C++ Compilation and Linking Process

Humayun Kabir
Professor, CS, Vancouver Island University, BC, Canada

C++ Development Environment

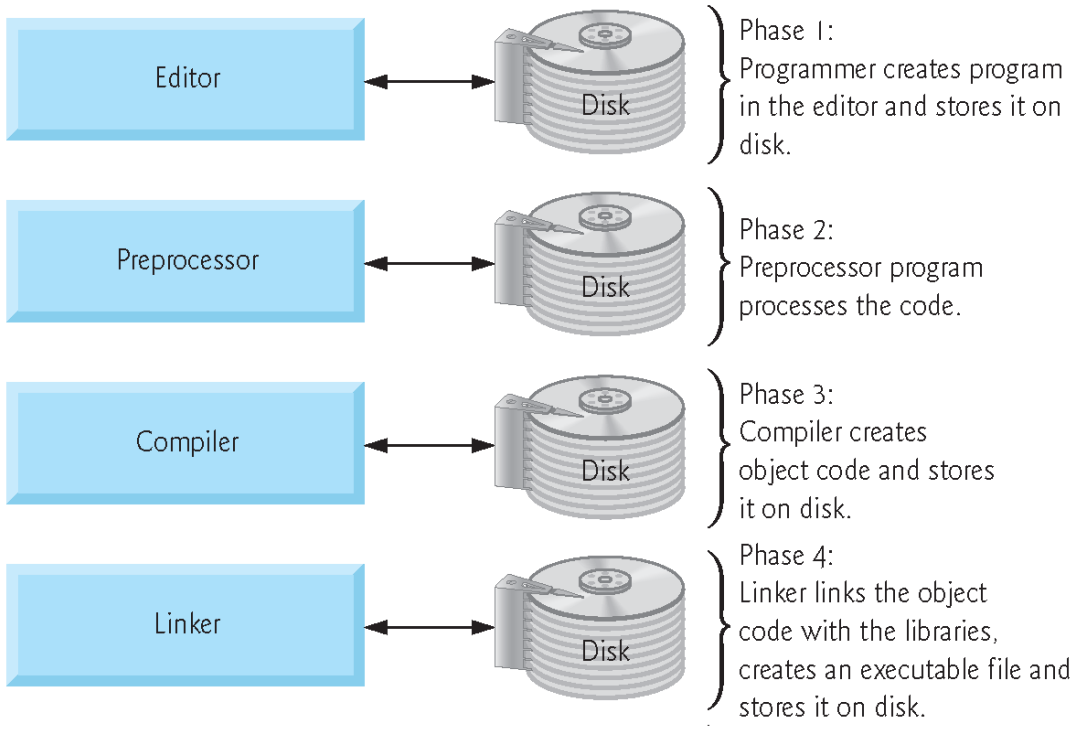
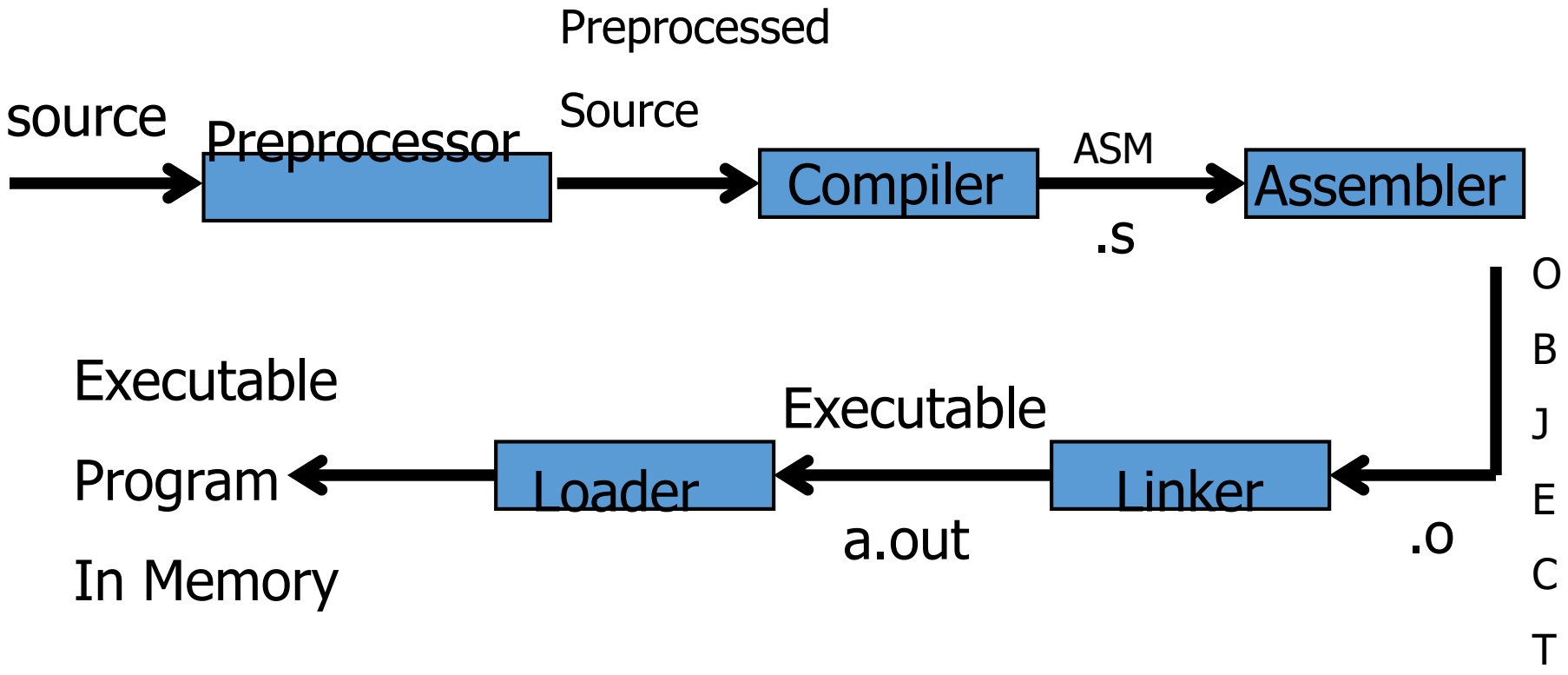
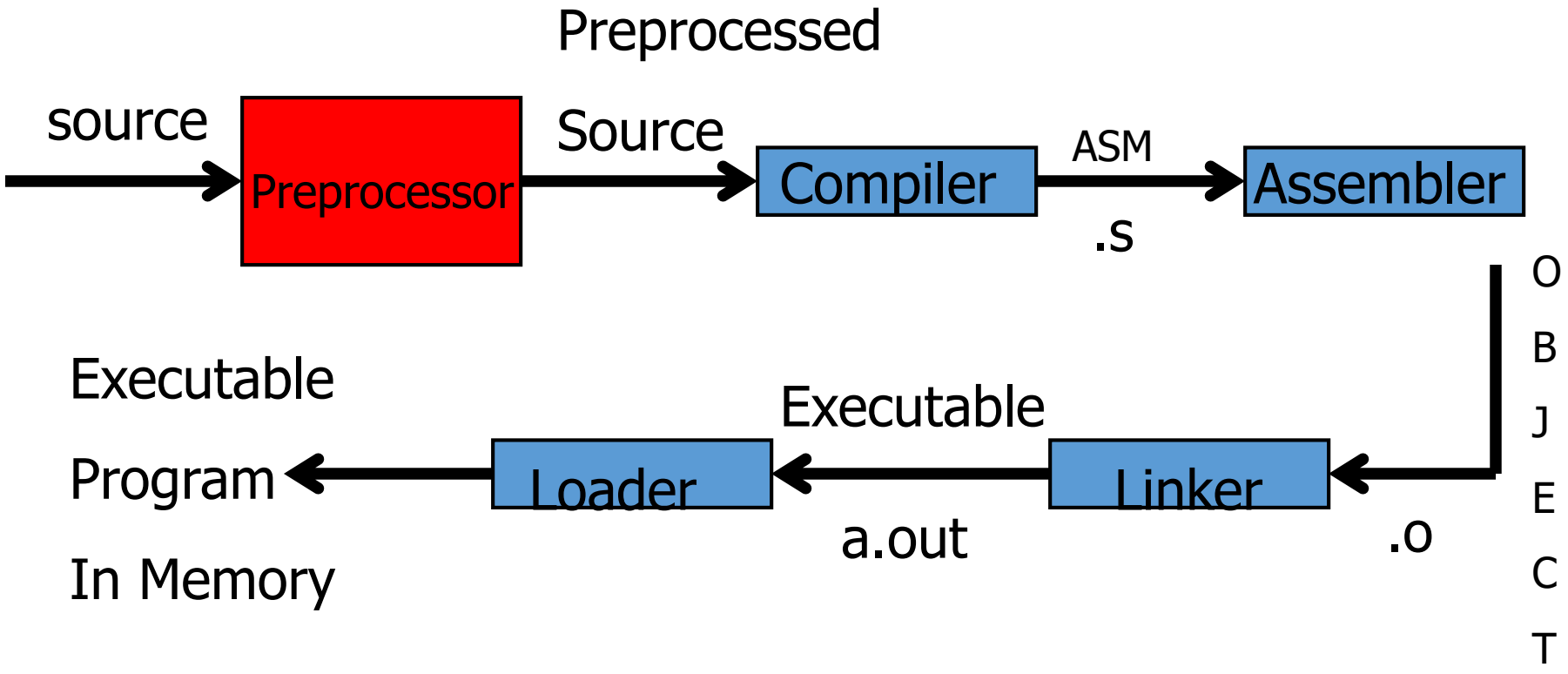


