Programming

Instructor: Dmitri A. Gusev

Fall 2007

CS 210: Computing and Culture

Lecture 3, September 17, 2007

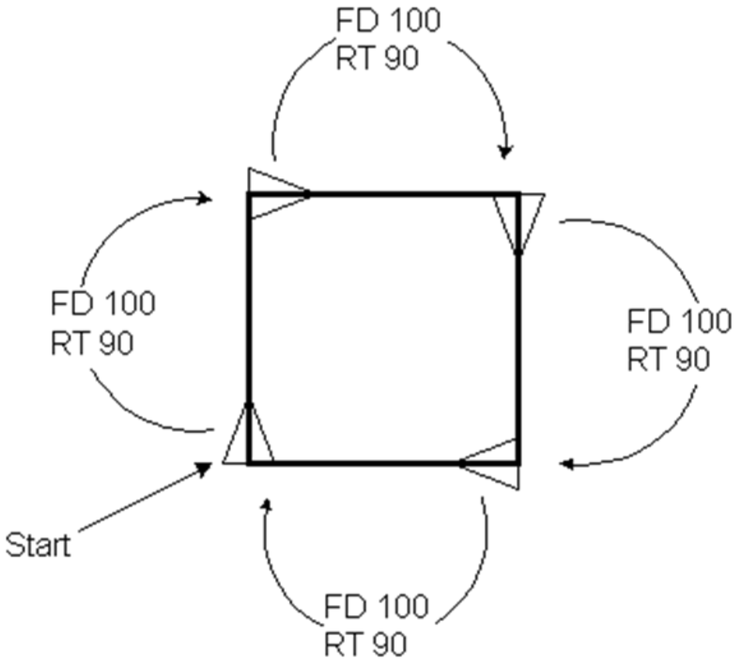
Programming Language

A *programming language* is an artificial language that can be used to control the behavior of a machine, particularly a computer. Programming languages, like human languages, are defined through the use of syntactic and semantic rules, to determine structure and meaning respectively. Unlike a human language, there is exact meaning in the programming language for every word and sentence.

Logo

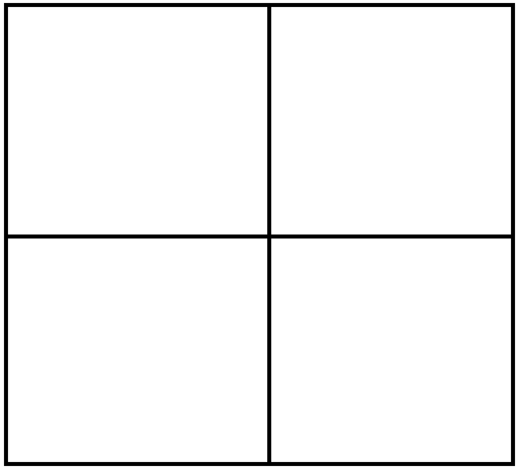
http://mckoss.com/logo/

```
TO SQUARE
FORWARD 100
RIGHT 90
FORWARD 100
RIGHT 90
FORWARD 100
RIGHT 90
FORWARD 100
END
```



Logo Subroutines

```
TO WINDOW
SQUARE
SQUARE
SQUARE
SQUARE
END
```



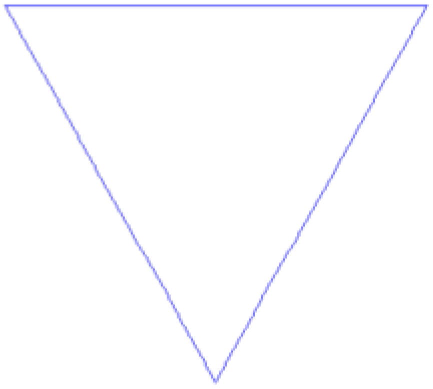
SQUARE is a subroutine of program WINDOW, which calls it.

