注意：不只是标准 I/O 函数！！！！！！！！！！！！

函数名: scanf

功　能: 执行格式化输入

用　法: int scanf(char *format[,argument,...]);

程序例:

```
#include <stdio.h>
#include <conio.h>

int main(void)
{
   char label[20];
   char name[20];
   int entries = 0;
   int loop, age;
   double salary;

   struct Entry_struct
   {
     char   name[20];
     int    age;
     float salary;
   } entry[20];

/* Input a label as a string of characters restricting to 20 characters */
   printf("\n\nPlease enter a label for the chart: ");
   scanf("%20s", label);
   fflush(stdin);   /* flush the input stream in case of bad input */

/* Input number of entries as an integer */
   printf("How many entries will there be? (less than 20) ");
   scanf("%d", &entries);
   fflush(stdin);    /* flush the input stream in case of bad input */

/* input a name restricting input to only letters upper or lower case */
   for (loop=0;loop<entries;++loop)
   {
     printf("Entry %d\n", loop);
     printf("   Name    : ");
     scanf("%[A-Za-z]", entry[loop].name);
     fflush(stdin);   /* flush the input stream in case of bad input */

/* input an age as an integer */
     printf("   Age     : ");
     scanf("%d", &entry[loop].age);
     fflush(stdin);   /* flush the input stream in case of bad input */

/* input a salary as a float */
```

```
      printf("   Salary : ");
      scanf("%f", &entry[loop].salary);
      fflush(stdin); /* flush the input stream in case of bad input */
    }

/* Input a name, age and salary as a string, integer, and double */
    printf("\nPlease enter your name, age and salary\n");
    scanf("%20s %d %lf", name, &age, &salary);


/* Print out the data that was input */
    printf("\n\nTable %s\n",label);
    printf("Compiled by %s   age %d   $%15.2lf\n", name, age, salary);
    printf("-----------------------------------------------------\n");
    for (loop=0;loop<entries;++loop)
      printf("%4d | %-20s | %5d | %15.2lf\n",
        loop + 1,
   entry[loop].name,
   entry[loop].age,
        entry[loop].salary);
    printf("-----------------------------------------------------\n");
    return 0;
}
```

scanf()和 printf()类似，也使用格式字符串和参数列表。scanf()中的格式字符串表明字符输入流的目标数据类型。两个函数主要的区别在参数列表中。printf()函数使用变量、常量和表达式，而 scanf()函数使用指向变量的指针。

如果用 scanf()读取基本变量类型的值，在变量名前加上一个&，如果用 scanf()把字符串读入字符数组中，不要使用&。

scanf()函数使用空白（换行符、制表符和空格）把输入分成多个字段。在依次把转换说明和字段匹配时跳过空白。唯一例外的是%c 转换说明。根据%c，scanf()会读取每个字符，包括空白；如果使用%s 转换说明，scanf()会读取除空白以外的所有字符。

假设 scanf()根据一个%d 转换说明读取一个整数。scanf()函数每次读取一个字符，跳过所有的空白字符，直至遇到第 1 个非空白字符才开始读取。因为要读取整数，所以 scanf()希望发现一个数字字符或者一个符号（+或-）。如果找到一个数字或符号，它便保存该字符，并读取下一个字符。如果下一个字符是数字，它便保存该数字并读取下一个字符。scanf()不断地读取和保存字符，直至遇到非数字字符。如果遇到一个非数字字符，它便认为读到了整数的末尾。然后，scanf()把非数字字符放回输入。这意味着程序在下一次读取输入时，首先读到的是上一次读取丢弃的非数字字符。最后，scanf()计算已读取数字（可能还有符号）相应的数值，并将计算后的值放入指定的变量中。

函数名: printf

功　能: 产生格式化输出的函数

用　法: int printf(char *format...);

