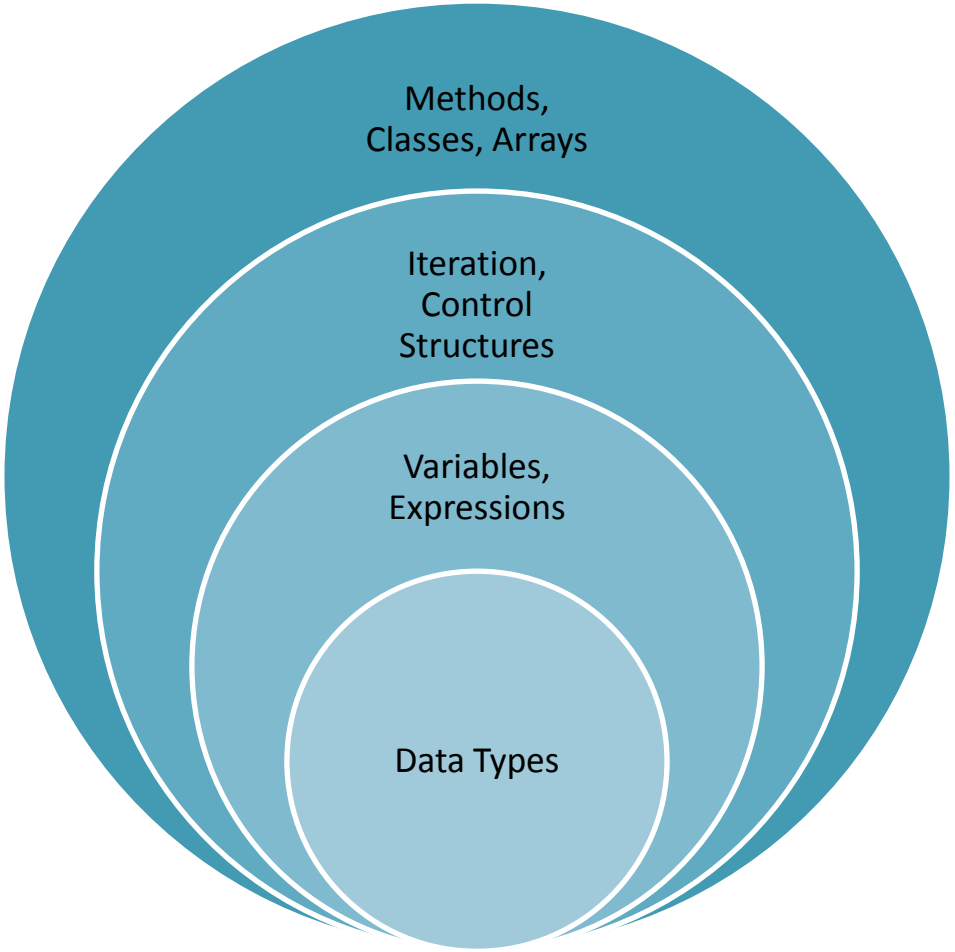
Introduction to Computing and Java

Programming Coverage



Course Design

- Instead
  - Lecture not a rehash of the book but covers same concepts from a different perspective
  - Lots of programming activities