Recursion

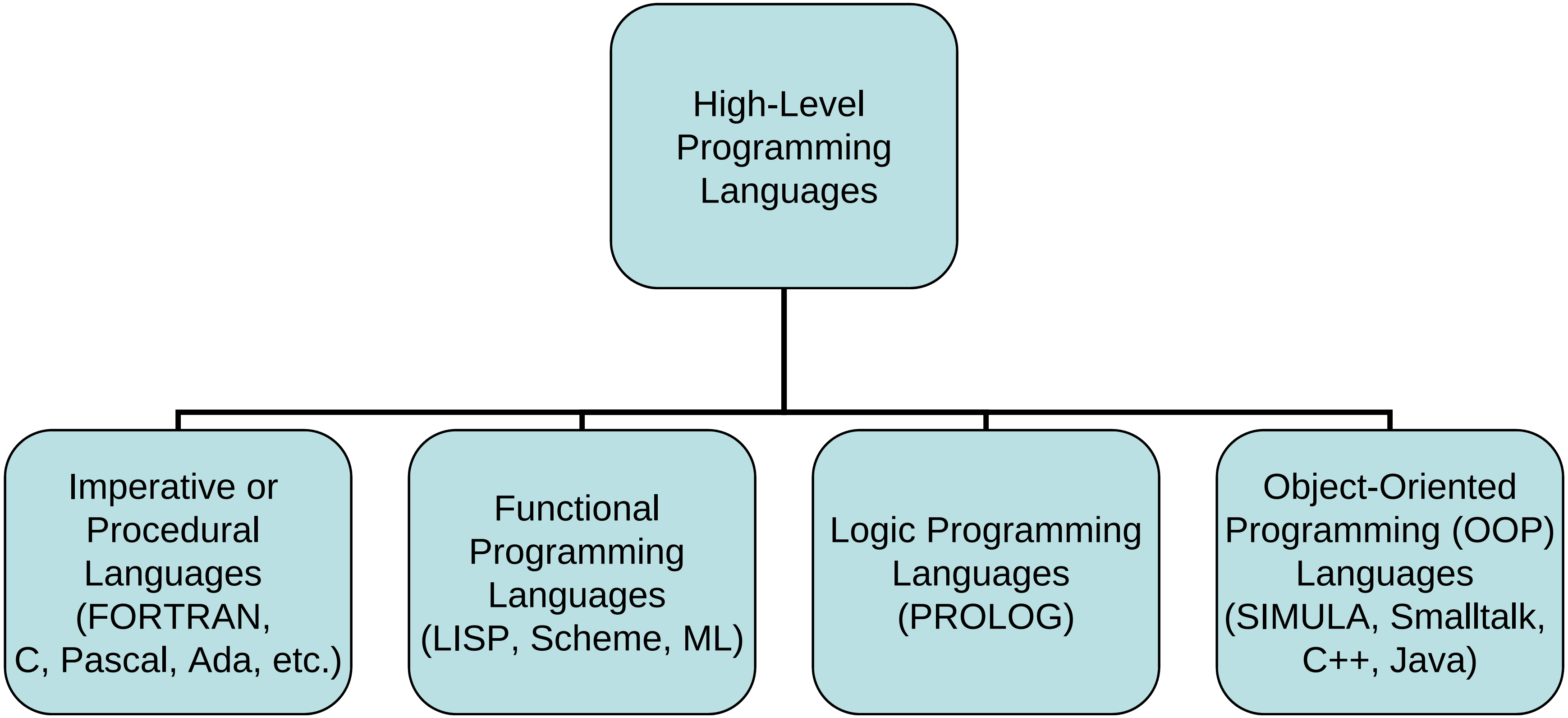
```
TO DESIGN
SQUARE
RIGHT 10
DESIGN
END
```

Fractals

Mandelbrot (1975): A fractal is a rough or fragmented geometric shape that can be subdivided in parts, each of which is (at least approximately) a reduced-size copy of the whole.



Classification of High-Level Programming Languages



High-Level Programming Languages

Imperative or Procedural Languages (FORTRAN, C, Pascal, Ada, etc.)

