Chapter 1: Overview of Computers and Programming

Problem Solving and Program Design in C 5th Edition

by Jeri R. Hanly and Elliot B. Koffman

1) What is a computer?

- **A computer is a device capable of performing computations and making logical decisions at a speed of millions, and even billions of times faster than human beings.**

- **Computers process *data* under the control of sets of instructions called *computer programs*.**

- **The various devices (such as keyboard, screen, disks, memory, and processing units) that comprise a computer systems are referred to as *hardware*.**

- **The computer programs that run on a computer are called *software*.**

Components of a computer:

- **i. *Input unit*: This is the "receiving" section of the computer (mostly keyboard).**
- **ii. *Output unit*: This is the "shipping" section of the computer (CRT screen).**
- **iii. *Memory unit*: This is the rapid access, relatively low-capacity "warehouse" section of the computer. It is often called main/primary memory. (board)**
- **iv. *Arithmetic and logic unit (ALU)*: This is the "manufacturing" section of the computer. It is responsible for all the calculations and decision-mechanisms.**
- **v. *Control unit (CU)*: This is the "administrative" section of the computer. It is the computer's coordinator and is responsible for supervising the operation of the other sections.**
- ***Central Processing unit (CPU) = ALU + CU***
- **vi. *Secondary storage unit*: This is the long-term, high-capacity "warehouse" section of the computer. Programs or data that are not currently active are placed here (disks).**

Human analogy:

- **(hearing=input, telling=output, thinking=cpu, keeping-in-mind=main memory, permanent storage (file/cabinet)=secondary storage)**
- **Hard to match with humans though. In the pursuit of matching human brain, computers are having more than one processor.**
- **personal computers typically have one processor**
- **university computer, Strauss has 16 CPU's, 32GB memory.**
- **Input=keyboards, outputs=CRT screens, printers**

2) Can a computer think?

- **Computers can think what they are programmed to think. They are not "intelligent enough" (if interested, there are courses on AI, Robotics, Computer Vision, etc.)**

Figure 1.1 **The Intel Pentium 4 Processor chip is an integrated circuit containing the full circuitry of a central processing unit. This processor can execute a simple instruction such as an integer addition in one six-billionth of a second.** **(Reprinted by permission of Intel Corporation, © Intel Corporation 2003)**

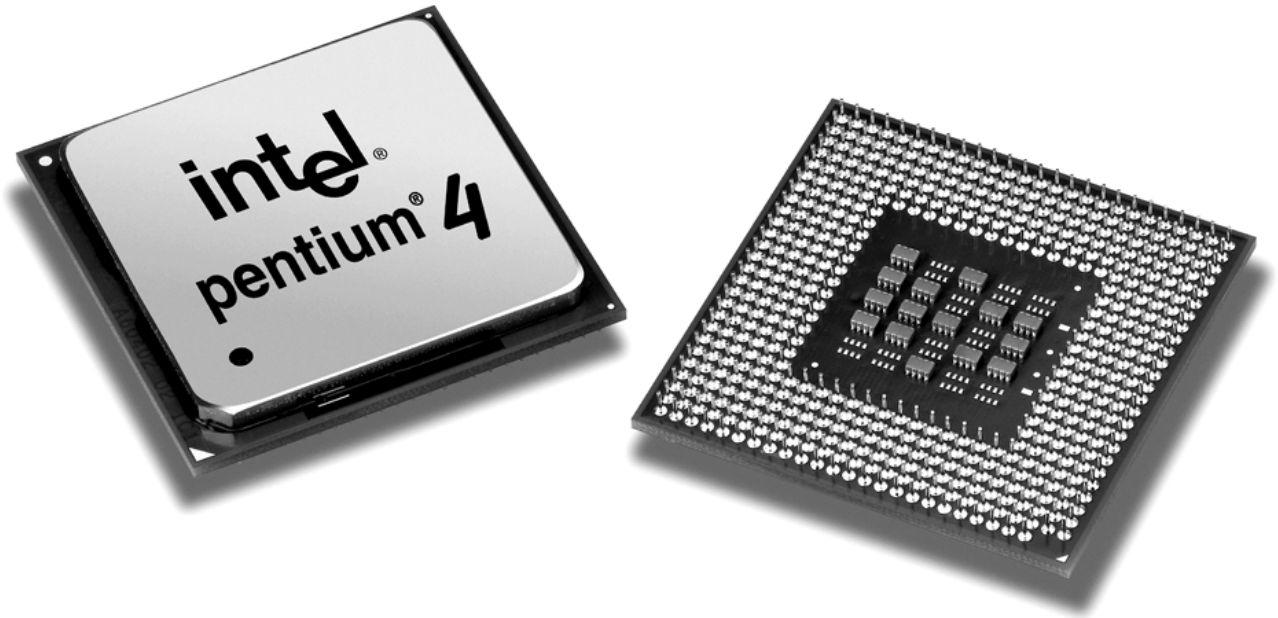
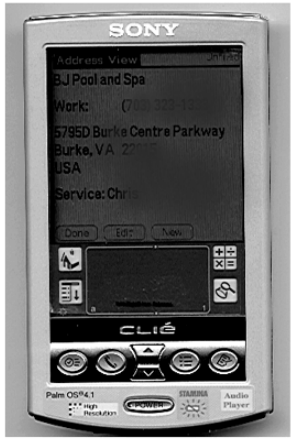