Fig. 1.7 | Typical C development environment. (Part 1 of 3.)

C++ Compilation Process



C++ Compilation Process: Preprocessing



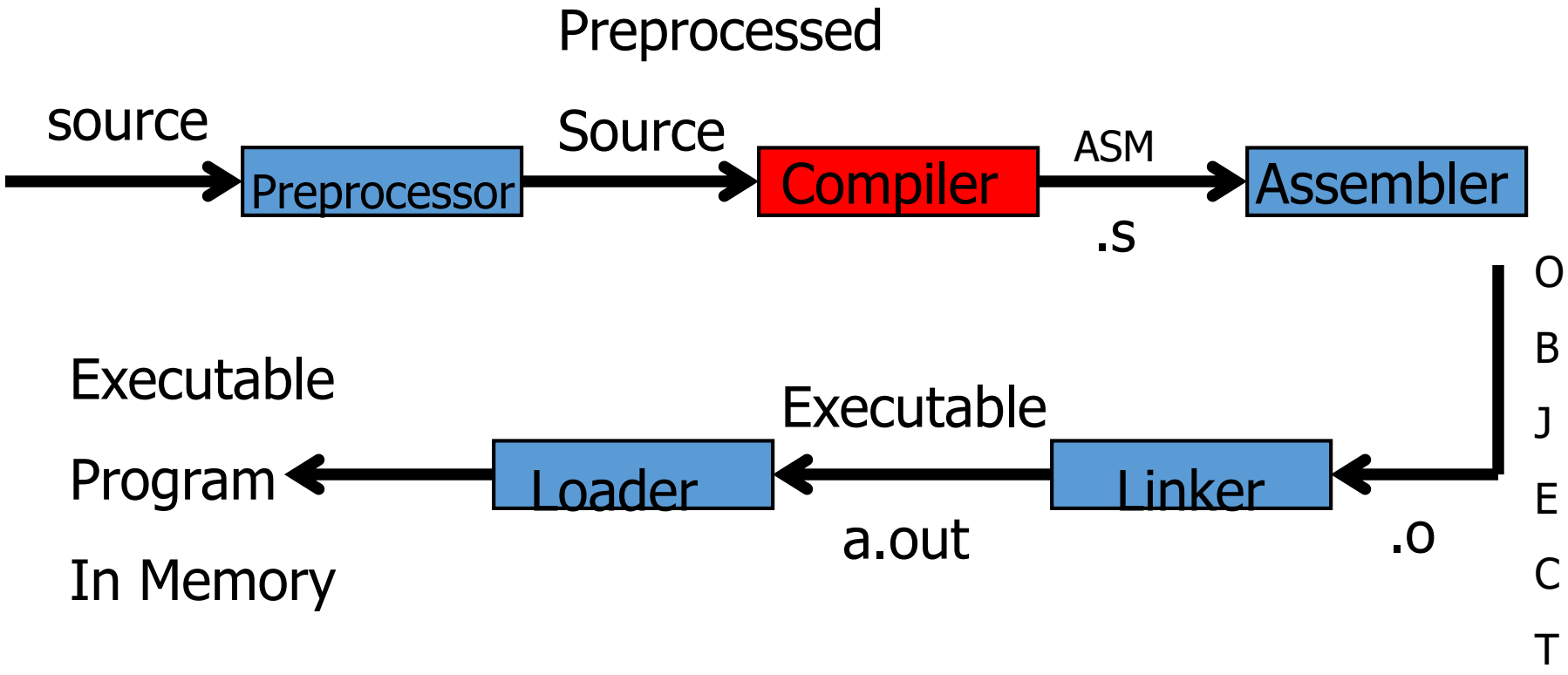
C++ Compilation Process: Preprocessing

- Pass over source
  - Insert included files
  - Perform macro substitutions

- Define macros
  - #define NUM 100
  - #define TASK(op, d1, d2) \
    (d1 op d2)