Functional Programming Languages (LISP, Scheme, ML)

Logic Programming Languages (PROLOG)

Object-Oriented Programming (OOP) Languages (SIMULA, Smalltalk, C++, Java)

Translating Programs

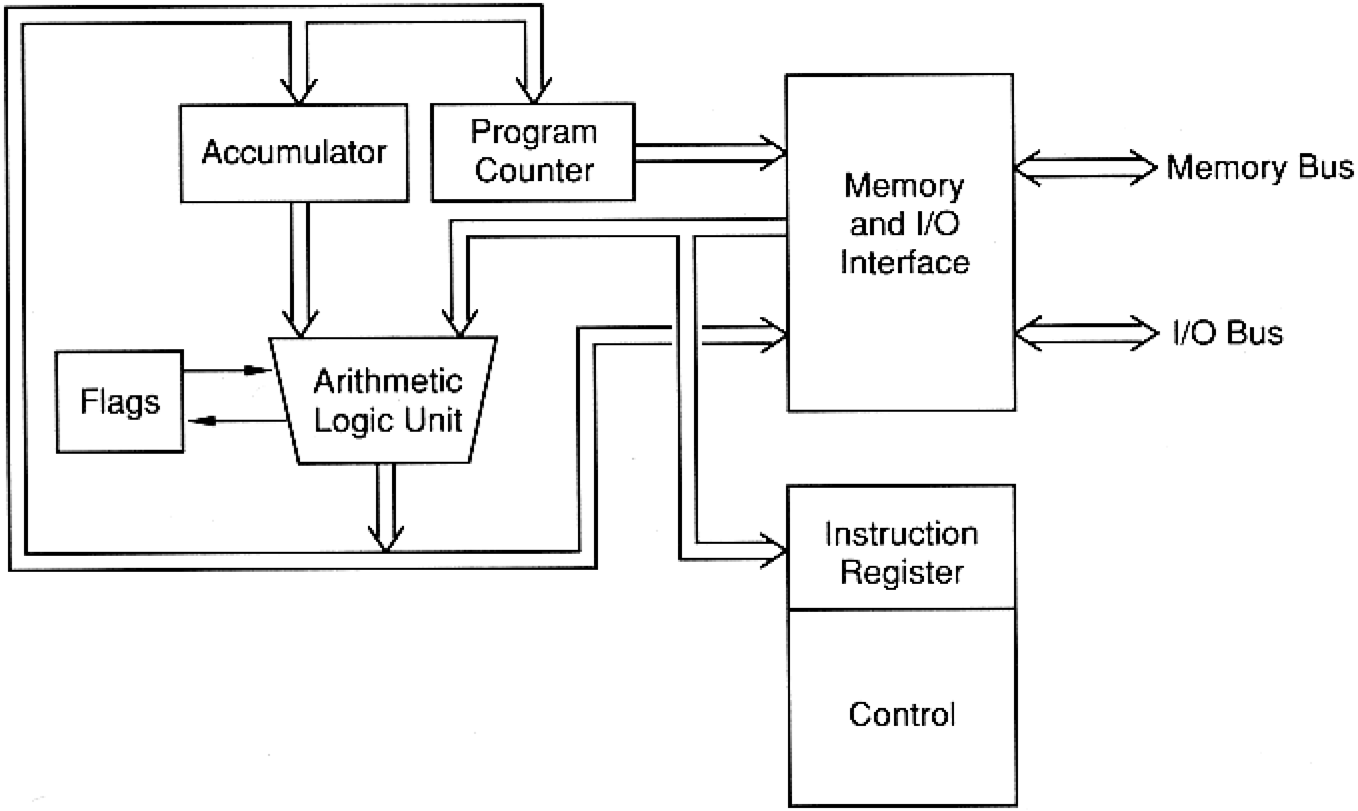
Assemblers translate the assembly-language instructions into machine code, or machine language. The assemblers are translating programs for low-level programming languages.

Programs that translate high-level language programs into machine code are called *compilers*. For a high-level programming language to be used on multiple types of machines, many compilers for that language are needed.

A program that translates from a low level language to a higher level one is a *decompiler*.

An *interpreter* is a translating program that inputs a program in a high-level language and directs the computer to immediately perform the actions specified in each statement. Interpreters can be viewed as simulators for the language in which a program is written.