Intro to Computing

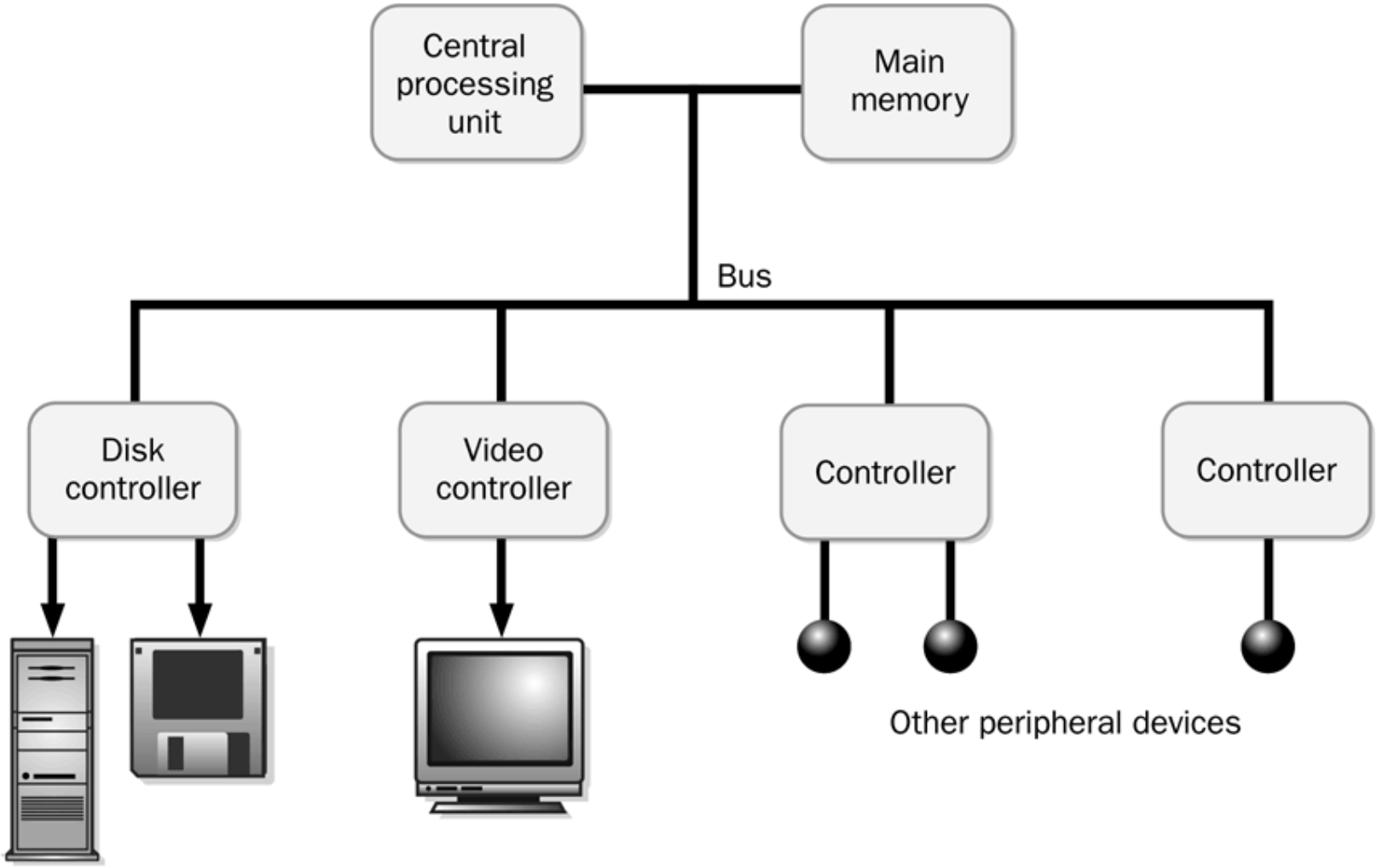


FIGURE 1.9 Basic computer architecture

The CPU

- Fetches instructions from main memory
- Carries out the operations commanded by the instructions
- Each instruction produces some outcome
- A *program* is an entire sequence of instructions
- Instructions are stored as *binary numbers*
- *Binary number* - a sequence of 1's and 0's