- **gcc –E example.c** sends preprocessor output to stdout

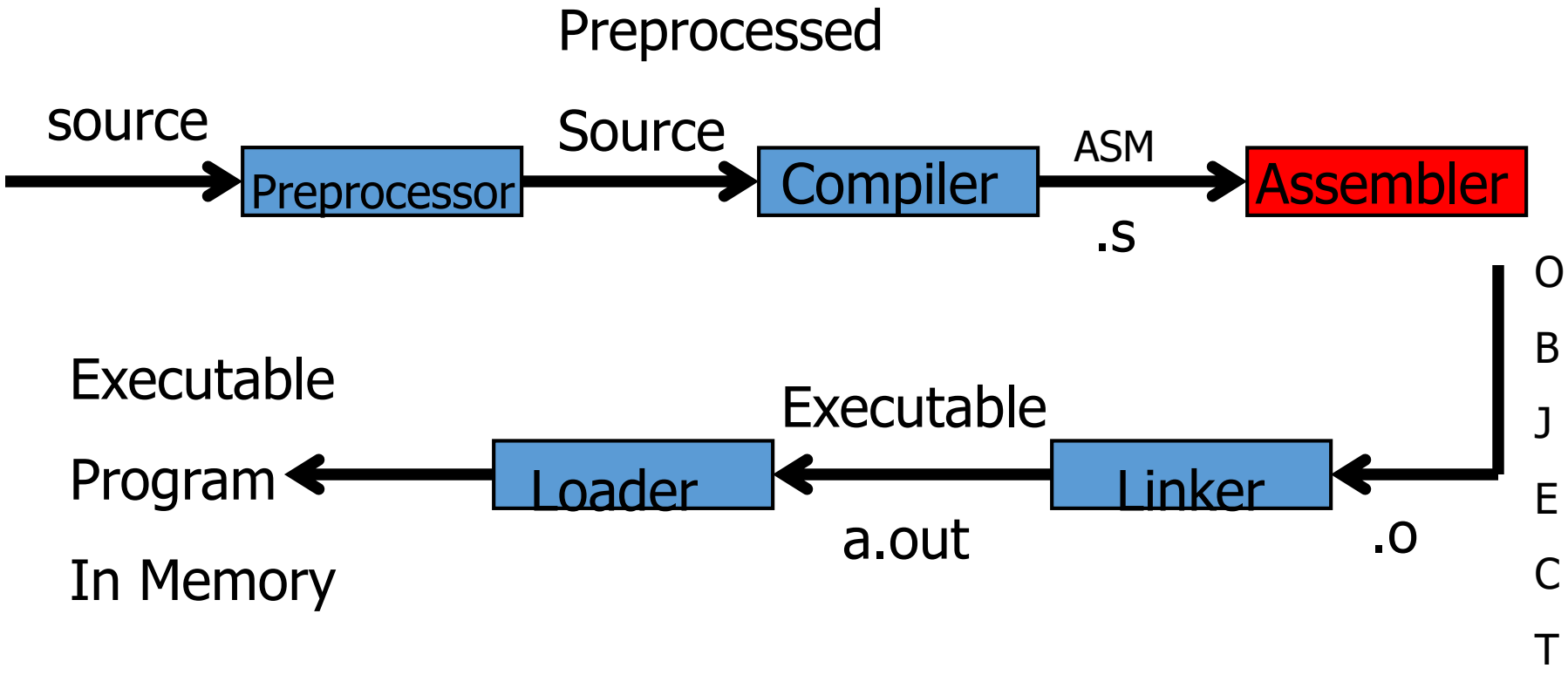
C++ Compilation Process: Compiling



C++ Compilation Process: Compiling

- gcc actually name of a script
- Compiler translates one language to another
- gcc compiler translates C to assembler
- **gcc –S example.c** “saves” assembler output to **example.s**
- Compiler consists of
  - Parser
  - Code generation
  - Mysticism

