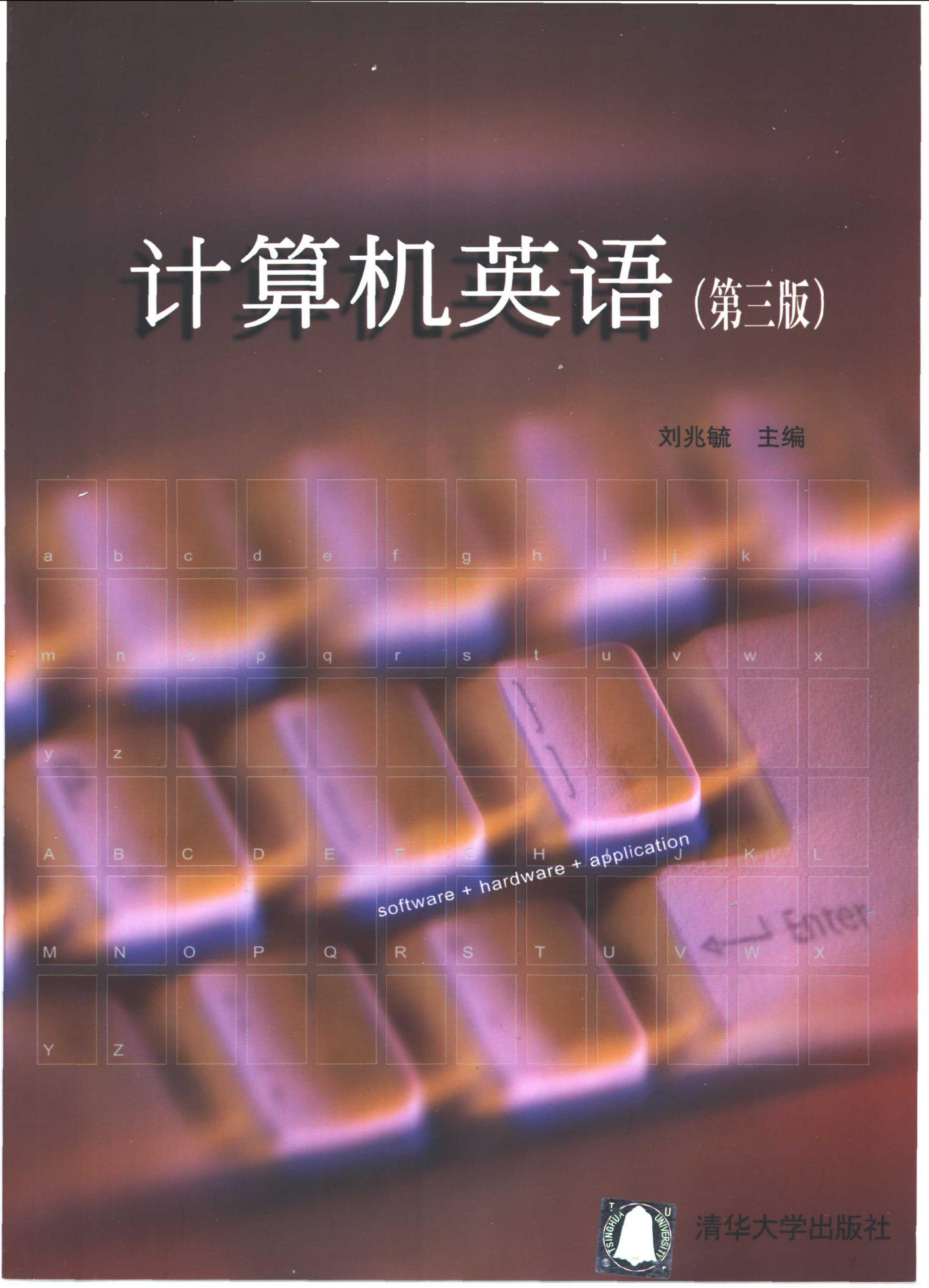
计算机英语（第三版）

刘兆毓　主编



清华大学出版社

计算机英语（第三版）

刘兆毓　主编

清华大学出版社
北京

内 容 简 介

本书是用英文编写的、按计算机知识结构的层次组成的计算机技术教材，书中内容涉及计算机的技术基础、系统和应用等各个方面，其中包括反映当前最新技术和应用的内容。全书由三部分组成：计算机硬件、计算机软件和计算机应用。

