C and Linux Programming
Eastern Washington University
Computer Science
March 30th – June 12th, 2020

Lecture 1

C and Linux Background

Joe Dumoulin

Mar 30, 2020

Eastern Washington University

Motivation and History

Motivation

What is C?

C was initially developed by Dennis Ritchie between 1969 and 1973 at AT&T Bell Lab.

Unix OS is written in C.

Many other languages borrow from C,

Including C#, D, Java, JavaScript, Limbo, LPC, Objective-C, Perl, PHP, Python, Verilog (hardware description language) and Unix's C shell.

Motivation

Why C?

- C is cross-platform.
  - A standards-compliant and portably written C program can be compiled for a very wide variety of computer platforms. from embedded microcontrollers to supercomputers.
- C is versatile.
  - Suitable for embedded systems, device drivers, OS kernels, small command-line utilities, large desktop applications, DBMS's, implementing other programming languages, and so on.

Motivation

Why C?

- C is fast.
  - Widely used to develop systems that demand performance.
  - OS, embedded and real-time systems, communication systems.
  - Most C implementations compile directly to machine code, and the programmer has full power over what happens at the machine level.
  - Even part of Java Virtual Machine is implemented in C/C++.
  - C remains very popular after 40+ years:
  - pypl
  - TIOBE

Motivation

What is Linux? Is it UNIX? No.

"Unix (trademarked as UNIX) is a family of multitasking, multiuser computer operating systems that derive from the original AT&T Unix, development starting in the 1970s at the Bell Labs research center by Ken Thompson, Dennis Ritchie, and others." - Wikipedia

Motivation

What is Linux? Is it UNIX? No.

It's UNIX-like!

"Linux is a family of open source Unix-like operating systems based on the Linux kernel,[11] an operating system kernel first released on September 17, 1991, by Linus Torvalds. Linux is typically packaged in a Linux distribution." - Wikipedia

Sometimes you will see *NIX

Motivation

Why Linux?

- Linux is used to run some of the largest and most popular services in the world.
- Linux is open source, usually free and easy to install and administer in a secure way.

C and Linux

Why C and Linux?

- Unix and Linux (like most modern operating systems) are written in C.
- Most services that work with the operating system are written in C.

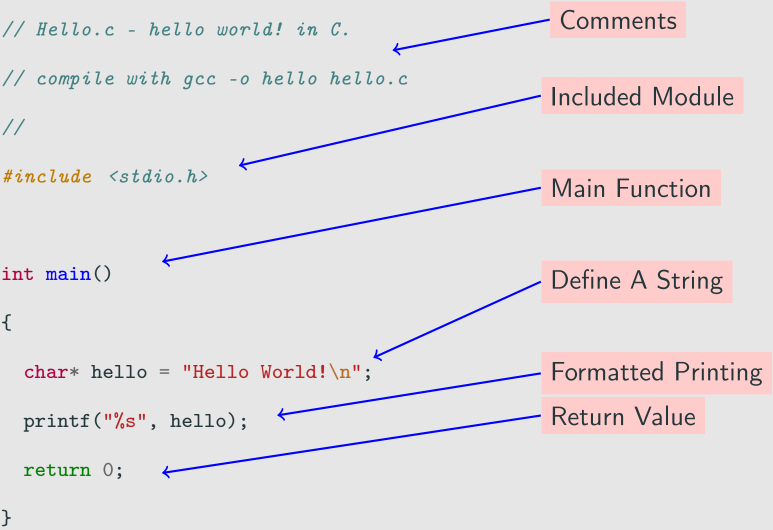
The C Programming Language

C Programs

- C is an imperative (procedural) language.
    - Programmers provide steps about how to solve a problem.
    - Programming task is broken into a collection of variables, data structures, and subroutines (functions).
- Whereas, in OO programming like Java,
    - We break down a programming task into objects that expose behavior and data using interfaces.

Hello World!