Figure 1.2
(a) Notebook Computer (ThinkPad®, Courtesy of IBM).
(b) Palmtop Computer (Sony Clié PDA ®, Courtesy of Sony).
(c) Desktop Computer (IBM NetVista Desktop, Courtesy of IBM).

(a)



(b)

(c)

Figure 1.3 **Components of a Computer**

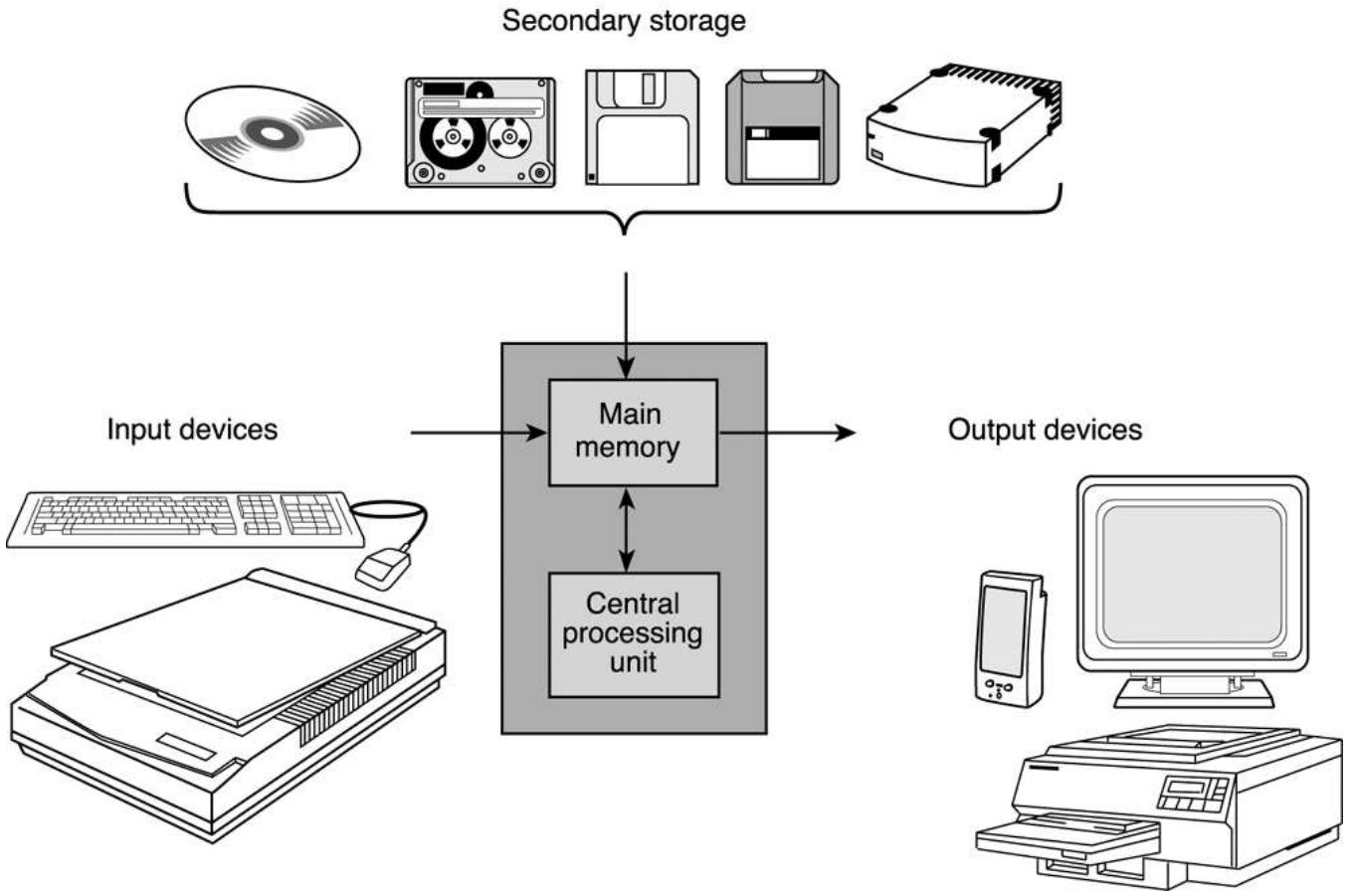


Figure 1.8 **Local Area Network**

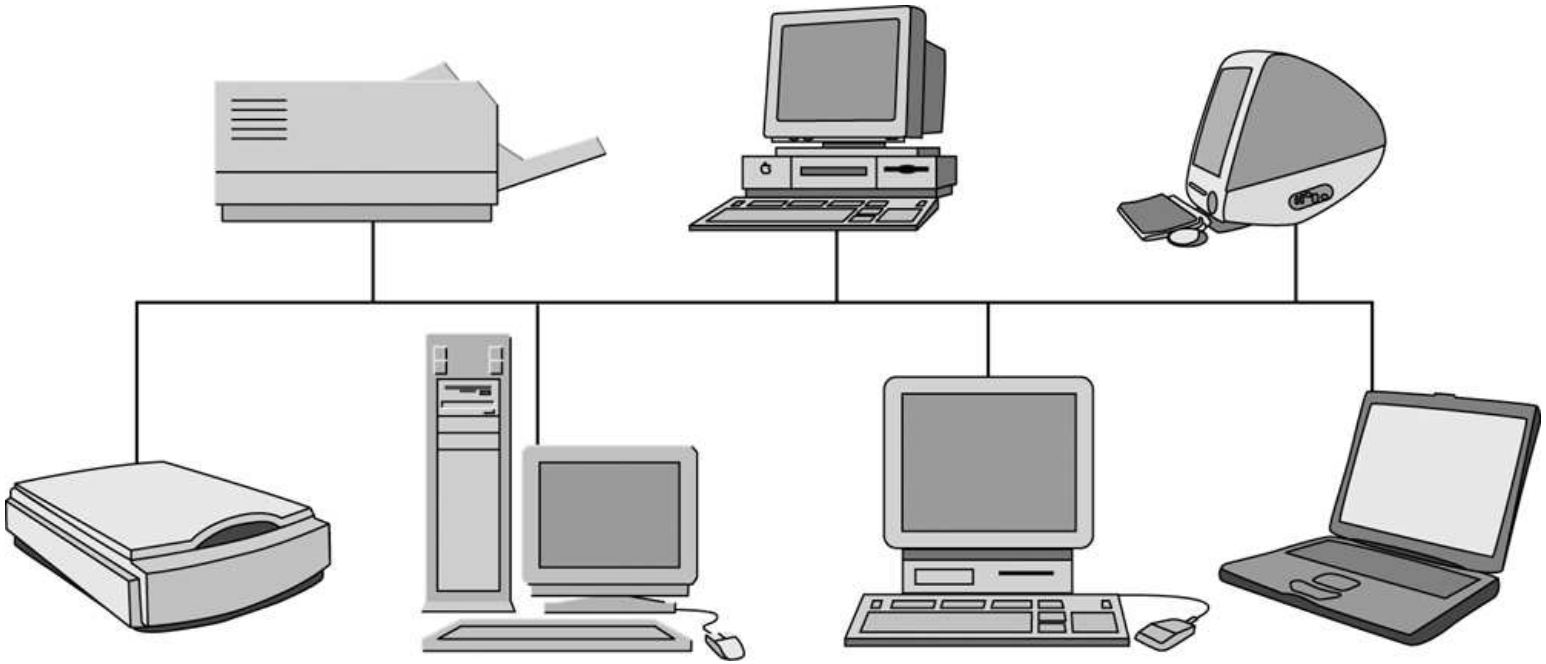