程序例:

```
#include <stdio.h>
#include <string.h>

#define I 555
#define R 5.5

int main(void)
{
   int i,j,k,l;
   char buf[7];
   char *prefix = buf;
   char tp[20];
   printf("prefix   6d      6o      8x        10.2e        "
          "10.2f\n");
   strcpy(prefix,"%");
   for (i = 0; i < 2; i++)
   {
     for (j = 0; j < 2; j++)
        for (k = 0; k < 2; k++)
    for (l = 0; l < 2; l++)
          {
            if (i==0)   strcat(prefix,"-");
            if (j==0)   strcat(prefix,"+");
            if (k==0)   strcat(prefix,"#");
            if (l==0)   strcat(prefix,"0");
            printf("%5s |",prefix);
            strcpy(tp,prefix);
            strcat(tp,"6d |");
            printf(tp,I);
            strcpy(tp,"");
            strcpy(tp,prefix);
            strcat(tp,"6o |");
            printf(tp,I);
            strcpy(tp,"");
            strcpy(tp,prefix);
            strcat(tp,"8x |");
            printf(tp,I);
            strcpy(tp,"");
            strcpy(tp,prefix);
       strcat(tp,"10.2e |");
       printf(tp,R);
       strcpy(tp,prefix);
       strcat(tp,"10.2f |");
       printf(tp,R);
       printf("   \n");
       strcpy(prefix,"%");
     }
```

```
    }
  return 0;
}
```

请求 printf()函数打印数据的指令要与待打印数据的类型相匹配。例如，打印整数时使用%d，打印字符时使用%c。这些符号被称为转换说明（conversion specification），它们指定了如何把数据转换成可显示的形式。

转换说明	输出
%a	浮点数、十六进制数和 p 记数法（C99/C11）
%A	浮点数、十六进制数和 p 记数法（C99/C11）
%c	单个字符
%d	有符号十进制整数
%e	浮点数，e 记数法
%E	浮点数，e 记数法
%f	浮点数，十进制记数法
%g	根据值的不同，自动选择%f 或%e。%e 格式用于指数小于-4 或者大于或等于精度时
%G	根据值的不同，自动选择%f 或%E。%E 格式用于指数小于-4 或者大于或等于精度时
%i	有符号十进制整数（与%d 相同）
%o	无符号八进制整数
%p	指针
%s	字符串
%u	无符号十进制整数
%x	无符号十六进制整数，使用十六进制数 0f
%X	无符号十六进制整数，使用十六进制数 0F
%%	打印一个百分号

同时与 puts()不同的是，printf()不会自动在每个字符串末尾加上一个换行符。因此，必须在参数中指明应该在哪里使用换行符。

getchar,getch

原型：extern int getchar(void);
用法：#include <ctype.h>
功能：读键

说明：从键盘上读取一个键，并返回该键的键值
getch 是到 getchar 的宏定义

举例：

```
// getchar.c
#include <stdio.h>
main()
{
  int c;
  clrscr();
  printf("Press key...");
  while((c=getchar())!='Q')
  {
    clrscr();
    printf("key: %c\nvalue: %x",c,c);
  }
}
```

putchar

原型：extern void putchar(char c);
用法：#include <stdio.h>
功能：在屏幕上显示字符 c
说明：字符输出在屏幕的当前位置。
　　可用 move 或 gotoxy 改变光标位置。
举例：

```
// putchar.c
#include <stdio.h>
#include <system.h>
#define CPR 14
main()
{
  int i,j,k;

  clrscr();

  textmode(0x00);
  for(i=1;i<6;i++)
  {
    k=i>3?(6-i):i;
    move(i,CPR/2-k);
    for(j=1;j<k*2;j++) putchar('*');
  }
  gotoxy(10,10);      // Hide Cursor

  getchar();
  return 0;
}
```

函数名: gets

功 能: 从流中取一字符串

用 法: char *gets(char *string);

