Overview of Computer Science: Part III
Programs and Programming Languages

Programming Languages

Machine Language: the set of instructions that CPU understands and executes.

- Machine instruction a sequence of binary numbers (instruction code, operands).
- **Extremely hardware-dependent**.
- Very hard to use for building complex software.
- Reasoning/actions at the level of registers and main memory cells.

Assembly Language: replaces machine instructions with mnemonics.

- Instead of binary codes, mnemonics for instructions: ADD, MOV, SUB, etc....
- More natural representation of operands (memory cell addresses, register addresses, numbers).
- Still, **hardware-dependent**.
- Still, hard to use for software development.
- Used for development of system software.

High-level Languages: hardware-independent languages for describing instructions for computer system.

- **Hardware-independence** means the same program can be used on many different hardware platforms.
- Require **translation** into hardware-specific assembly (and then - to machine code).
- Translation is done by special class of software programs: translators:

- Interpreters: take a program in high-level language and translate and execute each instruction in turn.
- Compilers: that a program in high-level language, translate it into machine language, and save the machine language code for future use.

Source file: a file containing the text of the program in a high-level language.

Executable file: a file containing the program in machine code, ready to be executed on the computer system. (also, sometimes called object file.)

Program execution, running a program: the process of transferring a program into main memory of the computer system and performing its instructions in prescribed sequence.

High-Level Programming Languages

A program in a high-level programming language is a text.

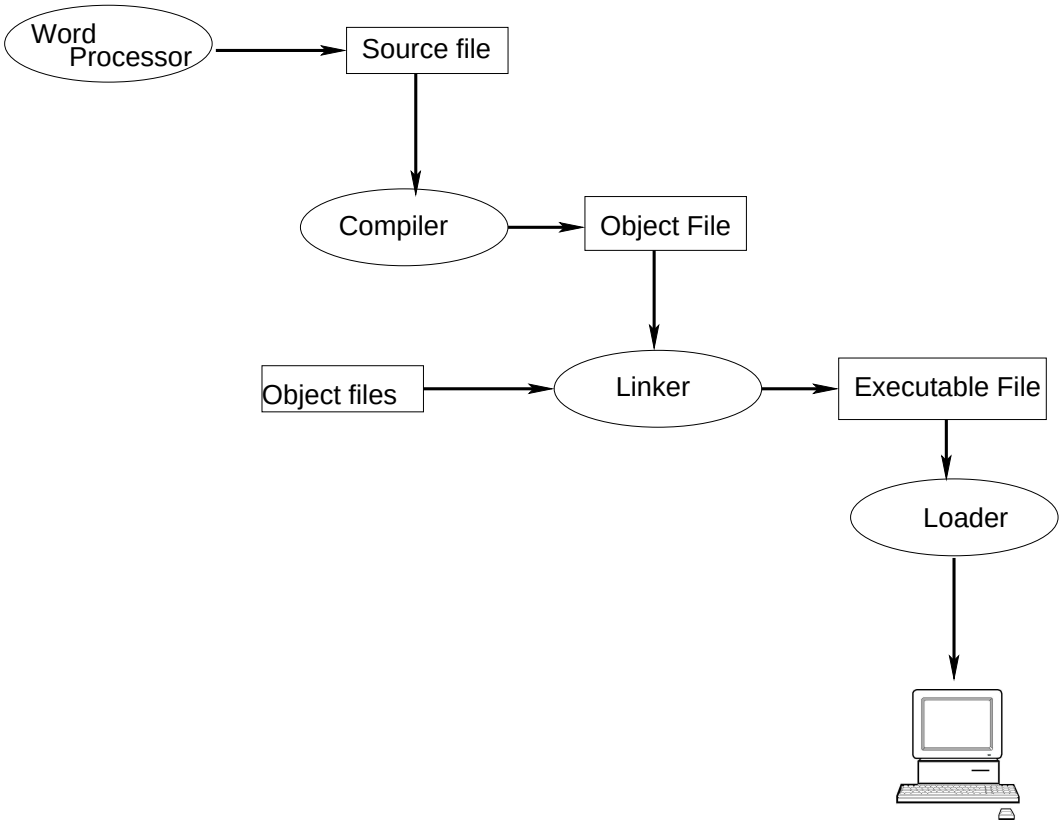
Syntax. The collections of rules specifying what constitutes a *proper* program in a specific language.

Each programming language has its own set of syntactic rules. These rules must be specific enough, so that for any "text" it can be determined if it is a proper program.

Semantics. Defines the **meaning** of the program written in a high-level language. Essentially, specifies, **what a computer system will do** when executing a program.

*To learn a high-level programming language one must learn both its **syntax** — to know how to compose programs correctly, and its **semantics** — to know how to write programs that achieve specific goals.*

From source to executable



- Word processor: source files are created using word processing software (or text editors).
- Compiler: translates source files into object files containing machine code.
- Linker: combines multiple object files into a single executable file.
- Loader: puts executable file into main memory, starts the execution of the program.