Main Memory – a big list of addresses

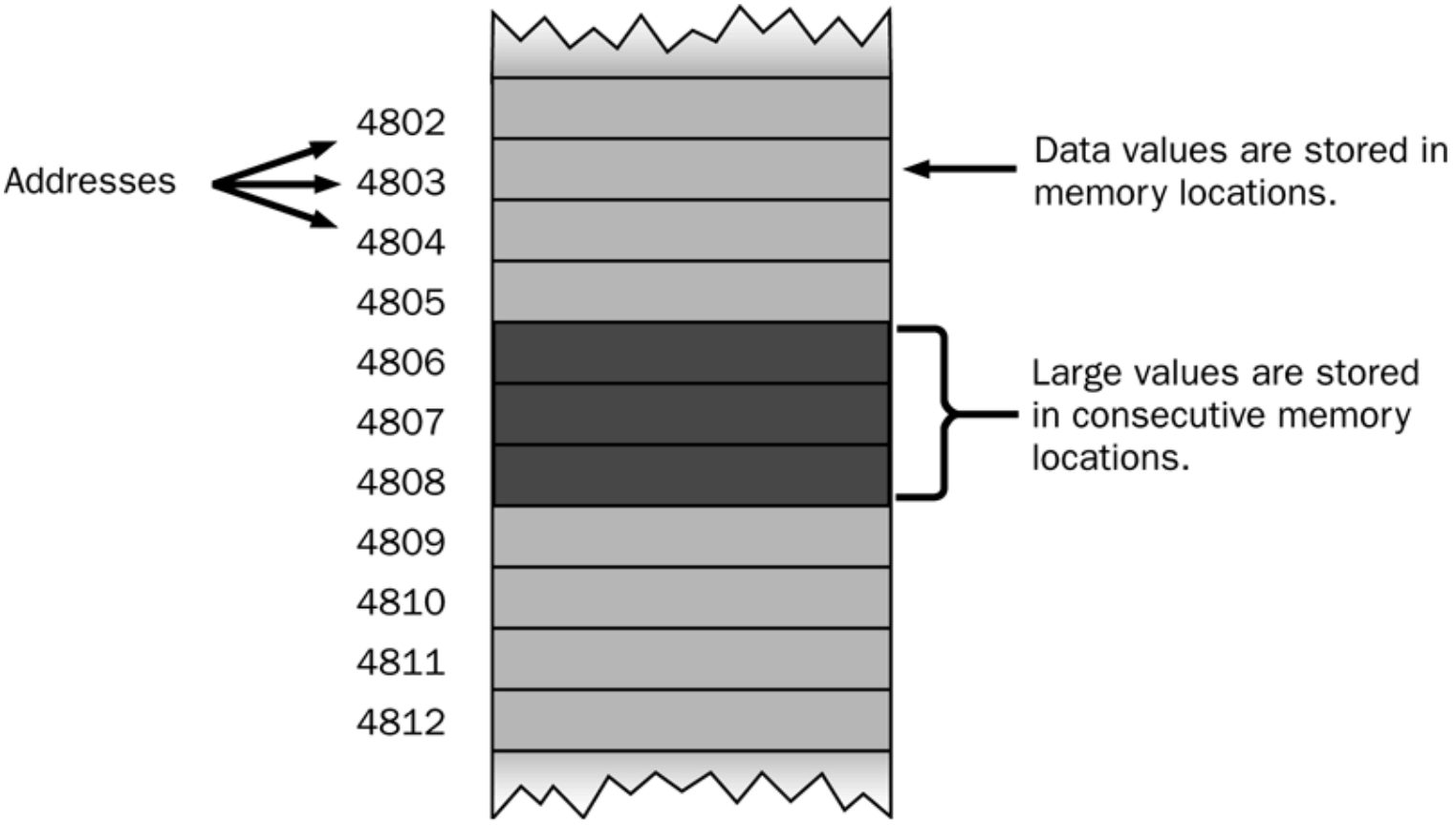


FIGURE 1.10 Memory locations

Knowing About: Computer Hardware

- **Bit**: smallest and most basic data item in a computer; represents a 0 or a 1
- **Byte**: a grouping of eight bits
  - E.g., 00010001
- **Word**: a grouping of one or more bytes

1 bit 2 items	2 bits 4 items	3 bits 8 items	4 bits 16 items	5 bits 32 items	
0	00	000	0000	00000	10000
1	01	001	0001	00001	10001
	10	010	0010	00010	10010
	11	011	0011	00011	10011
		100	0100	00100	10100
		101	0101	00101	10101
		110	0110	00110	10110
		111	0111	00111	10111
			1000	01000	11000
			1001	01001	11001
			1010	01010	11010
			1011	01011	11011
			1100	01100	11100
			1101	01101	11101
			1110	01110	11110
			1111	01111	11111

FIGURE 1.7 The number of bits used determines the number of items that can be represented

Patterns of bits could represent integer numbers

Unit	Symbol	Number of Bytes
byte		$2^0 = 1$
kilobyte	KB	$2^{10} = 1024$
megabyte	MB	$2^{20} = 1{,}048{,}576$
gigabyte	GB	$2^{30} = 1{,}073{,}741{,}824$
terabyte	TB	$2^{40} = 1{,}099{,}511{,}627{,}776$

