

CSCI 334:
Principles of Programming Languages

Lecture 17: C

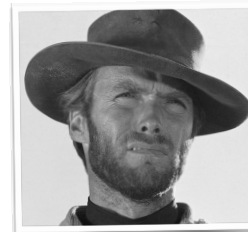
Instructor: Dan Barowy
Williams

Announcements

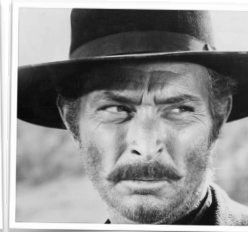
HW8 via email later today

C

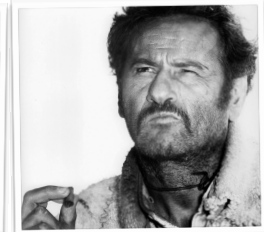
Why am I talking about C now?



"the good"
LISP



"the bad"
C



"the ugly"
C++

C



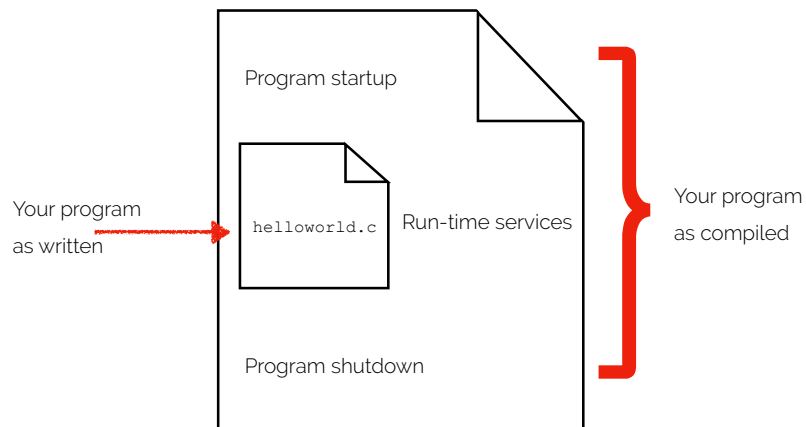
- Invented by Dennis Ritchie (seated) and Ken Thompson in 1969
- Intended to allow both efficient and portable code.
- As a result, most operating systems are written in C.

C

- C is efficient because of key design choices:
 - every feature in the language maps either directly to a machine instruction or via a small handful of machine instructions
- C does not abstract memory: allocating and cleaning up memory is the programmer's responsibility
- C has almost no "run-time" support.

What's a run-time?

- A language's *run-time* is responsible for any behavior of a program not directly attributable to the program itself.



What's a run-time?

The C run-time does

- startup
 - invoke the dynamic library loader (for DLLs)
 - initialize the call stack
 - map OS resources to program symbols (e.g., STDIN, STDOUT, STDERR, etc.)
 - call `_init`, `main()`
- do as little as possible while program runs
 - memory allocator
 - debug functions like `assert`
 - (optional) concurrency primitives
- shutdown
 - call `atexit`

What's a run-time?

The Java run-time does

- startup
 - everything that C does
 - initialize virtual machine
- shutdown
 - everything C does
 - run class finalizer code to clean up resources
 - shut down virtual machine

and...

What's a run-time?

The Java run-time does

- do lots of things while program runs
 - bytecode verification
 - dynamic class loading / initialization
 - dynamic type checking
 - automatic memory management: allocation & garbage collection
 - managing Java threads and thread pools
 - exceptions
 - program profiling, JIT compilation, and on-stack replacement
 - optional isolation

How to use C

- in file `helloworld.c`:

```
#include <stdio.h>

int main(int argc, char** argv) {
    printf("Hello world!\n");
    return 0;
}
```

- compile code:
\$ `clang helloworld.c -o helloworld`
- run program
\$ `./helloworld`

C Features

- Influenced by ALGOL, but simpler
- control: if/else, for, while, switch
- data types:
 - primitives: byte (8 bits), char (8 bits), short (16 bits), int (32 bits), long (64 bits), float (32 bits), double (64 bits)
 - complex: array, struct (demo), union

C Features

- user-defined functions (demo)
- explicit memory functions
 - manual storage (demo)
 - malloc
 - free
 - used when memory needs to outlive activation record (example)
 - "automatic" storage (demo)
 - "local" variable; allocated on the stack
 - otherwise, allocated on the heap
 - automatically "freed" when stack popped

C Features

- no memory abstraction
- pointers
 - a pointer is not a data type; it's just an int!
- operations
 - "address of" operator: &
 - takes any *variable* and returns its *memory address* (i.e., pointer)
 - "dereference" operator: *
 - takes any *pointer* and returns the *value* at that *memory address*
 - "member selection" operator: .
 - "pointer member selection" operator: ->
 - p->foo equivalent to (*p).foo