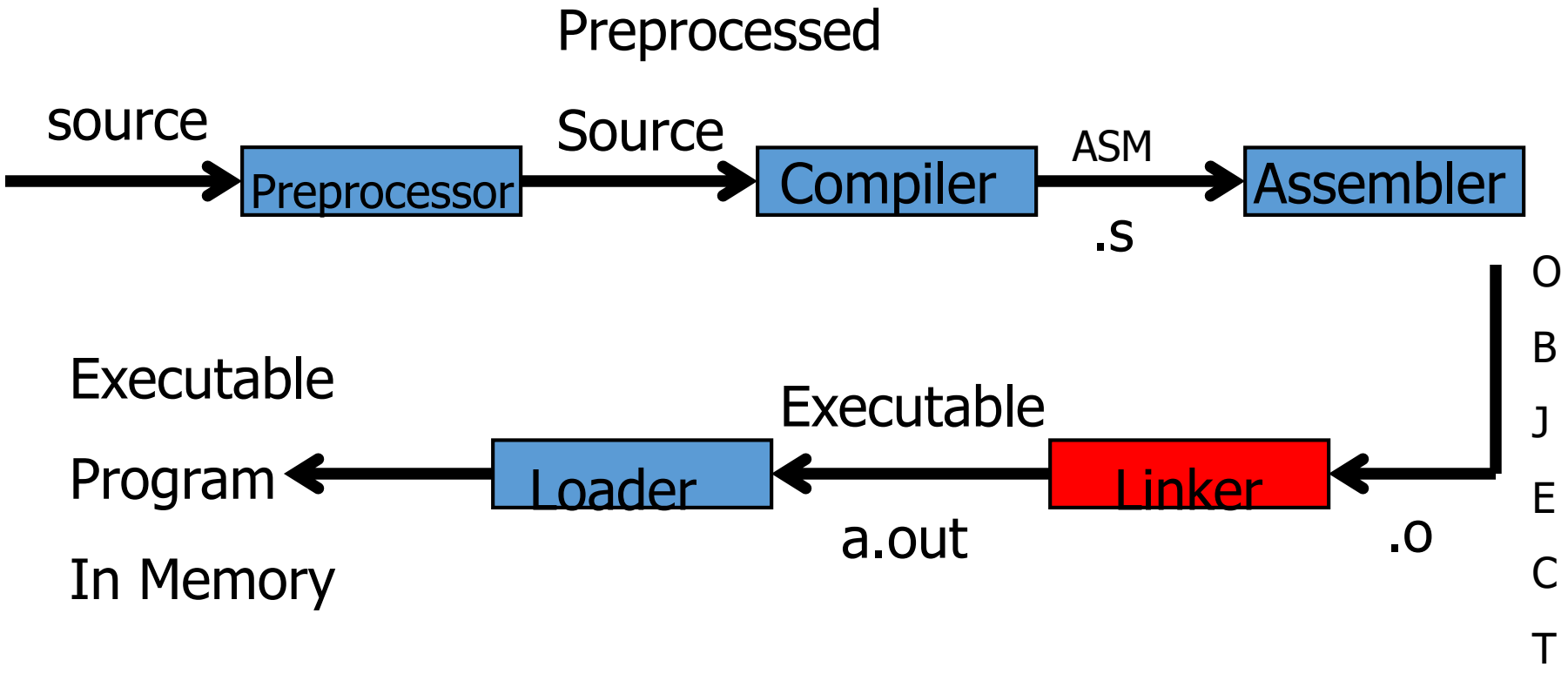
C++ Compilation Process: Assembling



C++ Compilation Process: Assembling

- Another translator ??? (as example.s)
- Assembler to (binary) object
- Why not compile straight to binary?
- **gcc –c example.c** to “save” (binary) object **example.o**
- Use **nm** to look at object (**nm example.o**)