FIGURE 1.11 Units of binary storage

Bits could represent characters

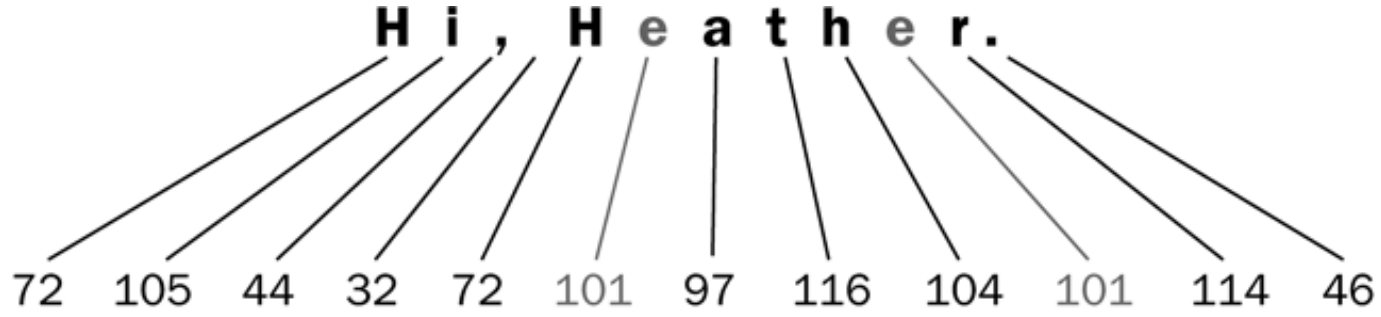


FIGURE 1.5 Text is stored by mapping each character to a number

Bits could represent sound

We said that 00010001 could represent anything, a number, sound, color, etc.