书中对一些较难翻译和理解的句子和单词进行了注释；每一节后面列出关键词汇，给出练习题；每一章后面还列出反映最新技术的一篇阅读材料以及相关词汇；以便提高读者阅读计算机英文文献的水平。书后附有各节习题答案及译文，供读者参考。本书可供大专院校学生使用，也可供参加计算机水平考试的考生和广大工程技术人员参考。

图书在版编目(CIP)数据

计算机英语/刘兆毓主编．—3版．—北京：清华大学出版社，2003

ISBN 7-302-06982-4

Ⅰ．计…　Ⅱ．刘…　Ⅲ．电子计算机—英语　Ⅳ．H31

中国版本图书馆 CIP 数据核字(2003)第 063542 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦

http://www.tup.com.cn　　**邮　　编**：100084

社 总 机：010-62770175　　**客户服务**：010-62776969

组稿编辑：尹芳平

文稿编辑：刘　彤

封面设计：常雪影

印 刷 者：北京市清华园胶印厂

发 行 者：新华书店总店北京发行所

开　　本：185×260　**印张**：25.25　**字数**：781 千字

版　　次：2003 年 7 月第 3 版　2003 年 8 月第 3 次印刷

书　　号：ISBN 7-302-06982-4/TP・5152

印　　数：20001～40000

定　　价：35.00 元

第三版前言

本书是1997年再版的《计算机英语》的第三版。由于计算机技术飞速发展，新课题、新内容不断涌现，因此本书只保留第二版中的个别章节，其余部分均为新材料。为了帮助读者学习，本书在每篇课文之后都安排了一定数量的练习题。

本书仍由刘兆毓主编。全书共分18章；北京印刷学院的郑家农编写第10章～第13章和第17章；施家元编写第14章和第15章；闫金平编写第9章和第18章；郭庆云编写第4章和第5章；南京工程学院的武华编写第6章和第7章，其余由刘兆毓编写。

由于作者水平有限，书中难免有不当之处，敬请读者批评指正。

编　者

2002.10 北京

CONTENTS

参考译文

PART I COMPUTER HARDWARE

CHAPTER 1 PRINCIPLES OF COMPUTER ORGANIZATION

1.1 COMPUTER HARDWARE

We build computer to solve problems. Early computer solved mathematical and engineering problems, and later computers emphasized information processing for business applications. Today, computers also control machines as diverse as automobile engines, robots, and microwave ovens. A computer system solves a problem from any of these domains by accepting input, processing it, and producing output. Fig. 1-1 illustrates the function of a computer system.

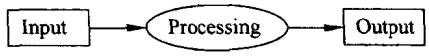


Fig. 1-1 The three activities of a computer system

Computer systems consist of hardware and software. *Hardware* is the physical part of the system. Once designed, hardware is difficult and expensive to change. *Software* is the set of programs that instruct the hardware and is easier to modify than hardware. Computers are valuable because they are general-purpose machines that can solve many different kinds of problems, as opposed to special-purpose machines that can each solve only one kind of problem. Different problems can be solved with the same hardware by supplying the system with a different set of instructions, that is, with different software.

Every computer has four basic hardware components:

- Input devices.
- Output devices.
- Main memory.
- Central processing unit(CPU).

Fig. 1-2 shows these components in a block diagram. The lines between the blocks represent the flow of information flows from one component to another on the *bus*, which is simply a group of wires connecting the components.[1] Processing occurs in the CPU and main memory. The organization in Fig. 1-2, with the components connected to each other by the bus, is common. However, other configurations are possible as well.