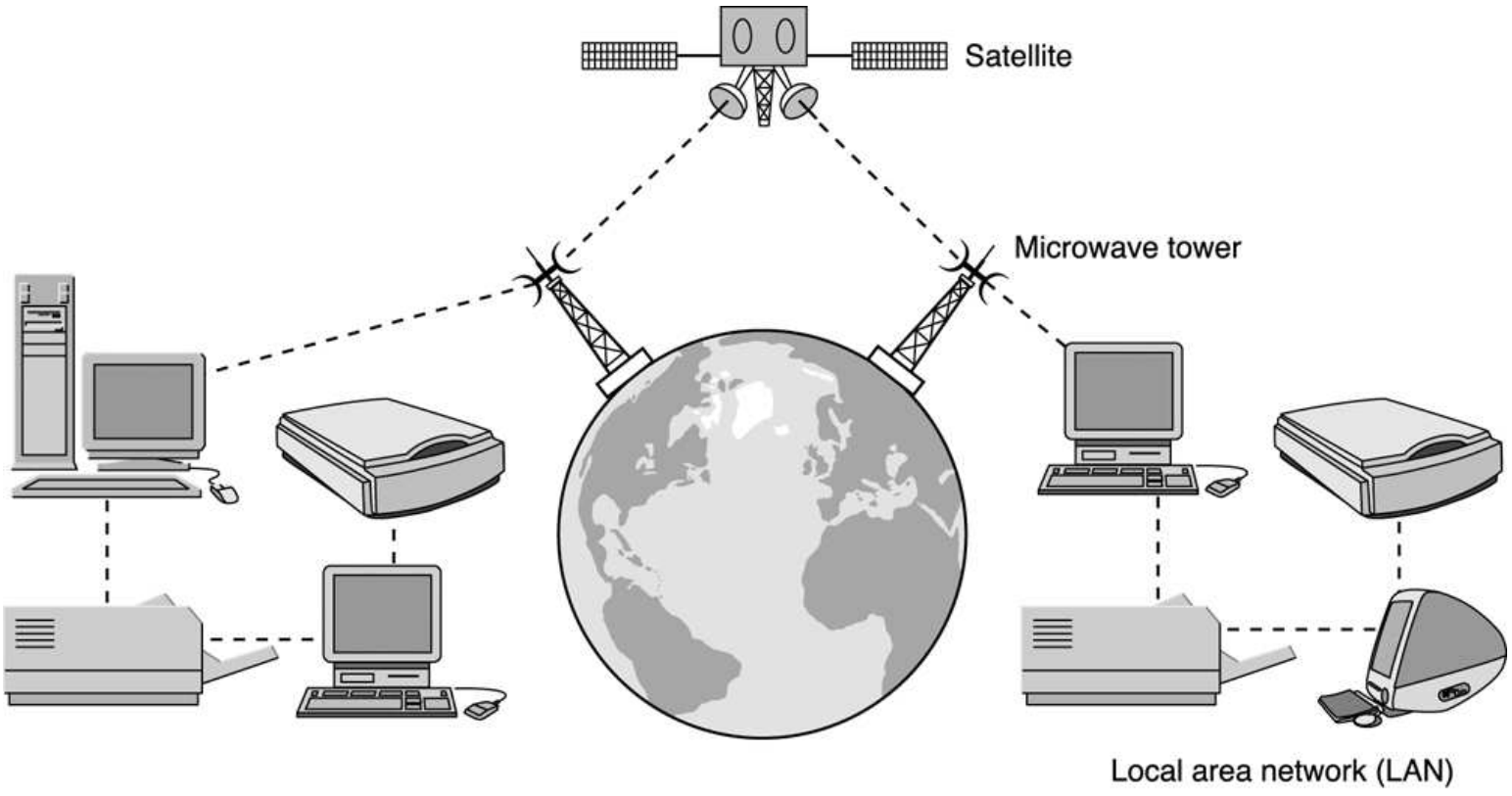


Figure 1.9 **A Wide Area Network with Satellite Relays of Microwave Signals**



3) How does a computer work?

