Why C?

There are dozens of programming languages. Why C?

C is “high level” — but not very.

- C provides functions, structured programming, complex data types, and many other powerful abstractions
- ...yet it also exposes many architectural details

Most operating system kernels are written in C.

Many runtimes and virtual machines are written in C.

C influences many other languages.
Effective C programming requires that you master the machine. You must be aware of its architecture and details of operation. We will be using C in Linux on x86-64. The dialect of C we will use is C99. The compiler we will use is gcc.

---

1K&R describes ANSI C (C89), but we will discuss the differences when important.
Learning C

We will not cover all details of C syntax.

We will cover key ideas and particularly important syntax.

You should consult:

- The C Programming Language (K&R)
- Unix man pages
- Given code
A Dedicated Computer

The POSIX platform provides a particular model.

That model is that each process has its own dedicated machine.

That isn’t strictly true, but it is approximated.

Each process appears to have:

- A dedicated CPU
- Private, dedicated memory
- Private input and output facilities
The C language also provides a particular machine model. The CPU manipulates data stored in memory. Data in memory is stored at accessible addresses. Program code is executed as a series of instructions:

- Also stored in memory
- Though possibly not accessible
Programs as Instructions

C programs are translated into machine instructions.

The computer executes these instructions in order.

We can instruct it to jump to a different instruction.

Instructions are things like:

- Add two numbers together
- Compare a number to zero
- Store a number to a location in memory

As we will see, it’s all numbers.
Administrivia

On attendance:
- Attendance to every lecture is mandatory
- Attendance to every lab is mandatory

On AI:
- The AI quiz is due by next Friday
- Don’t use online resources for your assignments
- Don’t talk details with your classmates

On Lab 01:
- Lab 01 is now due Saturday, February 9, at 11:59 PM
C Overview

As previously mentioned, C is a **high level language**.

It has the following characteristics:

- It is **very small**.
- It has **little abstraction**:
  - no generics, no objects, no polymorphism, *etc*.
- It is **unforgiving**.
- It does **exactly what you ask** of it.

There is **no success** in writing C code you do not understand!
Imperative Programming

C is an imperative language.

It consists of statements.

Statements are instructions to the computer to do something.

Statements can be grouped into procedures.

In C, procedures are called functions.
main()

Every C program starts with the function `main()`.

```c
int main() {
    return 0;
}
```

C functions take zero or more arguments and return a single value.

All arguments are pass-by-value, which means they are copies of whatever is passed to them.

\(^2\)Sort of ...
Returning from `main()`

All Unix programs return a value.

That value is often an integer indicator of success.

Convention is that a zero return value is success.

The value returned from `main` is this program return value!

You can view the return value of the most recently returned program from the Unix shell with the command `echo $?`. 
Program Arguments

The `main()` function takes two arguments:
```
int main(int argc, char *argv[])
```

The first is an `integer` containing the number of arguments passed to the program `on the command line`.

The second is the program arguments as an `array of strings`. `argv[0]` is always the program name.

We will discuss strings and arrays more later.
Compiling the Example

Assume that this code is in trivial.c:

```c
int main() {
    return 0;
}
```

We can compile it into an executable as follows:\(^3\)

gcc trivial.c

This will produce the file a.out, which is a native binary.

\(^3\)K&R uses cc, which will also work.
Compiler Options

There are several compiler options you will probably want to use:

```bash
gcc -o trivial -Wall -Werror -O \ 
   -std=c99 -D_DEFAULT_SOURCE trivial.c
```

- `-o <filename>`: Create the file `<filename>` instead of `a.out`
- `-Wall`: Warn for all potential errors
- `-Werror`: Treat all warnings as errors and abort compilation
- `-O`: Enable easy optimizations
- `-std=c99 -D_DEFAULT_SOURCE`: Use ISO C99
Developing Hello World

“Hello World” is a classic first program when learning a language.

We will develop a Hello World together.
Summary

- C is a **high level language used in systems programming**.
- **Architectural details** are important in C.
- The C/POSIX model is:
  - A **dedicated machine** for each program
  - Sequential execution of program instructions
  - Data is stored in accessible, **addressed memory**
- We explored some trivial C programs.
Next Time …

- Variables
- Strings
- Looping
Optional Readings

License

Copyright 2019 Ethan Blanton, All Rights Reserved.

Reproduction of this material without written consent of the author is prohibited.

To retrieve a copy of this material, or related materials, see https://www.cse.buffalo.edu/~eblanton/.