C++ Compilation Process: Linking

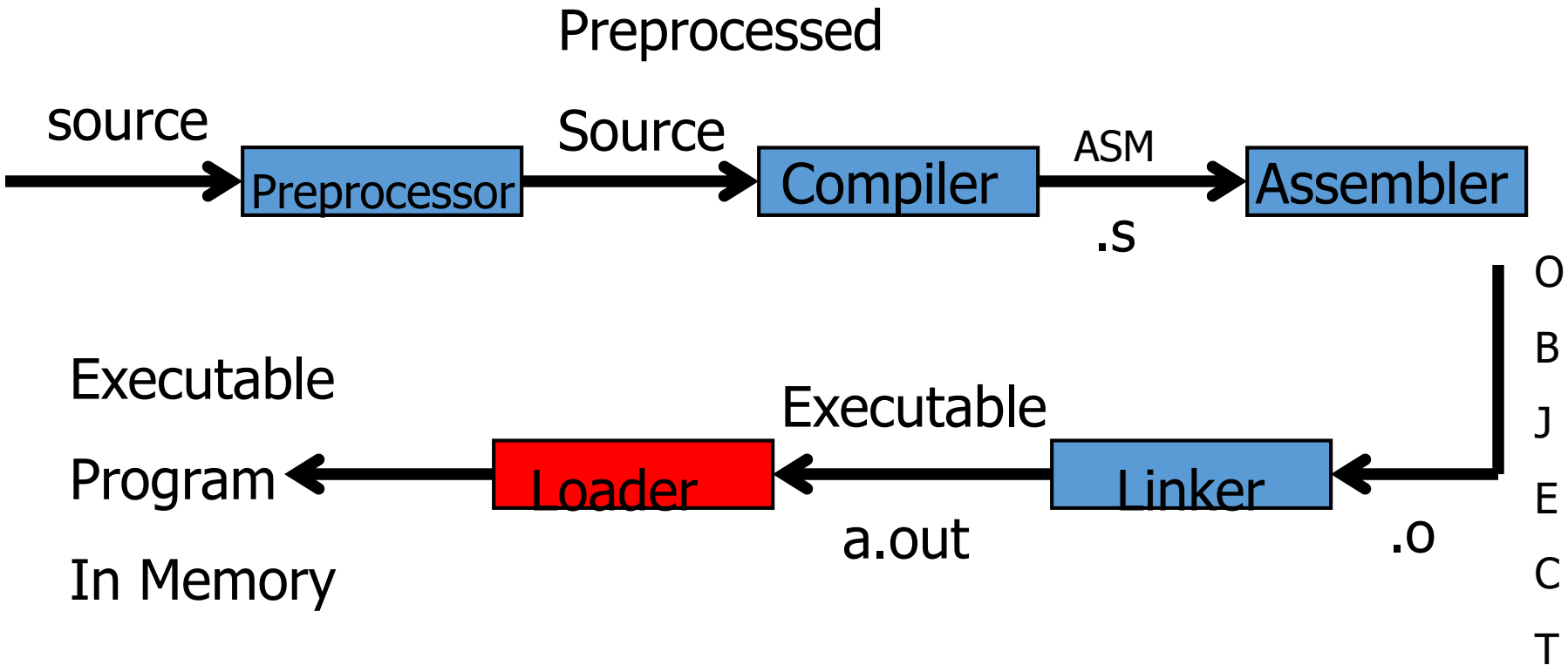


C++ Compilation Process: Linking

- Combines objects, both user .o files and libraries; makes an executable file
- **gcc *.o** yields executable **a.out** linking user and standard library objects.
- **gcc –o myExec *.o** yields executable **myExec** linking user and standard library objects.
- **gcc –o myExec *.o –lm** yields executable **myExec** linking user, standard library, and math library objects.

