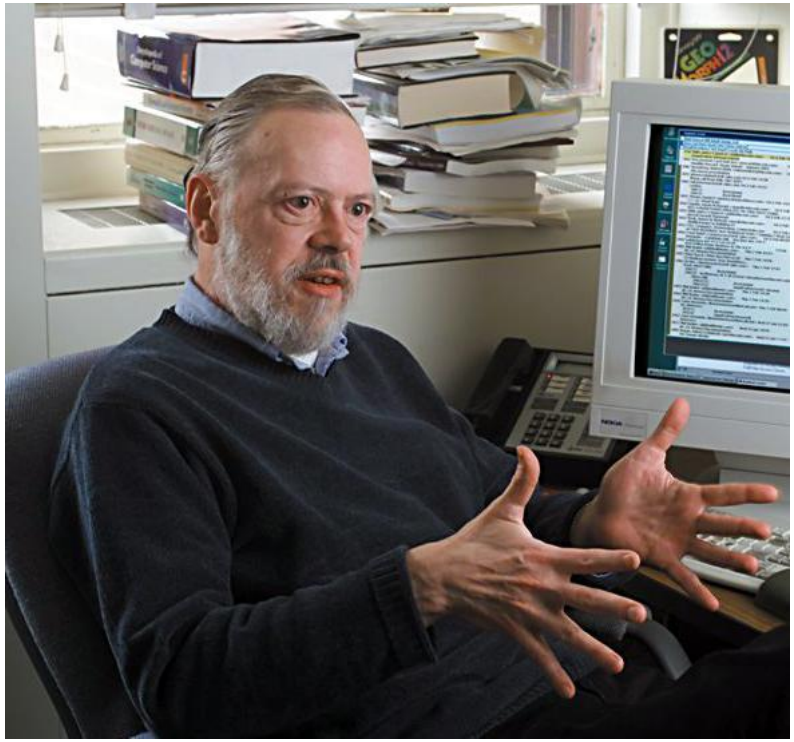
Dennis Ritchie（丹尼斯·里奇）

- 主要贡献
  - Unix操作系统
  - C语言
    - C++, C#, Objective-C, Java
- 获奖：图灵奖（1983）、IEEE汉明奖（1990）、美国国家技术与创新奖章（1999）、日本国际奖（2011）
- 有评论认为：他的贡献构成了四十年来人们直接或间接使用的计算机产品的DNA
- 很多介绍文献中说他1968年获得哈佛大学博士学位，事实似乎是：他提交并通过论文，但没有正式获得哈佛的博士学位
- 他终生未娶，大部分时间和父母、兄弟姐妹一起生活
- 2011年10月12日孤独去世（乔布斯去世后一周）



Geoffrey Hinton（杰弗里·辛顿）

- 学术界第二个获得图灵奖（2018年）和诺贝尔奖（2024年物理学奖）的科学家
- 深度学习“教父”

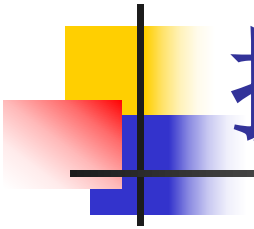
- 1947年12月6日，出生于英国温布尔登
  - 出身学术世家，George Boole是其高祖，祖父是数学家，父亲是昆虫学家
- 1970年获得剑桥大学实验心理学学士学位
  - 毕业后，做过一段时间木工
- 1978年获得爱丁堡大学人工智能学博士学位