Computer hardware is often classified by its relative physical size:

- *Small* microcomputer;
- *Medium* minicomputer;

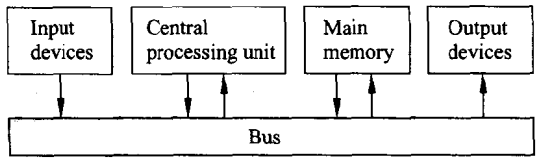


Fig. 1-2 Block diagram of the four components of a computer system

• *Large* mainframe.

Just the CPU of a mainframe often occupies an entire cabinet. Its input/output (I/O) devices and memory might fill an entire room. Microcomputers can be small enough to fit on a desk or in a briefcase. As technology advances, the amount of processing previously possible only on large machines becomes possible on smaller machines. Microcomputers now can do much of the work that only minicomputers or mainframes could do in the past.

The classification just described is based on physical size as opposed to storage size. A computer system user is generally more concerned with storage size, because that is a more direct indication of the amount of useful work that the hardware can perform. Speed of computation is another characteristic that is important to the user. Generally speaking, users want a fast CPU and large amounts of storage, but a physically small machine for the I/O devices and main memory.

When computer scientists study problems, therefore, they are concerned with space and time-the space necessary inside a computer system to store a problem and the time required to solve it. They commonly use the metric prefixes of Table 1-1 to express large or small quantities of space or time.

Table 1-1 Prefixes for powers of 10

Multiple	Prefix	Abbrev
10^{9}	giga-	G
10^{6}	mega-	M
10^{3}	kilo-	K
10^{-3}	milli-	m
10^{-6}	micro-	μ
10^{-9}	nano-	n

Example Suppose it takes 4.5 microseconds, also written 4.5 μs, to transfer some information across the bus from one component to another. (a) How many seconds are required for the transfer? (b) How many transfers can take place during one minute?[2]

(a) A time of 4.5μs is 4.5×10^{-6} from Table 1-1 or 0.000 004 5 s. (b) Because there are 60 seconds in one minute, the number of times the transfer can occur is (60 s)/(0.000 004 5 s/transfer) or 13 300 000 transfers. Note that since the original value was given with two significant figures, the result should not be given to more than two or three significant figures.

Table 1-1 shows that in the metric system the prefix kilo-is 1 000 and mega-is 1 000 000. But in computer science, a kilo-is 2^{10} or 1024. The different between 1 000 and 1 024 is less than 3%, so you can think of a computer science kilo-as being about 1 000 even though it is a little more. The same applies to mega-and giga-, as in Table 1-2. This time, the approximation is a little worse, but for mega-it is still within 5%.

Table 1-2 Computer science values of the large prefixes

Prefix	Computer science value
giga-	$2^{30}=1\ 073\ 741\ 824$
mega-	$2^{20}=1\ 048\ 576$
kilo-	$2^{10}=1\ 024$

NOTES

[1] 总线不仅仅是一组线缆,它还包括一些逻辑电路。此处是一种形象说法,有关总线的概念请见本章1.6节。

[2] 因为没有给出总线宽度,故每次传送可以理解为每秒钟或每分钟所传送的字节数、字数等。

KEYWORDS

computer	计算机	input device	输入设备
information processing	信息处理	output device	输出设备
hardware	硬件	main memory	主存储器
software	软件	central processing unit (CPU)	中央处理器
program	程序	bus	总线
general-purpose machine	通用(计算)机	microcomputer	微型计算机
special-purpose machine	专用(计算)机	minicomputer	小型计算机
instruction	指令	mainframe	主机,特大型机
set of instruction	指令集,指令系统		

EXERCISES

1. Multiple choices.

(1) When we store a problem into a computer, _____ is necessary.

a. space　　b. time

c. input device　　d. output device

(2) Early computer solved _____ problems.

a. control　　b. business applications

c. engineering　　d. mathematical

(3) We can use prefix micro to express _____.

a. time metric　　b. space metric

c. both time and space metrics　　d. 10^{-6}

(4) We can say a bus is simply _____.

a. a group of wires　　b. a wire

c. a 8-bit bus　　d. a 16-bit bus

(5) A computer system user generally more cares for _____.

a. physical size of the computer

b. storage size

c. speed of computation

d. efficiency of the computer

(6) According to the physical size of computers we can classify the computers into _____.

a. microcomputer
b. minicomputer
c. mainframe
d. supercomputer

(7) Prefix "mega" used for computer science is _____.

a. larger than 10^6
b. smaller than 10^6
c. equal to 2^{20}
d. 1 048 576

(8) The basic hardware components of any computer include _____.

a. CPU
b. main memory
c. input devices
d. output devices

2. Fill in the blank with appropriate words or phrases found behind this exercise.

(1) A computer system solves a problem by _____.