- **Someone has to tell it what to do (program) + what to do it on (data).**

Simplified view of the operation of a computer:

- **1. A programmer (you) writes a program + creates data.**
- **2. Program + data is typed, and will get stored on disk.**
- **3. Programmer commands the computer to "execute" on the specified data.**

- **CU copies program+data into memory, keeps track of current instruction to perform (sets PC to point to 1st instruction in program) (note: CU=control unit, pc=program counter)**

- **-Fetch instr. at PC from memory + store at small location in CPU.**
- **-Decode instr.**
- **-Fetch data/operands**
- **-Execute instructions (by ALU) (note: ALU=Arithmetic and Logic Unit)**
- **-Store result to memory**
- **-Increment PC to next instr.**
- **-repeat above steps until PC runs out of program.**

Some puzzling questions:

- **1. How do computers actually store programs and data? 0's and 1's (binary)**
- **2. How does a program get into this form? (compilers convert high-level language into low-level)**
- **3. What tells computer to take what you type and put on the disk, or how does it organize all the information it stores (Operating System, UNIX in our case) (board)**

- **The computer stores programs or data in terms of 0's and 1's. 0 or 1 is represented by electrical circuits ("on" or "off", or, "pulse" or "no-pulse"). Hence, everything is encoded in patterns of 0 and 1.**

- **For example, 5 is 0101, 8 is 1000 and so on.**

Figure 1.4
1000 Memory Cells in Main Memory

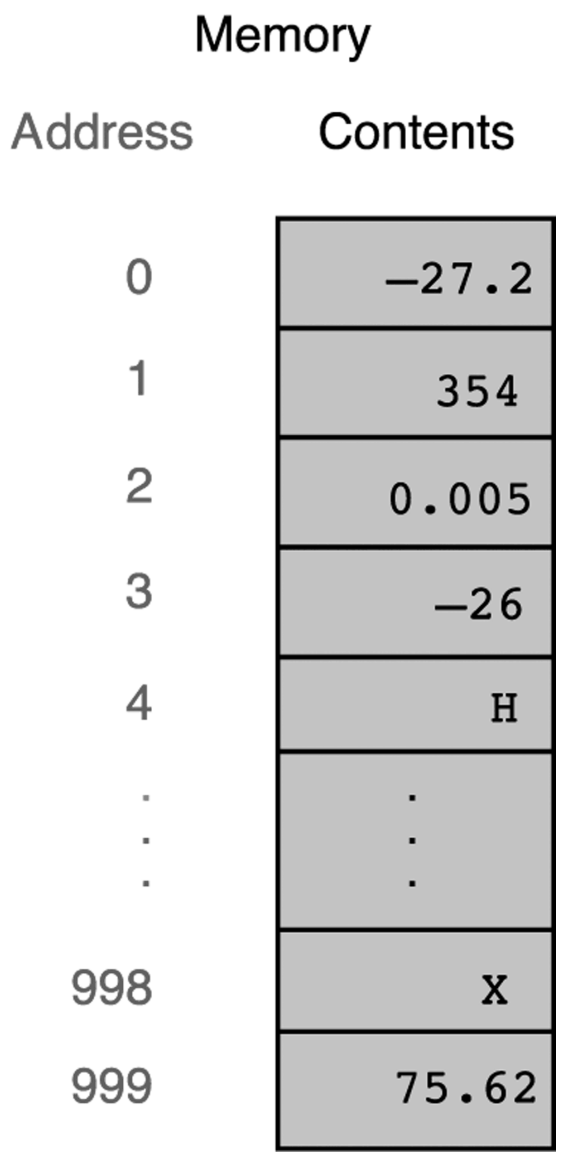


Figure 1.5 **Relationship Between a Byte and a Bit**

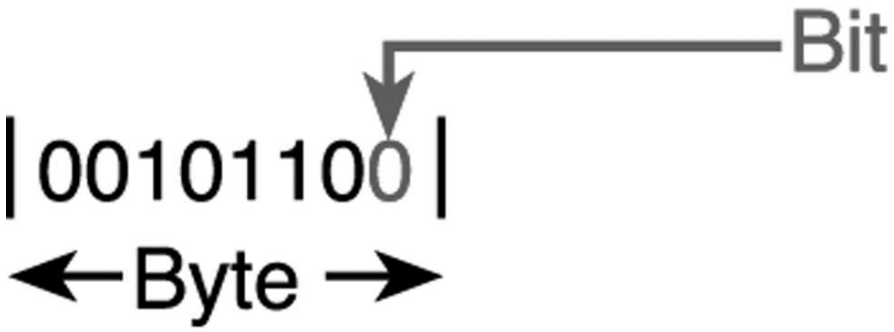


Figure 1.10 **Entering a UNIX Command for Directory Display**

```
1. mycomputer:~> ls temp/misc
2. Gridvar.c     Gridvar.exe   Gridok.dat
3.
4. mycomputer:~>
```