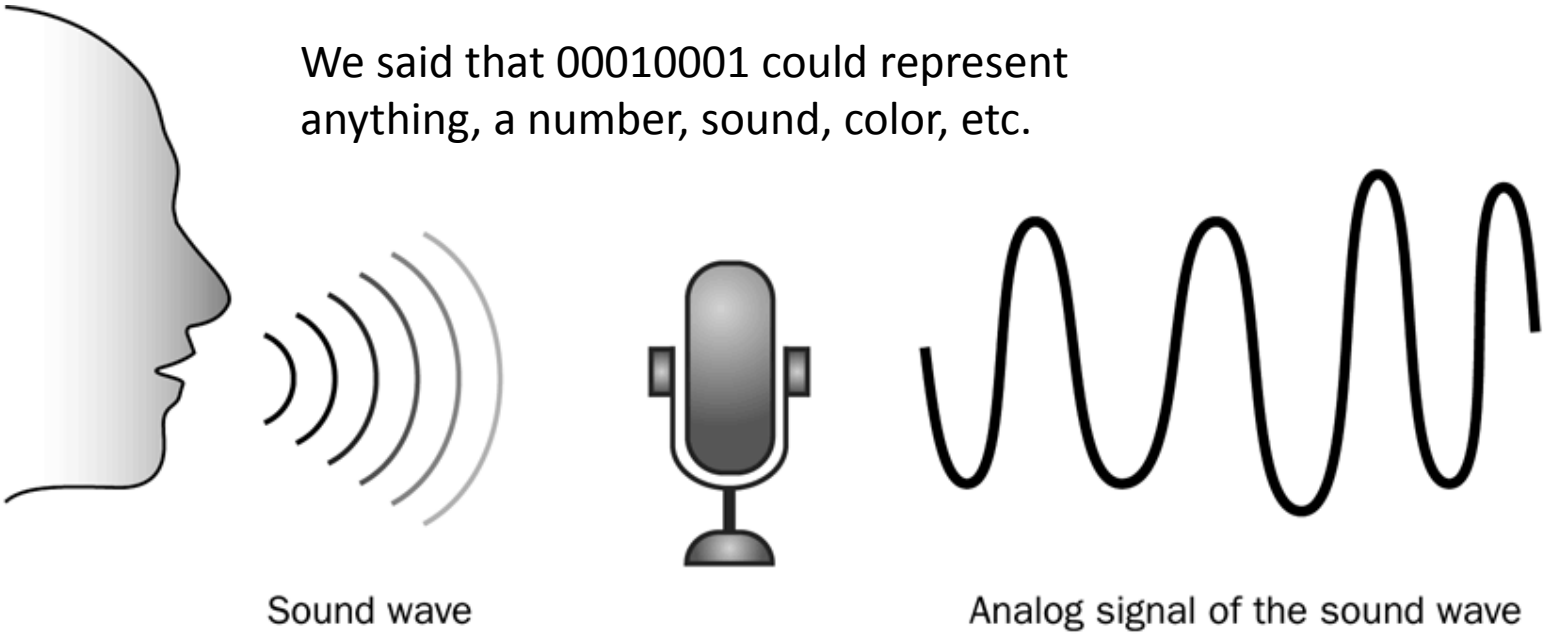


FIGURE 1.3 A sound wave and an electronic analog signal that