(2) The amount of effective work of a computer can be indicated by _____ directly.

(3) Computer systems consist of _____.

(4) Computer that can solve only one kind of problem is a _____.

(5) Computer that can solve many different kinds of problems is a _____.

(6) _____ instruct the hardware.

(7) _____ is difficult and expensive to change.

(8) We usually show the computer components in a _____.

a. general-purpose machine
b. hardware
c. accepting input, processing problem, and producing output
d. block diagram
e. software
f. storage size
g. special-purpose machine
h. hardware and software

1.2 COMPUTER CODES

Although the capacity of their storage locations can vary, every computer stores numbers, letters, and other characters in a coded form.

Every character in storage is represented by a string of 0s and 1s—the only digits found in the binary numbering system. Let's see how it's possible to use just two digits to represent any character.

Most computers use a coded version of true binary to represent decimal numbers.

Although many coding schemes have been developed over the years, the most popular of these use a binary coded decimal (BCD) approach.

The BCD Approach.

With BCD, it's possible to convert each decimal number into its binary equivalent rather than convert the entire decimal value into a pure binary form.

The BCD equivalent of each possible symbol is shown in Fig. 1-3.

Since 8 and 9 place value, all decimal digits are presented in BCD by 4 bits.

You've just seen that 202 is equal to 1100 1010 in a pure binary form.

Converting 202 into BCD, however, produces the following result: 202(10) in BCD = 0010 0000 0010 or 0010 0000 0010

Decimal Digit	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Fig. 1-3 BCD numeric bit configurations and their decimal equivalent

Zone bits		Numeric bits			
B	A	8	4	2	1

(a) Format for 6-bit BCD code

Character	Standard BCD interchange code
0	00 0000
1	00 0001
2	00 0010
3	00 0011
4	00 0100
5	00 0101
6	00 0110
7	00 0111
A	11 0001
B	11 0010
C	11 0011
D	11 0100
E	11 0101

(b) The coding used to represent selected characters in a standard 6-bit code

Fig. 1-4 The 6-bit BCD code

Six-bit BCD Code.

Instead of using 4 bits with only 16 possible characters, computer designers commonly use 6, 7, 8 bits to represent characters in alphanumeric versions of BCD.

In the 6-bit code, the four BCD numeric place positions (1, 2, 4, and 8) are retained, but two additional zone positions are included (Fig. 1-4(a)).

With 6 bits, it's possible to represent 64 different characters (2^6).

This is a sufficient number to code the decimal digits (10), capital letters (26), and other special characters and punctuation marks (28).

Fig. 1-4(b) shows you how a few of the 64 possible characters are represented in a standard 6-bit BCD code.

Seven and Eight-Bit Codes.

Since 64 possible bit combinations isn't sufficient to provide decimal numbers (10), lower-case letters (26), capital letters (26), and a large number of other characters (28), designers have extended the 6-bit BCD code to 7 and 8 bits.

With 7 bits, it's possible to provide 128 different arrangements (2^7); with 8 bits, 256 variations are possible (2^8).

In addition to the four numeric place positions, there are three zone bit positions in a 7-bit code, and four zone bit positions in an 8-bit code.

The 7-bit American Standard Code for Information Interchange (ASCII) is widely used in data communications work and is by far the most popular code used to represent data internally in personal computers.

The ASCII format and the coding used to represent selected characters are shown in Fig. 1-5.

These are other two 8-bit codes in common use.

One is the Extended Binary Coded Decimal Interchange Code (EBCDIC).

This code is used in IBM mainframe models and in similar machines produced by other manufactures.

The other 8-bit code is ASCII-8, an 8-bit version of ASCII that is frequently used in the larger machines produced by some vendors.

