Topics Assaf mentioned for further reading/study at end of day 3 lecture 2016-11-15
email: **Assaf Tzur-El** <assaf@tzurel.co.il>

Internals

globals - when are they initialized
C and assembly (e.g. ++ operator translates to assembly's INC opcode)
Calling a function - how does it translate to machine code?
Calling conventions (last two items are covered here, but you'll probably want your assembly teacher to help you with that…)
X86 stack implementation, stack frames, and basically this whole WikiBook

Features and keywords

extern keyword (functions and globals)
static keyword:

- Automatic static variables (inside functions)
- static globals and functions

register keyword
volatile keyword
Function pointers (if you can sort an array of structs using qsort() by various fields, ascending and descending, and for each one do a binary search using bsearch(), you are getting the hang of it)
Pointers to pointers (AKA "char **") - the way to pass pointers by reference (wait, what?)
Variadics (the "..." at the end of printf()'s params)
Preprocessor concatenations using # and ## (here's a taste)

Optimizations

Duff's device (do NOT try this in public!) (note the obsolete C syntax in the example!)
A few examples of compiler optimization power

Debugging

Ever wondered why all your uninitialized variables have the same "garbage" value of -800 million? Ever tried to ask Visual Studio to show you the values in hexadecimal?
After doing this little exercise, you may now read this (my personal favorites are 0xBAADFOOD and 0xDEADBEEF)

C and the Operating System

Memory management (some of it we already covered in class)
OS API (e.g. Win32 API) - interacting with the OS
Threading and multiprocessing

Words of wizdom

Object-oriented languages are much cooler (and popular) than procedural ones. If you want a tough challenge, go for C++. If you want to be more productive (concentrate more on what you want to do than how to do it and where did my memory leak to), check out Java and C#. Every language has its dark corners, even languages that were created to simplify things (yes, I‘m looking at you, Java). Watch out.

register is simple to teach but has big implications

Reverse engineering
Assembly

How C code is translated into assembly
What it mean when calling from/returning from a function
What does a stack look like in assembly

Book: The C Programming Language by Kernighan and Ritchie 2nd edition (first edition is outdated and different from modern C)
(PDFs are out there if you google)