represents the wave

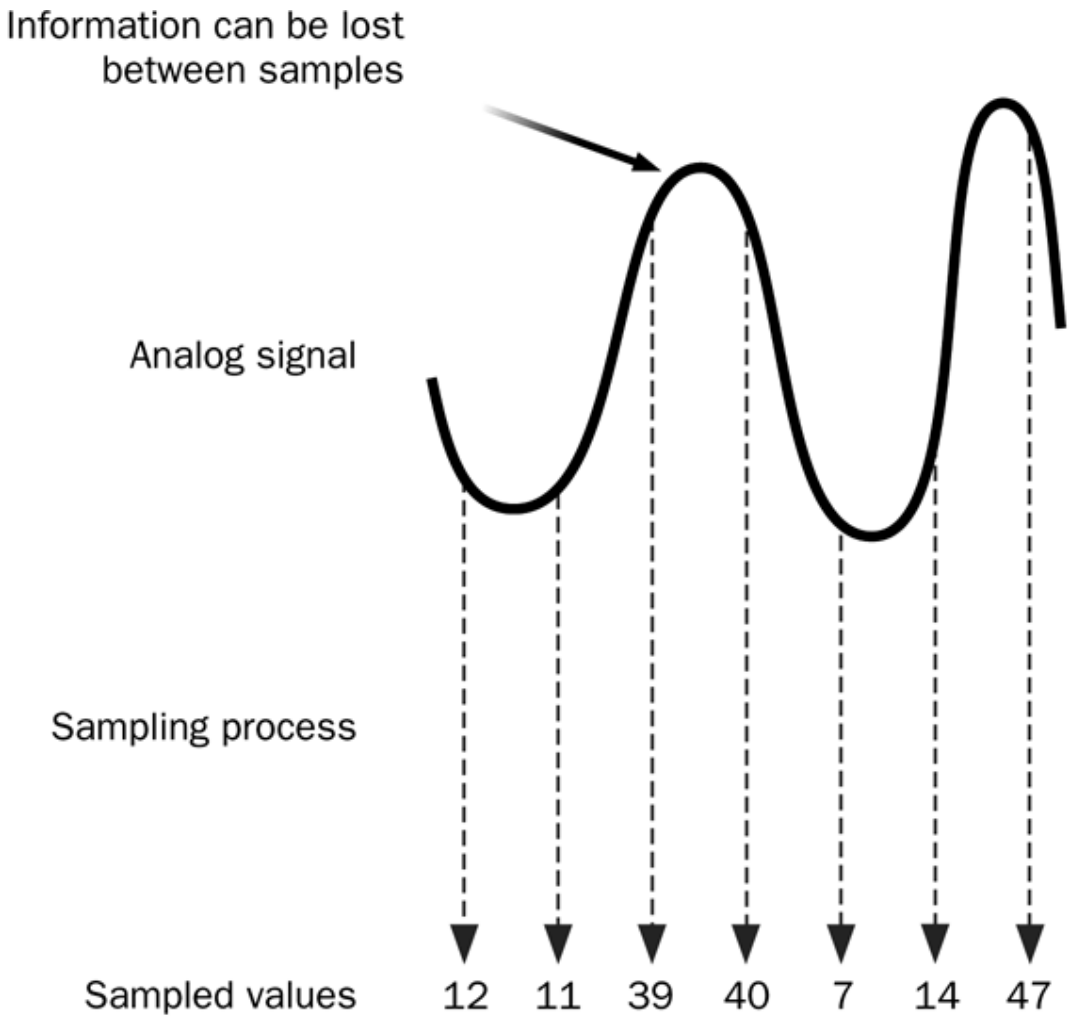
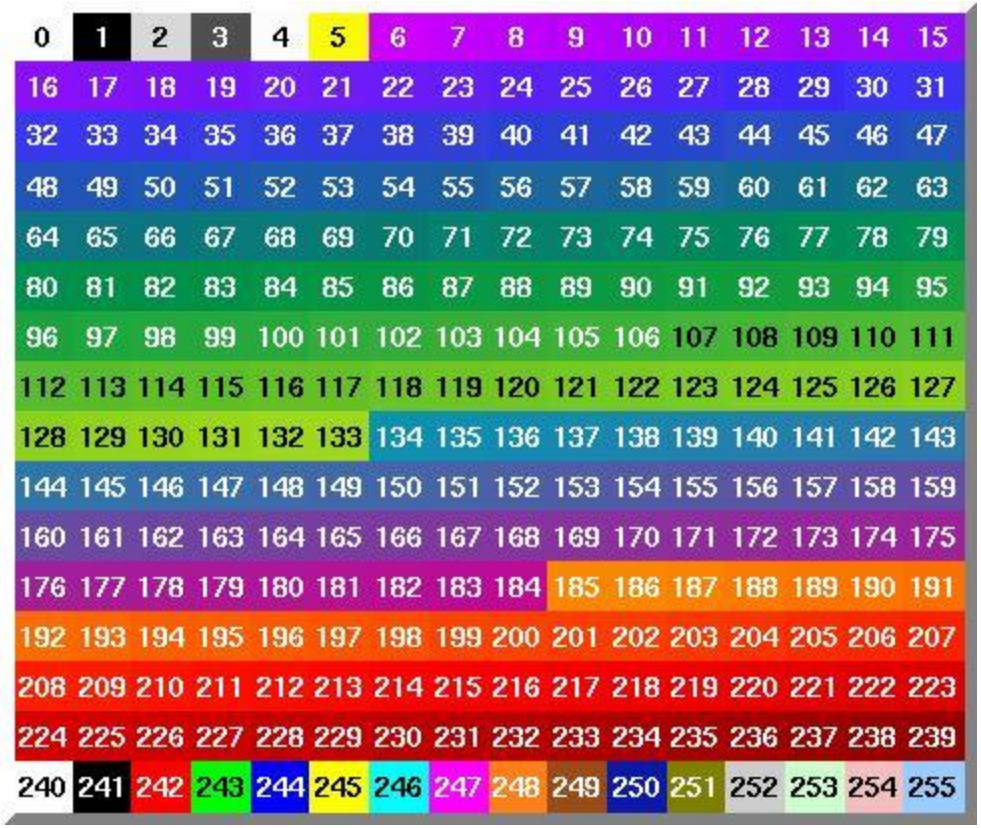


