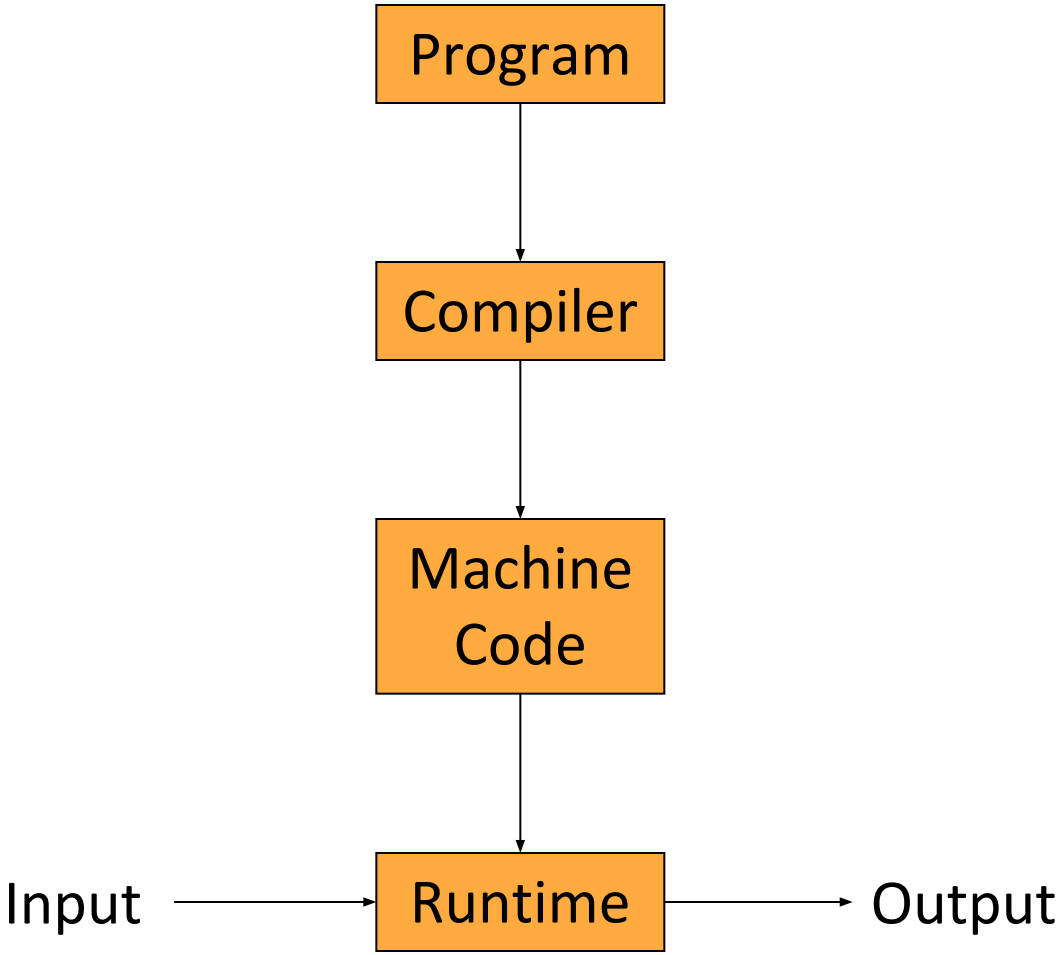
IN3: What is a Compiler

CMPT 379
Compilers

Anoop Sarkar

http://anoopsarkar.github.io/compilers-class/

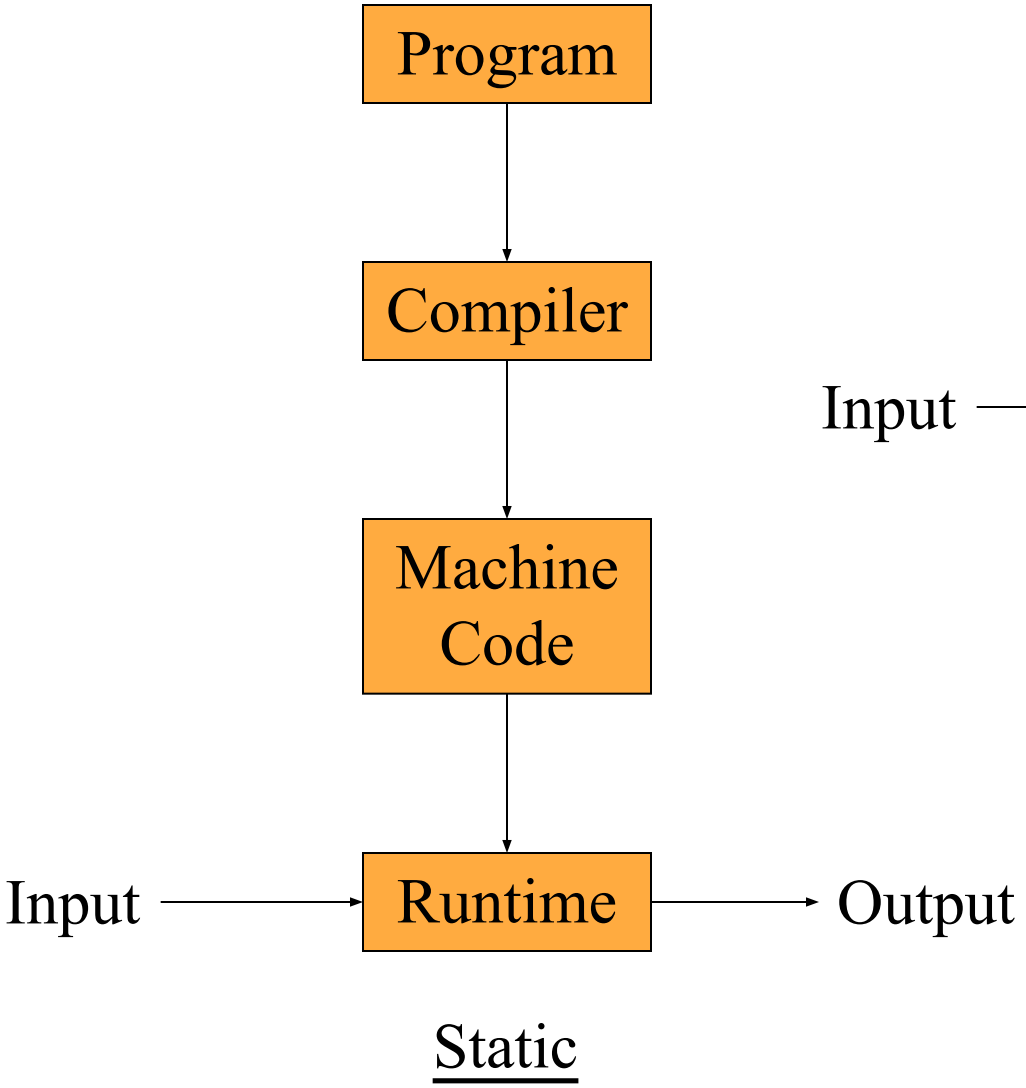

Program
Compiler
Machine Code
Input
Runtime
Output

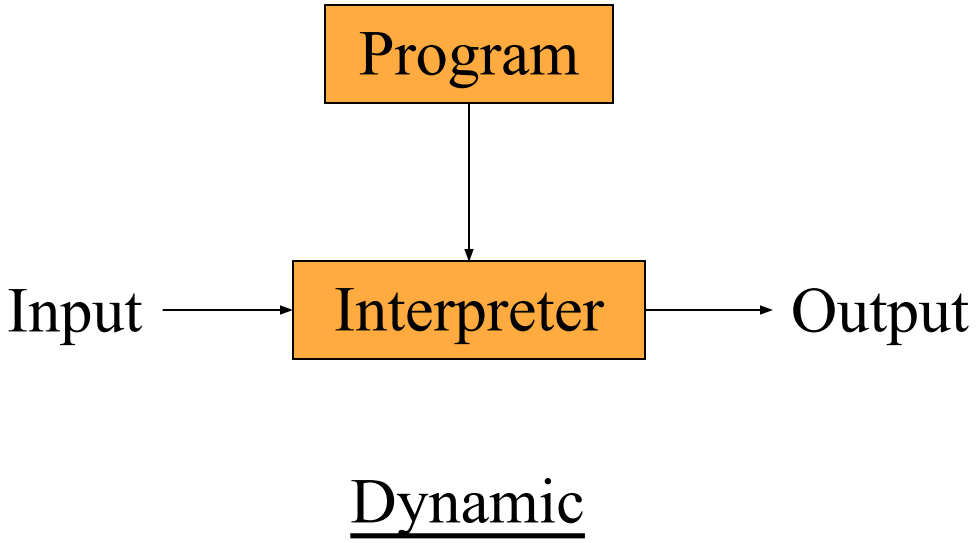
Compilers

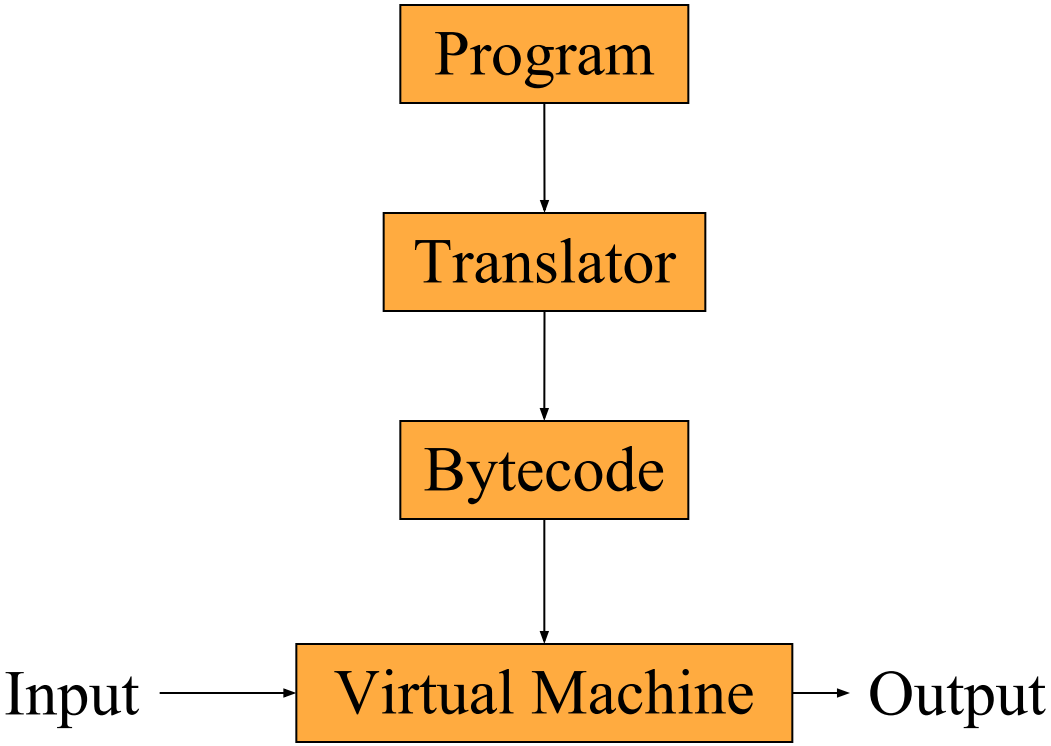
- Analysis of the source (front-end)
- Synthesis of the target (back-end)
- The *translation* from user **intention** into intended **meaning**
- The requirements from a Compiler and a Programming Language are:
  - Ease of use (high-level programming)
  - Speed

Cousins of the compiler