程序例:

```
#include <string.h>
#include <stdio.h>

int main(void)
{
char string[80];

printf("Input a string:");
gets(string);
printf("The string input was: %s\n",
string);
return 0;
}
```

gets()函数简单易用，它读取整行输入，直至遇到换行符，然后丢弃换行符，储存其余字符，并在这些字符的末尾添加一个空字符使其成为一个 C 字符串。它经常和 puts()函数配对使用，该函数用于显示字符串，并在末尾添加换行符。

缺点是 gets()唯一的参数是 words，它无法检查数组是否装得下输入行。上一章介绍过，数组名会被转换成该数组首元素的地址，因此，gets()函数只知道数组的开始处，并不知道数组中有多少个元素。如果输入的字符串过长，会导致缓冲区溢出（buffer overflow），即多余的字符超出了指定的目标空间。如果这些多余的字符只是占用了尚未使用的内存，就不会立即出现问题；如果它们擦写掉程序中的其他数据，会导致程序异常中止；或者还有其他情况。

函数名: fgets

功 能: 从流中读取一字符串

用 法: char *fgets(char *string, int n, FILE *stream);

程序例:

```
#include <string.h>
#include <stdio.h>

int main(void)
{
FILE *stream;
char string[] = "This is a test";
char msg[20];

/* open a file for update */
stream = fopen("DUMMY.FIL", "w+");

/* write a string into the file */
fwrite(string, strlen(string), 1, stream);

/* seek to the start of the file */
fseek(stream, 0, SEEK_SET);

/* read a string from the file */
fgets(msg, strlen(string)+1, stream);

/* display the string */
printf("%s", msg);

fclose(stream);
return 0;
}
```

fgets()函数通过第 2 个参数限制读入的字符数来解决溢出的问题。该函数专门设计用于处理文件输入，fgets()函数的第 2 个参数指明了读入字符的最大数量。如果该参数的值是 n，那么 fgets()将读入 n-1 个字符，或者读到遇到的第一个换行符为止。如果 fgets()读到一个换行符，会把它储存在字符串中。这点与 gets()不同，gets()会丢弃换行符。fgets()函数的第 3 个参数指明要读入的文件。如果读入从键盘输入的数据，则以 stdin（标准输入）作为参数，该标识符定义在 stdio.h 中。

fputs()函数返回指向 char 的指针。如果一切进行顺利，该函数返回的地址与传入的第 1 个参数相同。但是，如果函数读到文件结尾，它将返回一个特殊的指针：空指针（null pointer）。在代码中，可以用数字 0 来代替，不过在 C 语言中用宏 NULL 来代替更常见（如果在读入数据时出现某些错误，该函数也返回 NULL）。

函数名: puts

功　能: 送一字符串到流中

用　法: int puts(char *string);

程序例:

```
#include <stdio.h>
int main(void)
{
   char string[] = "This is an example output string\n";

   puts(string);
   return 0;
}
```

puts()在遇到空字符时就停止输出，同时在显示字符串时会自动在其末尾添加一个换行符。

函数名: fputs

功 能: 送一个字符到一个流中

用 法: int fputs(char *string, FILE *stream);

程序例:

```
#include <stdio.h>

int main(void)
{
/* write a string to standard output */
fputs("Hello world\n", stdout);

return 0;
}
```

fputs()函数的第 2 个参数指明要写入数据的文件。如果要打印在显示器上，可以用定义在 stdio.h 中的 stdout（标准输出）作为该参数。与 puts()不同，fputs()不会在输出的末尾添加换行符。

函数名: sprintf

功　能: 送格式化输出到字符串中

用　法: int sprintf(char *string, char *farmat [,argument,...]);

程序例:

```
#include <stdio.h>
#include <math.h>

int main(void)
{
   char buffer[80];

   sprintf(buffer, "An approximation of Pi is %f\n", M_PI);
   puts(buffer);
   return 0;
}
```

sprintf()的第 1 个参数是目标字符串的地址。其余参数和 printf()相同，即格式字符串和待写入项的列表。 sprintf()函数获取输入，并将其格式化为标准形式，然后把格式化后的字符串储存在 formal 中。