Hello World! in C



```
// Hello.c - hello world! in C.
// compile with gcc -o hello hello.c
//
#include <stdio.h>

int main()
{
  char* hello = "Hello World!\n";
  printf("%s", hello);
  return 0;
}
```

Comments

Included Module

Main Function

Define A String

Formatted Printing

Return Value

C is procedural

C programs are mostly composed of function calls

C is programming with procedures or functions

```
// pseudocode to illustrate a typical simple c main function

int main()

{

        int counter;

        counter = get_value();

        use_value(counter);


        printf("%d\n", counter);

        exit(EXIT_SUCCESS);

}
```

From Code to Program

The following steps are needed to create a C program:

- Edit
- Compile
- Link
- Load
- Run!

Edit a C Program file

The human readable program is a text file.

There are many editors that can be used:

- vim
- nano
- emacs
- graphical editors (vscode, sublime, eclipse, etc.)

Open the editor, write or edit the program, save.

Compile your Program

Once you have written the program, you compile it to object code. This is sometimes called translation, as in translating C source code to object code (direct commands to the CPU and other devices). In this class, you use **gcc** to compile your programs.

```
>gcc -c hello.c
>ls
hello.c  hello.o
```

C code -> assembly -> object (machine) code

Compile your Program

What is an object file? - Addresses

```
>objdump -s hello.o

hello.o:     file format elf64-x86-64

Contents of section .text:
 0000 554889e5 4883ec10 488d0500 00000048  UH..H...H......H
 0010 8945f848 8b45f848 89c6488d 3d000000  .E.H.E.H..H.=...
 0020 00b80000 0000e800 000000b8 00000000  ................
 0030 c9c3                                 ..
Contents of section .rodata:
 0000 48656c6c 6f20576f 726c6421 0a002573  Hello World!..%s
 0010 00                                   .
Contents of section .comment:
 0000 00474343 3a202855 62756e74 7520372e  .GCC: (Ubuntu 7.
 0010 342e302d 31756275 6e747531 7e31382e  4.0-1ubuntu1~18.
 0020 30342e31 2920372e 342e3000           04.1) 7.4.0.
Contents of section .eh_frame:
 0000 14000000 00000000 017a5200 01781001  .........zR..x..
 0010 1b0c0708 90010000 1c000000 1c000000  ................
 0020 00000000 32000000 00410e10 8602430d  ....2....A....C.
 0030 066d0c07 08000000                    .m......
```

Compile your Program

What is an object file? - Machine Code

```
>objdump -d hello.o
hello.o:     file format elf64-x86-64
Disassembly of section .text:

0000000000000000 <main>:
   0:         55                        push   %rbp
   1:         48 89 e5                  mov    %rsp,%rbp
   4:         48 83 ec 10               sub    $0x10,%rsp
   8:         48 8d 05 00 00 00 00      lea    0x0(%rip),%rax  # f <main+0xf>
   f:         48 89 45 f8               mov    %rax,-0x8(%rbp)
  13:         48 8b 45 f8               mov    -0x8(%rbp),%rax
  17:         48 89 c6                  mov    %rax,%rsi
  1a:         48 8d 3d 00 00 00 00      lea    0x0(%rip),%rdi  # 21 <main+0x2
  21:         b8 00 00 00 00            mov    $0x0,%eax
  26:         e8 00 00 00 00            callq  2b <main+0x2b>
  2b:         b8 00 00 00 00            mov    $0x0,%eax
  30:         c9                        leaveq
  31:         c3                        retq
```

Linking your Program

compiling and linking is usually a single step

```
# in the old days:
ld /usr/lib/crt0.o -o hello -lc

# but the modern way is easier: compile and link in one step!
gcc -o hello hello.c
```

Loading your Program

To Load a run the program, type a dot and a slash and the name of the program.

```
>./hello
Hello World!
```

Lab 1

- Install Ubuntu Linux 18.04 Desktop
- Install gcc and build essentials
- Install git
- Become familiar with important command line programs
- Download the hello.c code
- Compile and run hello.c