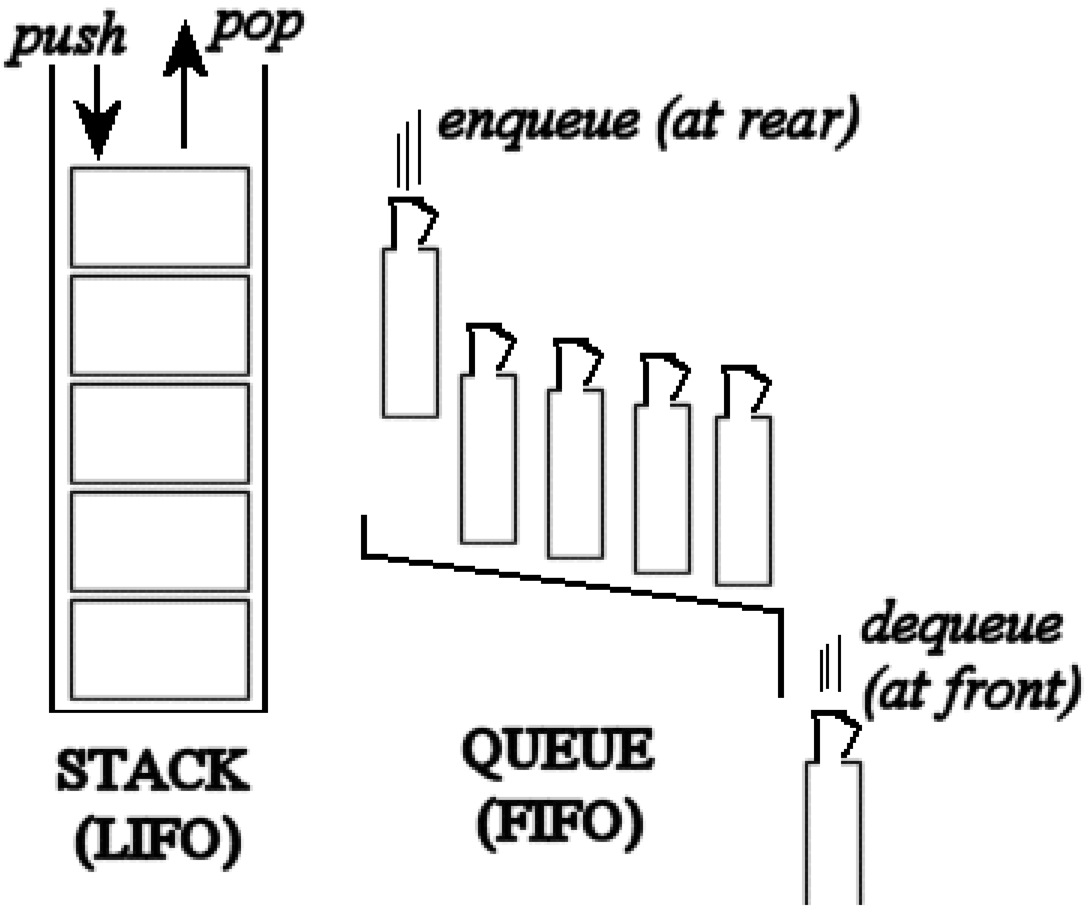
Block Diagram of a Simple Processor



Instructions

- *Processing instructions* move data to and from the memory and perform arithmetic and logic functions
- *Control instructions* determine the address of the next instruction to be fetched; this address is stored in a special register called the program counter (PC)

Stacks and Queues



Functions of an Operating System

The *operating system (OS)* is the core of the system software. It manages computer resources (memory, input/output devices) and provides an interface for *human-computer interaction (HCI)*.

Computer hardware is wired to initially load a small set of system instructions stored in permanent (*nonvolatile*) memory (ROM). Its popular name, *BIOS*, stands for *Basic Input/Output System*. BIOS *boots* the computer by loading a larger portion of systems software, usually from the hard disk. Nowadays, BIOS usually resides on *EEPROM* (Electrically Erasable Programmable Read-Only Memory) or *flash memory*.

The terms *dual-boot* and *multi-boot* system apply to computers that have two or more operating systems, respectively.

Functions of an Operating System (cont’d)

Multiprogramming is the technique of keeping multiple programs in main memory at the same time.

Memory management means keeping track of what programs are in memory and where in memory they reside.

A program in execution is called a *process*. A process may get interrupted during execution. A *context switch* is the procedure of storing and restoring the state (*context*) of a CPU so that multiple processes can share a single CPU resource.

Process management means keeping track of information for active processes.

CPU scheduling determines which process in memory is executed by the CPU at any given point.

- With the exception of its input/output mechanisms, the computer we’ve just described is simply a finite-state machine connected to memory.