Compiler

(definition and error handling)

Interpreter

(definition and error handling)

Assembler

(definition and error handling)

Compiler

(Benefit and Drawback)

Interpreter

(Benefit and Drawback)

Why use high-level programming languages?

- **Converts** a high level program into machine code for execution at a later time (as an executable file).
- The entire program is converted (not one line at a time like an interpreter).
- Error details are stored in a diagnostic file.

- **Converts and executes** a high level program into machine code one line at a time.
- As soon as it hits a problem, the error is immediately reported to the user and further execution of the program is halted.

- **Converts** a **low level assembly language** into machine code.

Benefit:
- creates more efficient code than interpreters so compiled programs run faster.

Drawback:
- displaying multiple errors at the same time means compilers tend to be more difficult to use.

Benefit:
- easier to use as errors are reported and corrected one at a time, not all at once.

Drawback:
- slower than a compiler because looped code can take a long time to get through one line at a time.

- instructions use English so easier to read/write than a low-level language
- easier to maintain code
- leads to fewer errors
- simple commands perform complex tasks, *such as sort() in Python.*