- Use nm to look at executable

C++ Loading


source
Preprocessor
Preprocessed
Source
Compiler
ASM
.s
Assembler
OBJECT
.o
Linker
Executable
a.out
Loader
Executable
Program
In Memory

Loader

- Runs when you type ./a.out

- Gets an address to place program (from the operating system)
- Changes necessary addresses (if any)
- Places code into memory

Loader

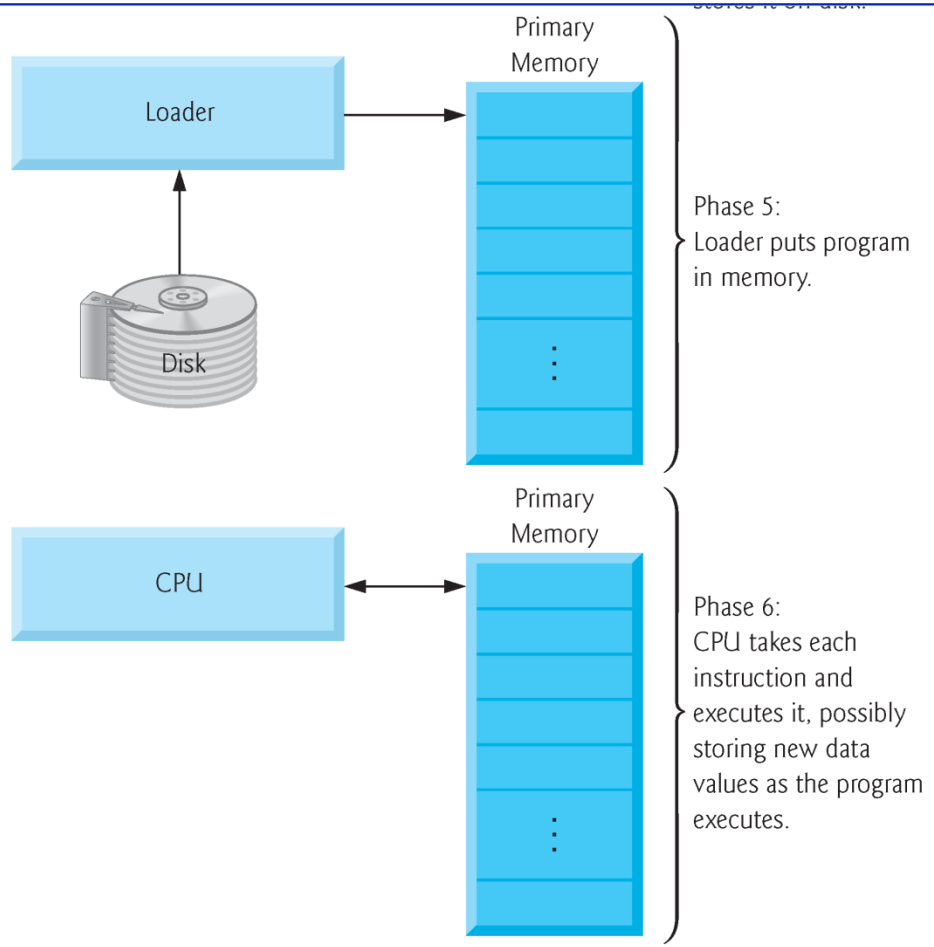


Fig. 1.1 | Typical C development environment. (Part 2 of 2.)