Fig. 1-6 presents the 8-bit format and shows how selected characters are represented in these 8-bit codes.

Zone bits			Numeric bits			
Z	Z	Z	8	4	2	1

(a) Format for 7-bit ASCII code

Character	ASCII
0	01 10000
1	01 10001
2	01 10010
3	01 10011
4	01 10100
5	01 10101
6	01 10110
7	01 10111
A	10 00001
B	10 00010
C	10 00011
D	10 00100
E	10 00101

(b) The coding used to represent selected characters in the ASCII 7-bit code

Fig. 1-5 The 7-bit ASCII format and selected ASCII character codes

Zone bits				Numeric bits			
Z	Z	Z	Z	8	4	2	1

(a) Format for 8-bit EBCDIC and ASCII-8 codes

Character	Extended BCD Interchange code (EBCDIC)	ASCII-8
0	1111 0000	0101 0000
1	1111 0001	0101 0001
2	1111 0010	0101 0010
3	1111 0011	0101 0011
4	1111 0100	0101 0100
5	1111 0101	0101 0101
6	1111 0110	0101 0110
7	1111 0111	0101 0111
A	1100 0001	1010 0001
B	1100 0010	1010 0010
C	1100 0011	1010 0011
D	1100 0100	1010 0100
E	1100 0101	1010 0101

(b) The coding used to represent selected characters in the EBCDIC and ASCII 8-bit codes

Fig. 1-6 The format for EBCDIC and ASCII 8-bit codes and selected characters represented by these codes. Each 8-bit unit used to code data is called a byte

The main difference is in the selection of bit patterns to use in the zone positions.

Detecting Code Errors.

Computers are very reliable, but they're not infallible.

If just one bit in a string of 6, 7, or 8 bits is lost during data input, processing, or output operations, an incorrect character code will be created.

Such an error can be caused by dust particles on storage media, by improper humidity levels near the computer, or by many other factors.

Fortunately, however, computer designers have developed a method for detecting such errors by adding an extra check bit or parity bit to each 6, 7, or 8-bit character represented in storage.

Thus, as you can see in Fig. 1-7, a total of 7, 8, or 9 bits may actually be stored.

The designers of a particular computer model may then use the check bit to make sure that every valid character code will always have an even number of 1 bits.

The 6-bit coding used to represent selected characters was presented earlier in Fig. 1-4.

Several of these characters have been reproduced in Fig. 1-8.

Check Zone

bit	bits		Numeric bits			
C	B	A	8	4	2	1

(a)

Check

Bit	bits		Numeric bits				
C	Z	Z	Z	8	4	2	1

(b)

Check

bit	Zone bits				Numeric bits			
C	Z	Z	Z	Z	8	4	2	1

(c)

Fig. 1-7

(a) Format of 6-bit BCD code with check bit included.
(b) Format of 7-bit ASCII code with check bit included.
(c) Format of 8-bit codes with check included.

Check Zone

	bit	bits		Numeric bits			
1	1	0	0	0	0	0	1
2	1	0	0	0	0	1	0
3	0	0	0	0	0	1	1
A	1	1	1	0	0	0	1

Fig. 1-8 Using the check bit to make sure that every character has an even number of 1 bits. This is an example of the use of an even-parity format. Other computers are also designed to have the check bit to produce an odd-parity format

You'll notice that if the basic code for a character such as 1, 2, or A requires an odd number of 1 bits, an additional 1 bit is added in the check-bit location so that there will always be an even number of such bits.

This is an example of an even-parity format, but other computers use the check bit to produce an odd parity.

Since every valid character in a computer that uses even parity must always have an even number of 1 bits, circuits for parity checking are built into the computer to constantly monitor the data moving through the system.

The computer operator is notified if a bit is lost and a parity error is detected. Of course, parity checking will only detect coding errors.

It cannot signal the fact that incorrect data have been entered into the system if the data are properly coded.

NOTES

[1] 短语 by far 修饰比较级与最高级，强调数量、程度等，意为"……得多，最……"，这是一个用 and 连接的并列句。