Languages, syntax, and semantics

- programming languages vs natural languages
- preponderance of programming languages
- language syntax vs semantics
- relationship between languages and hardware
- translation tools (compilers, assemblers, interpretters)

Programming vs natural languages

- natural languages (english, french, etc) have evolved over centuries to provide rich and nuanced communication

- programming languages aim to be clear, unambigous, and precise - allowing developer to instruct a machine exactly what to do

- programming languages are much much smaller than natural languages, with simpler grammar rules but stricter enforcement of those grammar rules (very limited ability to figure out what you mean unless you follow the rules exactly)

programming languages

- a plethora of different programming languages available (rosettacode.org/wiki/Category:Programming_Languages)
- some languages designed to be general purpose, others designed to best serve specific purposes (web development, network coding, system administration, etc)
- each language has its own precise grammar rules - the *syntax* of the language
- *semantics:* the actual meaning of a segment of code

Layers of programming language

- the actual machine hardware runs by interpretting binary electrical signals (on/off, low/high, 0/1, false/true)

- programs at the machine level are sequences of binary patterns - not easy for humans to read/write/modify

- assembly languages give names to specific patterns, e.g. 00011100001000110 might mean move value 6 into storage register 8, and be given a name like MOVE R8, #6

- easier for humans to read/write than binary patterns

- people write in assembly language, then use software tool (an assembler) to translate into machine code

High level languages

- The development of assembly language and assemblers was first big step to more readable code, but still not very human friendly

- high level languages (HLLs) are much closer to natural languages, allow developers to express complex actions in a more natural way

- another software tool, the compiler, is used to translate code written in a high level language into executable machine code (i.e. the final binary patterns)