

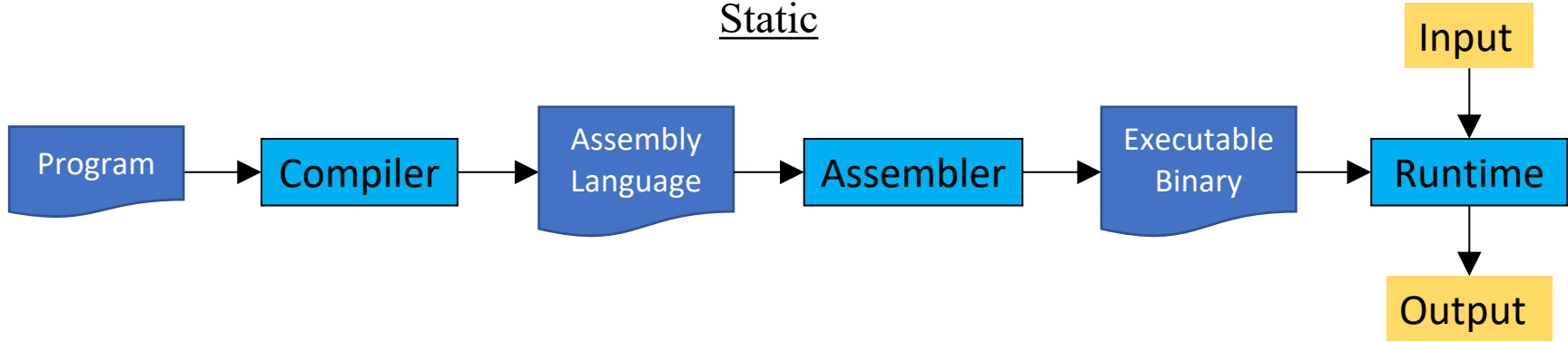
Introduction to Compilers

CMPT 379: Compilers

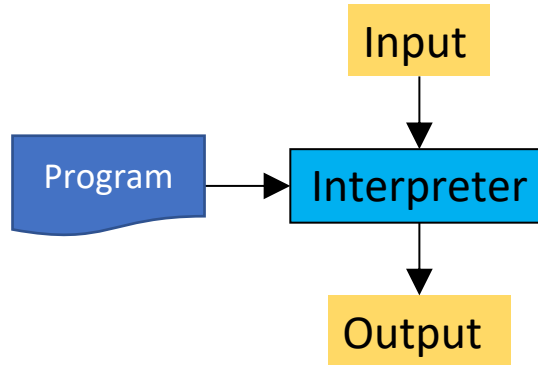
Instructor: Anoop Sarkar

anoopsarkar.github.io/compilers-class

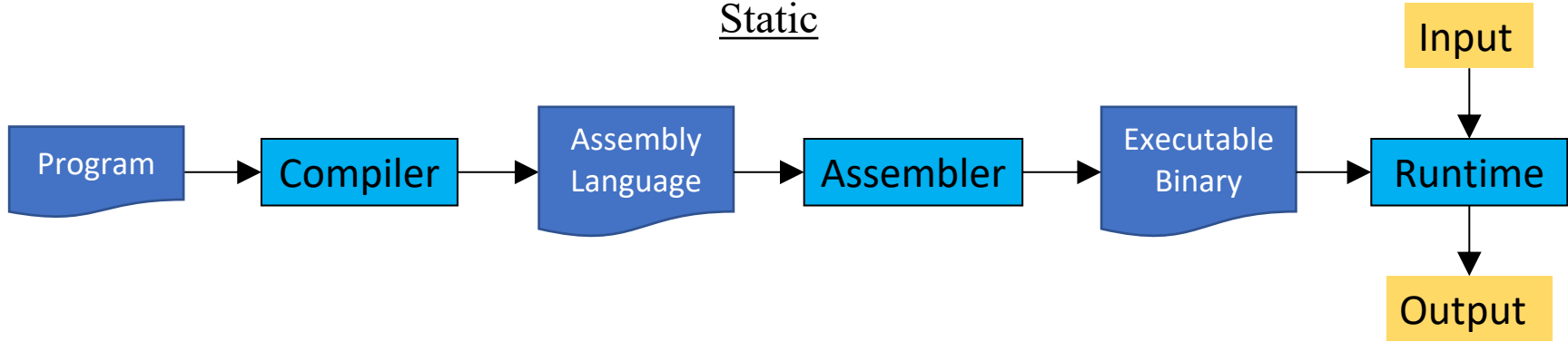
Static



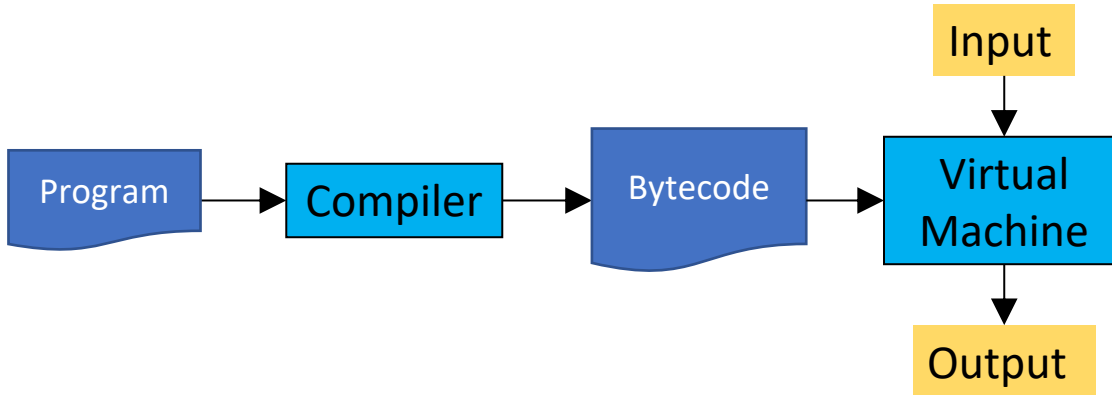
Dynamic



Static



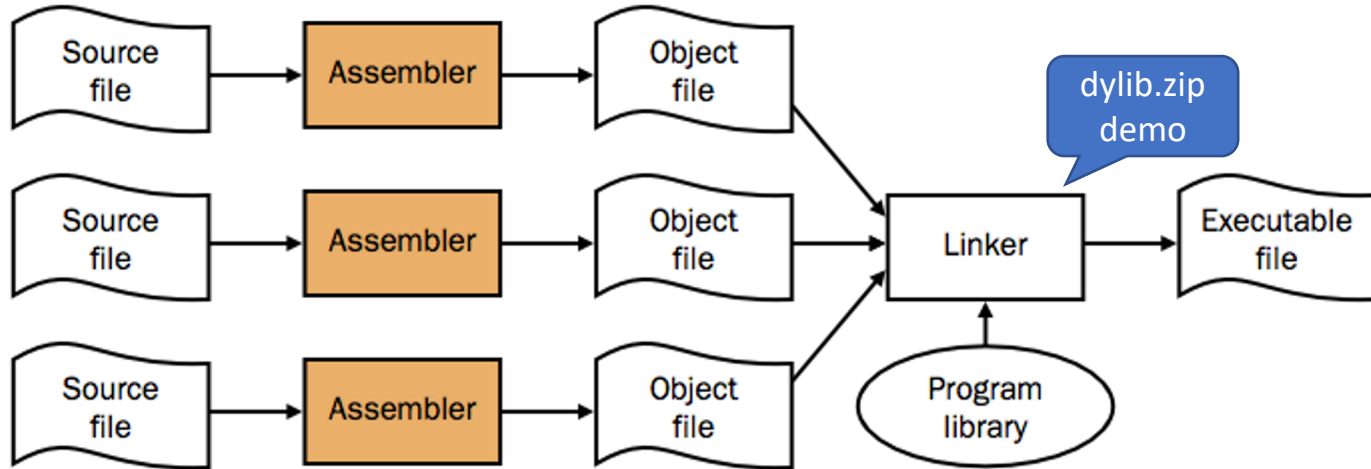
Static/Dynamic



Compilers

- Analysis of the source (front-end)
- Synthesis of the target (back-end)
- The *translation* from source code to executable
- Requirements from a Compiler are:
 - Support high-level programming languages
 - Good error messages
 - Speed of compilation
 - Produce fast executables with small footprint

The UNIX toolchain



Bootstrapping a Compiler

- Machine code at the beginning
- Make a simple subset of the language, write a compiler for it
- Use that subset for the rest of the language definition
- Bootstrap from a simpler language
- Interpreters: write it in a lower level language,
 - e.g. Python interpreter is written in C
- Cross compilation
 - generate ARM executables with compiler running on x86

Modern challenges

- Instruction Parallelism
 - Out of order execution; branch prediction
- Parallel algorithms:
 - New programming languages for [grid computing](#), [multi-core computers](#)
- Memory hierarchy: register, cache, memory
- New architectures: GPUs, quantum computers, DNA computers
- Hardware synthesis (extreme inner loop optimization)