Translation

- Converting from a source language to a target language
- Necessary when there is no CPU or interpreter that can execute programs in the source language
- The target language is usually ISA-level binary instructions
- Compiler
  - Translates from a high-level language (like C or Java) to machine language or a symbolic representation of machine language (assembly language)
- Assembler
  - Translates from a symbolic representation of a machine language to the machine language itself

Assembly Language

- A human-readable symbolic representation of a machine language
- Actual ISA instructions are encoded in binary, assembly instructions have mnemonic names (add, sub, mult, etc)
- Assembly programming allows the use of ISA-level features that may not be accessible from a high-level language (like accessing overflow)

Assembly process

- Translating from assembly language to machine language
- Cannot be done line-by-line in one pass
  - Labels exist at the assembly level, but they become addresses at the machine level
  - Branching requires jumping to an instruction address (through the program counter)
  - Jumping forward, the address of the label is not yet known
- First pass builds a symbol table which maps label strings to instruction addresses

Code modules

- Complex programs benefit from being split into multiple files
  - Easier to read
  - Easier to test individual functions and classes
  - Modules can be compiled independently. After a change, only the modified modules need to be re-compiled
- Modules are compiled into object files
- Object files are combined to form executables

Linking

- Often a compiled object file contains jump instructions that jump to an instruction in a different object file
  - Other modules in the same program
  - External libraries
- Linker
  - Creates a single executable file from multiple object files
  - Inserts jump instruction addresses into the machine code once the final locations are known

Dynamic linking

- System libraries need not be compiled into every program that uses them
  - C standard library
  - Networking code
  - etc.
- One copy of the library's machine code can be loaded into memory, and every running process that uses the library can be linked to the one copy at runtime

- Windows: Dynamically Linked Libraries (DLLs)
- POSIX: Shared libraries