FIGURE 1.4 Digitizing an analog signal by sampling

Bits can represent colors



Bits can represent instructions

- 110110
  - might be the instruction to add two numbers
- 110100
  - might be the instruction to increment a number
- Called binary code
- Assembly Code - Mnemonics

```
Loop:  L.D      F0, 0(R1)
       ADD.D    F4, F0, F2
       S.D      0(R1), F4      ; Drop DADDUI and BNEZ
       L.D      F6, -8(R1)
       ADD.D    F8, F6, F2
       S.D      -8(R1), F8     ; Drop DADDUI and BNEZ
       L.D      F10, -16(R1)
       ADD.D    F12, F10, F2
```

The Fetch-Decode Execute Cycle

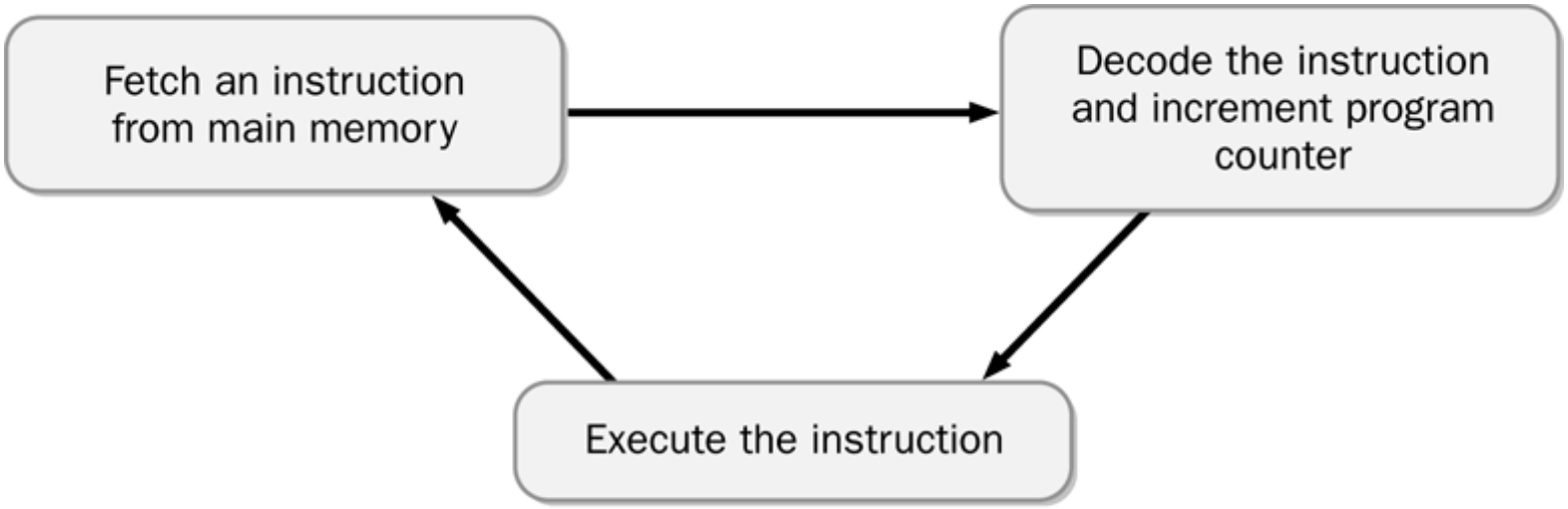


FIGURE 1.14 The continuous fetch-decode-execute cycle

Layers of Programming Languages

High-Level Language	Assembly Language	Machine Language

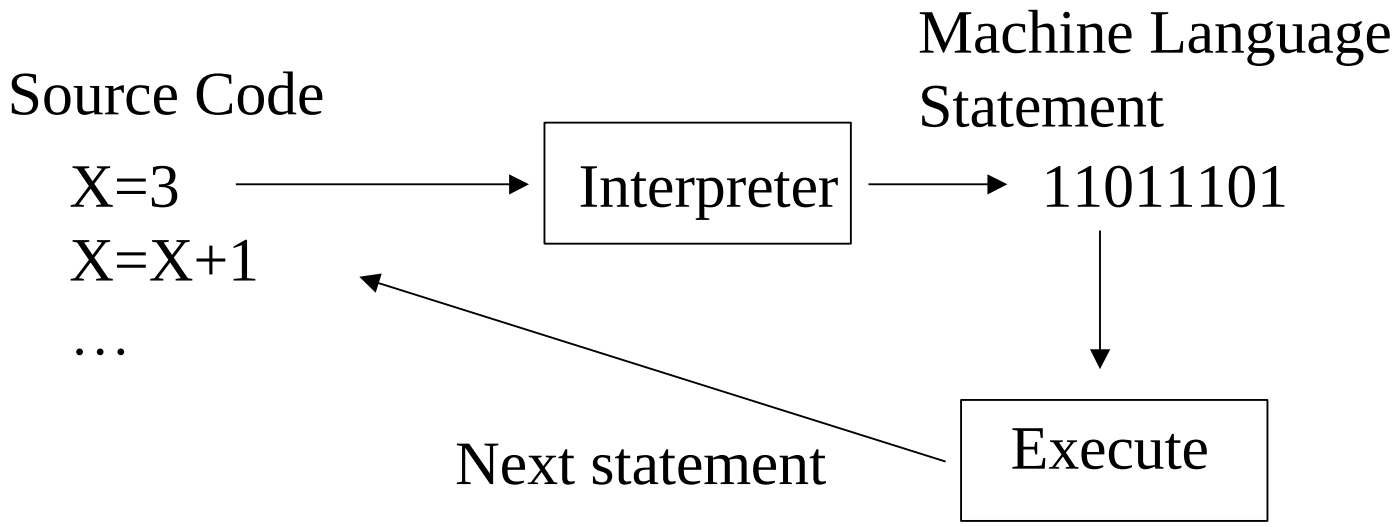
a + b	ld [%fp-20], %o0
	ld [%fp-24], %o1
	add %o0, %o1, %o0
		...
		1101 0000 0000 0111
		1011 1111 1110 1000
		1101 0010 0000 0111
		1011 1111 1110 1000
		1001 0000 0000 0000
		...

