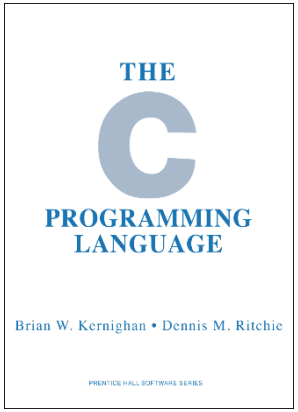
1968

- "[Ken] Thompson wanted to create a comfortable computing environment constructed according to his own design, using whatever means were available."
- "Thompson was faced with a hardware environment cramped and spartan even for the time: the DEC PDP-7 on which he started in 1968 was a machine with 8K 18-bit words of memory and no software useful to him. While wanting to use a higher-level language, he wrote the original Unix system in PDP-7 assembler."

1968

1969-1978

- 1969 Thompson creates the B language; invents the ++ and -- operators
- 1972 Dennis Ritchie and Thompson write a Unix kernel in early C
- 1973 Ritchie and Thompson complete the essentials of modern C; preprocessor introduced
- 1978 Ritchie and Thompson publish *The C Programming Language*



1989-present

- 1989/90 C language standard adopted as ANSI C and ISO C
- 1999 C99
- 2011 C11
- 2017 C17: Corrections and clarifications of C11

Where is C used?

- most OS kernels
- embedded systems
- many interpreted languages are implemented in C
  - Python, Perl, Ruby, PHP
- many programming languages can call C functions
- computer graphics applications
- computationally intensive applications
- and many more
- ranked #1 or #2 on TIOBE index since 2001
  - https://www.tiobe.com/tiobe-index/c/

C development tools

- at a minimum, you need a C compiler
- Windows (WSL)
  - install gcc
    - https://code.visualstudio.com/docs/cpp/config-wsl
- Mac
  - install Command Line Tools for Xcode
    - `xcode-select --install`
  - or install Homebrew (which installs the above)
    - https://brew.sh/
  - or install Xcode from App store (~40GB of disk space)

C development tools

- you also will want a C-aware text editor
- Visual Studio Code is freely available on Windows, Mac, and Linux
  - https://code.visualstudio.com/docs/cpp/config-clang-mac
- so is eclipse
  - but not an option for WSL

Documentation

- https://devdocs.io/
- https://en.cppreference.com/w/c/language

Fun stuff

- https://www.ioccc.org/

Structure of a simple C program

- a simple C program is made up of:
  - one or more C source code files
    - plain text
    - extension **`.c`**
  - zero or more C header files
    - plain text
    - extension **`.h`**
- to run the program, a **`main`** function is required in one of the C source code files

One version of Hello, world!

```c
#include <stdio.h>

/* hello1.c */

int main(void) {
    puts("Hello, world!");
    return 0;
}
```

Another version of Hello, world!

```c
#include <stdio.h>

/* hello2.c */

int main(void) {
    printf("%s\n", "Hello, world!");
    return 0;
}
```

Compiling a C source code file

- compiling is the process of transforming code written in one language to another language
- when people talk about compiling C code, they usually mean transforming C code into a language that the computer can run directly (machine code)
- a compiler is a program that compiles

C compilers

- there are many C compilers to choose from but students in CISC220 will probably use one of:
  - gcc (Linux)
  - Clang/LLVM (Mac)

Compile our programs like so:

```
gcc hello1.c -o hello1
gcc hello2.c -o hello2
```

Run our programs like so:

```
./hello1
./hello2
```

Preprocessor directive

- **`#include <stdio.h>`** is a preprocessor directive
  - the C preprocessor processes C source code files before the compiler transforms the code into machine code
    - the preprocessor can be a separate program from the compiler, or it can be part of the compiler itself
- an **`include`** directive tells the preprocessor to insert the contents of a file into the C source code file
  - we need to include **`stdio.h`** in our program because it contains the function declarations for **`puts`** and **`printf`**
  - you need to include the declarations for any library functions that you use in your program

Comments

- prior to C99, comments in C used **/*** and ***/** to delimit a comment
  - exactly the same as in Java
- from C99 onwards, line ending comments starting with **//** are also allowed

The `main` function

- every C program coded to run in a hosted execution environment contains the definition (not the prototype) of a function called **`main`**, which is the designated start of the program
  - hosted execution environment means that the full C standard library is supported
  - a freestanding implementation need only support a limited subset of the C standard library
    - typical examples are embedded systems

The **`main`** function

- main has two common allowable forms

```
int main(void) { body }

int main(int argc, char* argv[]) { body }
```

- C99 also allows implementation defined **`main`** functions
  - these are compiler dependent

One version of Hello, world!

```c
#include <stdio.h>

/* hello1.c */

int main(void) {
    puts("Hello, world!");
    return 0;
}
```

returns an int, has no parameters

One version of Hello, world!

```c
#include <stdio.h>

/* hello1.c */

int main(void) {
    puts("Hello, world!");
    return 0;
}
```

Writes every character from a null-terminated string and one additional newline character '\n' to the output stream stdout

Another version of Hello, world!

```c
#include <stdio.h>

/* hello2.c */

int main(void) {
    printf("%s\n", "Hello, world!");
    return 0;
}
```

Formatted print. Loads the data from the given locations, converts them to character string equivalents, and writes the result to stdout.

One version of Hello, world!

```c
#include <stdio.h>

/* hello1.c */

int main(void) {
    puts("Hello, world!");
    return 0;
}
```

Returns 0. In a main method, the return value defines the exit status of the program.