(Board)

Figure 1.11 **Accessing Disk Drive through Windows**

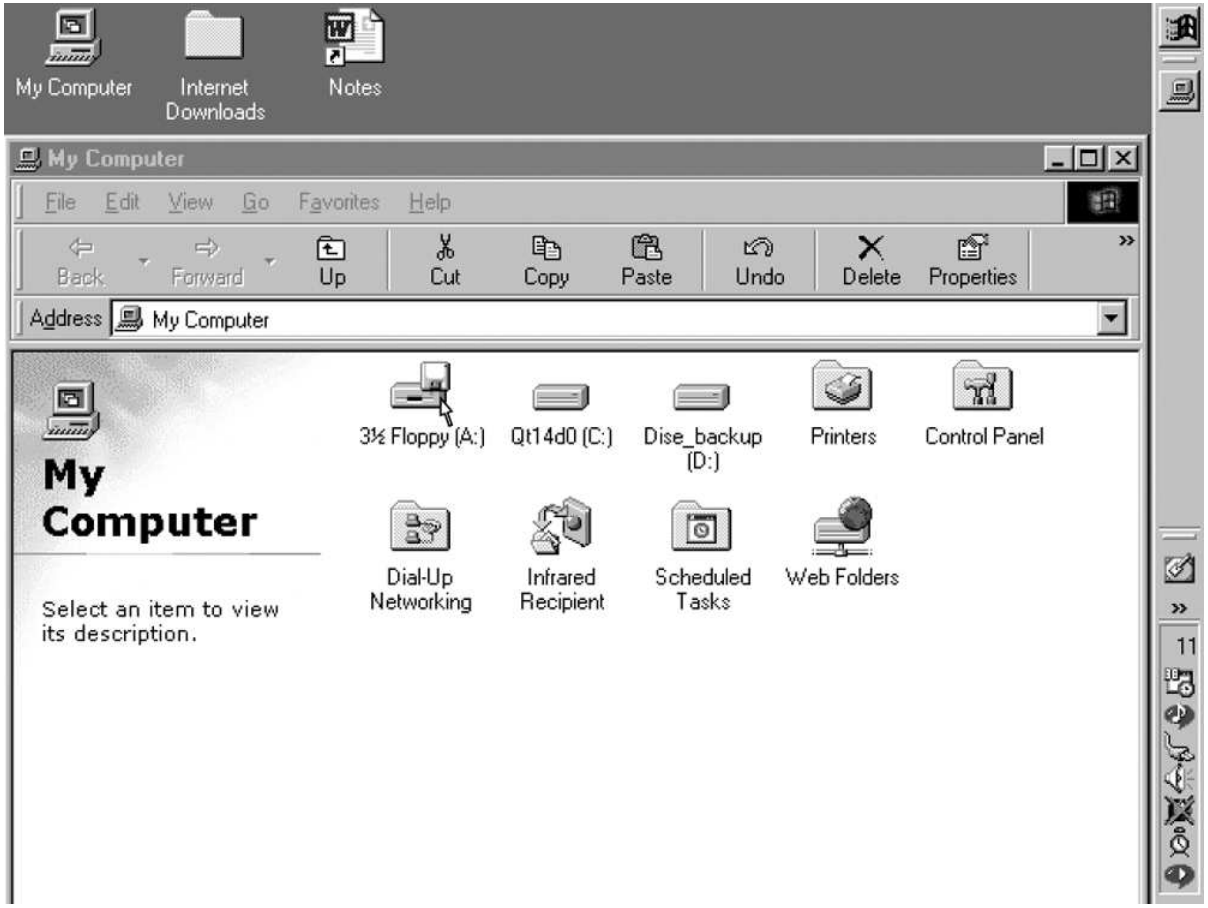


Figure 1.12 **Entering, Translating, and Running a High-Level Language Program**

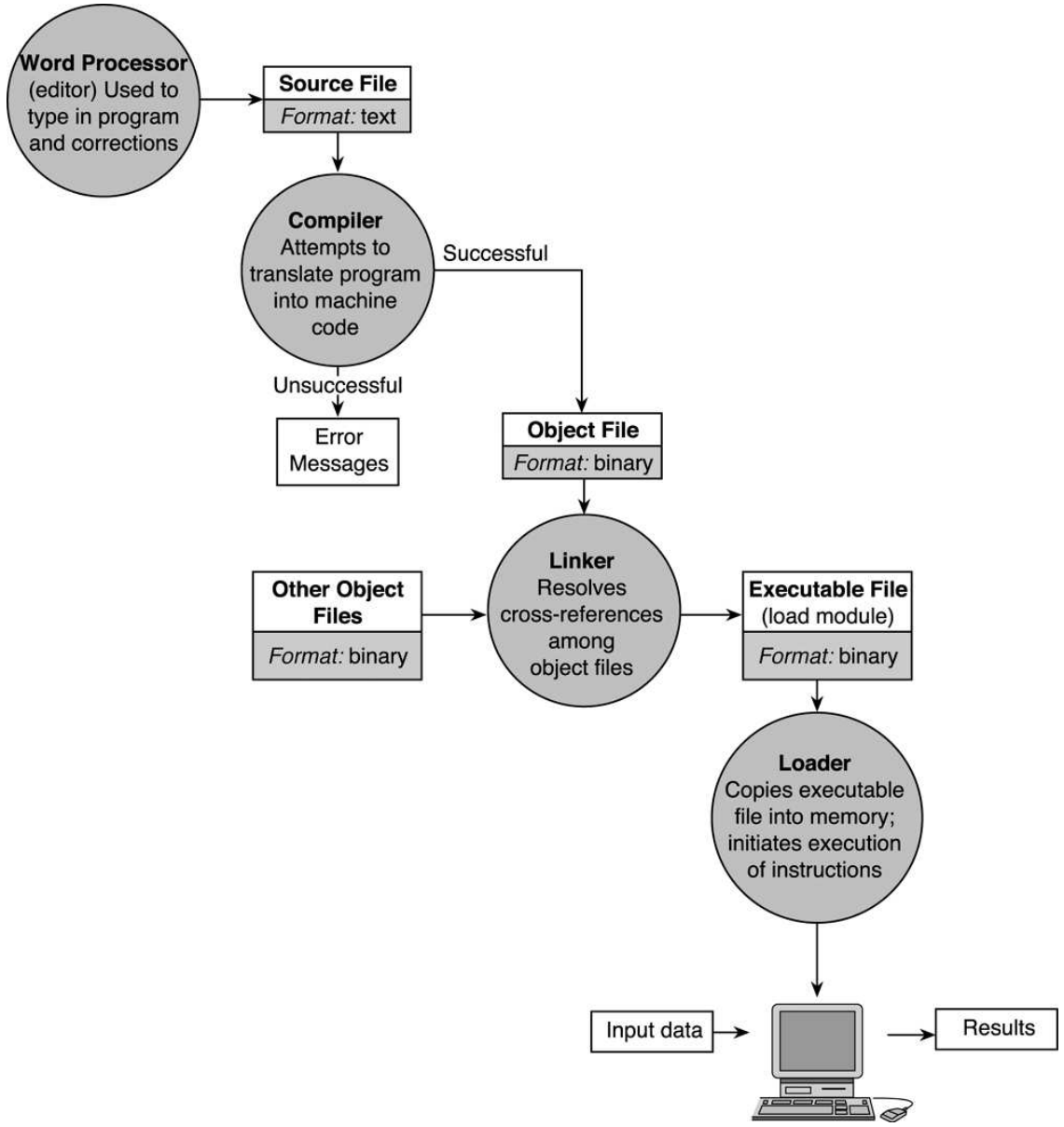


Figure 1.13 **Flow of Information During Program Execution**

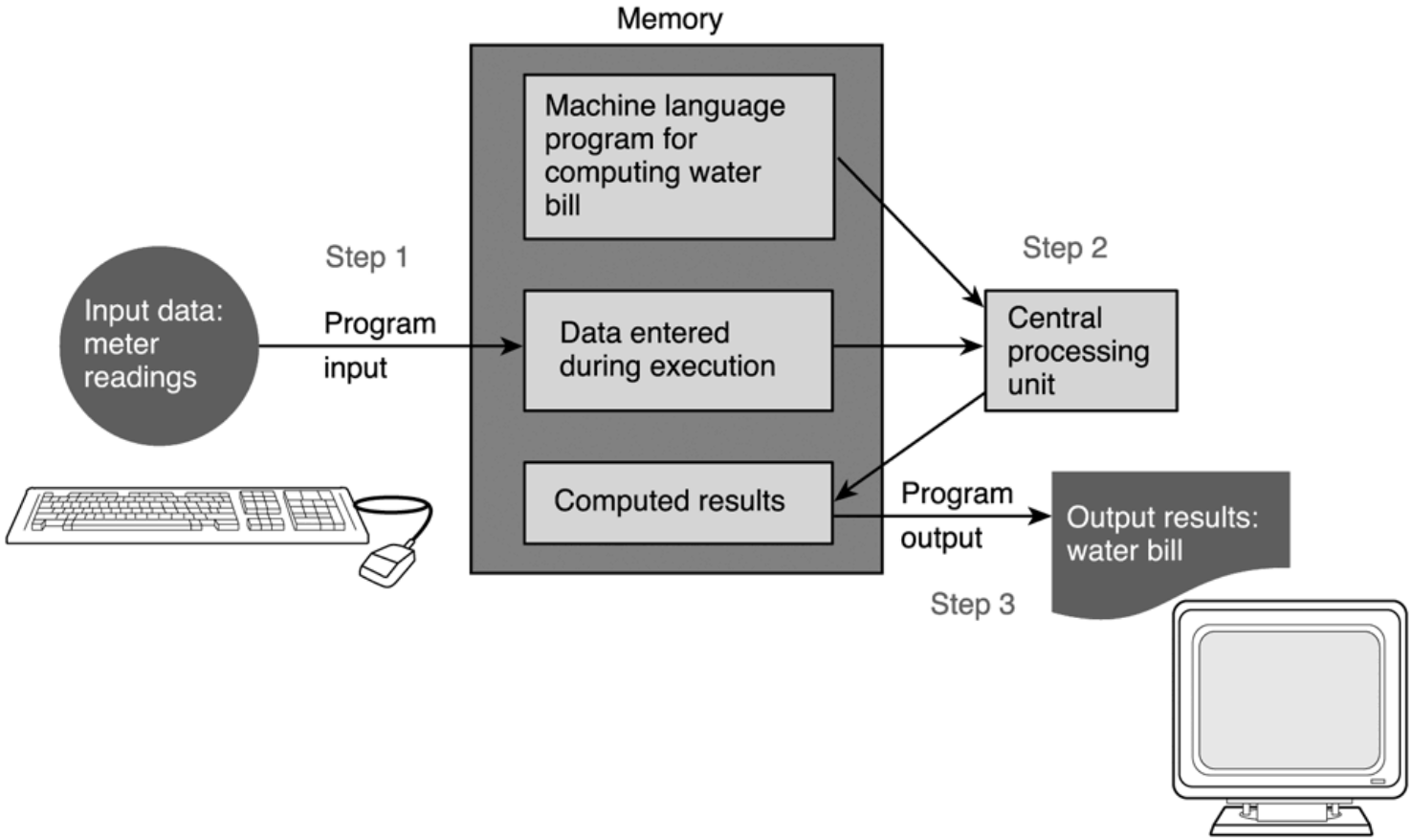


Figure 1.14 **Miles-to-Kilometers Conversion Program** (board+shell)

```
1.  /*
2.   * Converts distance in miles to kilometers.
3.   */
4.  #include <stdio.h>              /* printf, scanf definitions */
5.  #define KMS_PER_MILE 1.609      /* conversion constant       */
6.
7.  int
8.  main(void)
9.  {
10.       double miles,  /* input - distance in miles.       */
11.              kms;    /* output - distance in kilometers  */
12.
13.       /* Get the distance in miles. */
14.       printf("Enter the distance in miles> ");
15.       scanf("%lf", &miles);
16.
17.       /* Convert the distance to kilometers. */
18.       kms = KMS_PER_MILE * miles;
19.
20.       /* Display the distance in kilometers. */
21.       printf("That equals %f kilometers.\n", kms);
22.
23.       return (0);
24. }
```

Sample Run

```
Enter the distance in miles> 10.00
That equals 16.090000 kilometers.
```