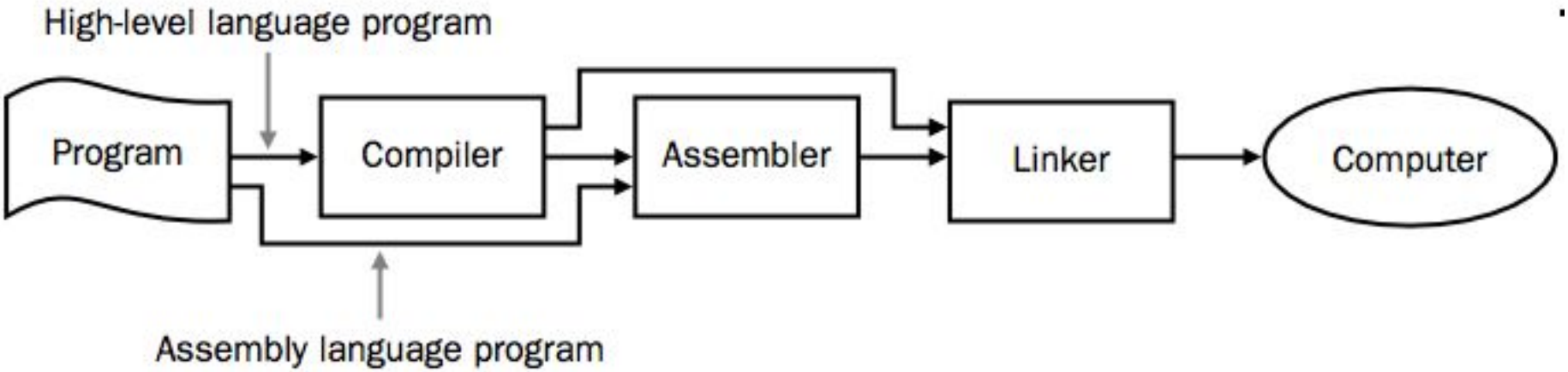
- “Smart” editors for structured languages
  - static checkers; pretty printers
- Structured or semi-structured data
  - Trees as data: s-expressions; XML
  - query languages for databases: SQL
- Interpreters (for PLs like lisp or scheme)
  - Scripting languages: perl, python, tcl/tk
  - Special scripting languages for applications
  - “Little” languages: awk, eqn, troff, TeX
- Compiling to Bytecode (virtual machines)


Program
Compiler
Machine Code
Input
Runtime
Output
Static


Program
Input
Interpreter
Output
Dynamic



Static/Dynamic


High-level language program
Program
Compiler
Assembler
Linker
Computer
Assembly language program

High level programs

```
extern void print_int(int);

class C {
  bool foo() { return(true); }
  int main() {
    if (foo()) {
      print_int(1); }
    }
}
```

Assembly language

```
; ModuleID = 'C'

declare void
@print_int(i32)

define i1 @foo() {
entry:
  ret i1 true
}
```

```
define i32 @main() {
entry:
  br label %ifstart
ifstart:
%calltmp = call i1 @foo()
  br i1 %calltmp, label %iftrue, label
%end
iftrue:
call void @print_int(i32 1)
  br label %end
end:
  ret i32 0
}
```

Translation from assembly to machine code

The UNIX toolchain

`as, ar, ranlib, ld, …`