FIGURE 1.21 A high-level expression and its assembly language and machine language equivalent

A program called a **compiler** translates from high-level to machine language

Interpreter

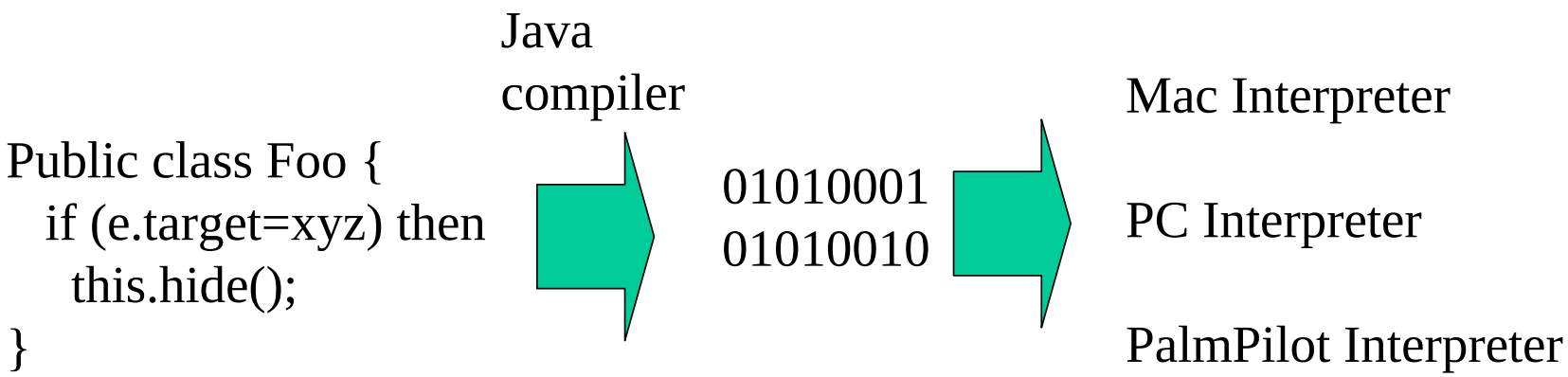
- Compiling combined with execution



Often easier to program, debug, but will run slower than compiled programs

Java – Both Interpreted/Compiled

- Somewhat of a simplification with JIT compilers



Programming

- A program is a list of instructions for the computer to follow
- Algorithm
  - Sequence of steps to solve a problem
  - Example: Searching a list of names for a number
    1. Atto, Tom (6-1102)
    2. Attrick, Jerry (6-9089)
    3. DeBanque, Robin (6-0022)
    4. Dente, Al (6-8722)
    5. Fresco, Al (6-8723)
    6. Guini, Lynn (6-8834)
    7. Oki, Kerry (6-9213)
    8. Wright, Eaton (6-4441)

Pseudocode

- Somewhere between English and actual code to help figure out how to write the actual code
- Binary search pseudocode
  - Given a list of names
    - If the list is empty then target not found
    - Otherwise:
      - Get the name in the middle of the list
      - If this name is the same as the target, then the target is in the list
      - If this name is alphabetically before the target then
        - » Repeat the process on the bottom half of the list
      - If this name is alphabetically after the target then
        - » Repeat the process on the first half of the list

Java Example

- In-class: Entering and running a “Hello, World” program using Textpad

File: HelloWorld.java

```
/*
 *  Normally you would put your name and assignment info here
 *  This program prints out "Hello, World".
 */
public class HelloWorld
{
   public static void main(String[] args)
   {
      System.out.println("Hello, world!");
   }
}
```