CSE 374: Programming Concepts and Tools

Eric Mullen
Spring 2017
Lecture 8: intro to C
• Posting shell scripts: They’ll be up on the website today (thanks several people who asked)

• Homework 2: Due tomorrow night at midnight. Late days would be inadvisable to use.

• Homework 3: Out by Friday lecture

• Make sure you’re using klaatu for your homework!
• Put on your archeologist hat…
• Think back to a wilder time (1971)…
• Let’s dive in!
C

- Contrast with Java:
  - Lower level, closer to machine
  - More unsafe (there are NO training wheels)
  - Procedural: no more objects
  - Standard library is small
  - Similar control and syntax
  - Fundamentally different mental model
C

• C I’m going to teach you is not all technically allowed by the standard

• rather, is how it actually works on x86_64 machines

• I will try to tell you when we’re deviating from the standard, but it can be subtle
Why C?

- Despite being old, it’s extremely ubiquitous
- Lots of new code, and lots of existing systems
- How anyone writes software to interact with hardware
C: Today

• Language basics
• Hello World
• Pointers
• Hello with arguments
C: Today

- Language basics
- Hello World
- Pointers
- Hello with arguments
Language Basics

• Basic types:
  • int
  • float
  • char

• More types:
  • pointers (*)
  • void (type of function with nothing to return)
Syntax

• Functions are declared similarly to Java:
  ```java
  int foo(int x) {
    return x;
  }
  ```

• Variables are declared similarly as well:
  ```java
  int x;
  int y = 0;
  ```

• While you can declare without initializing, don’t.
Standard Library

• You can “include” different functionality by using

#include<nameoflibrary.h>

• stdio.h contains printf, which prints to stdout
C: Today

- Language basics
- Hello World
- Pointers
- Hello with arguments
C: Today

- Language basics
- Hello World
- Pointers
- Hello with arguments
DEMO!
DEMO!

#include<stdio.h>

int main() {
    printf(“Hello World!\n”);
}

C: Today

- Language basics
- Hello World
- Pointers
- Hello with arguments
C: Today

- Language basics
- Hello World
- Pointers
- Hello with arguments
On the High Cs

- Memory is 1 dimensional array full of bytes
- You can make maps which refer to things
  - Maps are just a number, we call them pointers
- You can follow them wherever they lead

<table>
<thead>
<tr>
<th>Code</th>
<th>Globals</th>
<th>Heap -&gt;</th>
<th>&lt;- Stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000</td>
<td>0xFFFF</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Pointer Syntax

- When declaring a type, `int*` means “pointer to an int”

- When used in an expression, `*x` means follow `x` to where it goes
Careful!

- Would you always trust a pirate's map?
- **Never** blindly trust a pointer!
- What happens if you do?
C: Today

• Language basics

• Hello World

• Pointers

• Hello with arguments
C: Today

- Language basics
- Hello World
- Pointers
- Hello with arguments
Arrays

• How might you implement an array in C?
Arrays

• Arrays are just multiple things right next to each other in memory

• We hold on to an array by remembering where it starts

  • declare type with “int x[]”

• We get elements with square braces

  • e.g. x[3]
Wait a minute…

- Arrays sound a lot like something else…
Command Line Arguments

```c
#include<stdio.h>

int main(int argc, char* argv[]) {
    ...

    }
```
char* argv[]

• You can read this as: “argv is an array of pointers to characters”

• You can *implicitly know* that it’s really more like an array of strings

• In C, a string is really just a bunch of characters next to each other in memory, followed by a special “\0” character (a null byte)

• More on strings in a couple lectures
Demo