- 先后在加州大学圣地亚哥分校、剑桥大学、卡耐基梅隆大学、多伦多大学和Google工作

第5讲
结构化程序设计（Part II）

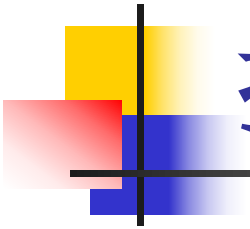
周水庚

2024年10月10日



提要

- C程序简介
- 函数基础知识
- 文件的简单用法
- 简单程序设计实例

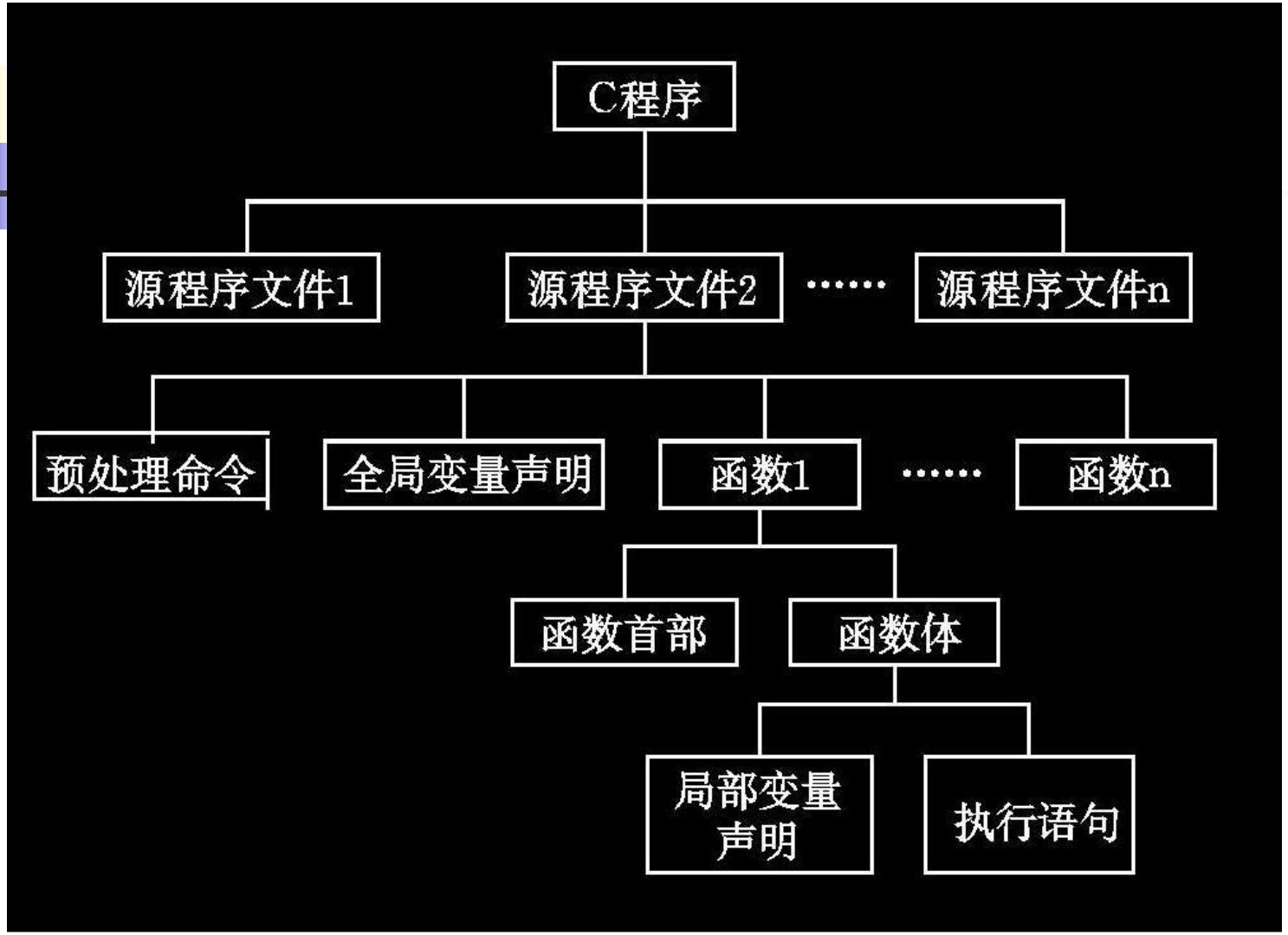


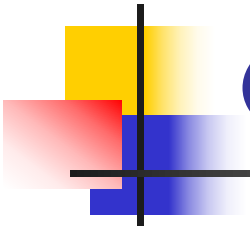
提要

- C程序简介
- 函数基础知识
- 文件的简单用法
- 简单程序设计实例

C程序简介

- 一个C程序可由若干个**源程序文件**组成
- 一个源文件可以由若干个**函数**和**预处理命令**以及**全局变量声明**部分组成
- 一个函数由**变量定义部分**和**执行语句**组成
- 每个源程序文件都是可独立编译的，所以C程序可以按源程序文件分别编写、分别编译


C程序
源程序文件1
源程序文件2
……
源程序文件n
预处理命令
全局变量声明
函数1
……
函数n
函数首部
函数体
局部变量
声明
执行语句



C语句

- C语句是C程序的基本组成单元
- C语句可以分为以下5类
  - 空语句
  - 表达式语句
  - 复合语句
  - 控制语句
  - 函数调用语句

C语句（续）

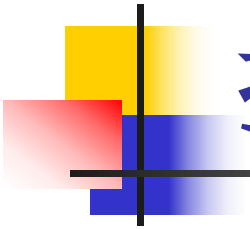
- 控制语句有9种
  - if() … else …　　(条件语句)
  - switch　　(多分支选择语句)
  - for()　　(循环语句)
  - while()　　(循环语句)
  - do … while()　　(循环语句)
  - continue　　(结束本次循环语句)
  - break　　(中止执行switch或循环语句)
  - goto　　(转向语句)
  - return　　(从函数返回语句)

C语句（续）

- 函数调用语句
  - 由一次函数调用加一个分号构成
- 表达式语句
  - 由一个表达式加一个分号构成，如赋值语句
- 空语句
  - 只有一个分号的语句，它什么也不做
- 复合语句
  - 用 {} 把一些语句括起来构成，又称分程序

C程序的三种基本结构

- 顺序结构
- 条件控制结构
  - if … [else …]
  - switch…case
- 循环结构
  - while()
  - do … while()
  - for()



提要

- C程序简介
- **函数基础知识**
- 正文文件的简单用法
- 简单程序设计实例

函数基础知识

- 结构化程序设计中，将复杂的功能分解成若干简单的子功能，用函数实现子功能，通过调用函数实施子功能要求
- 函数是一个实现指定功能、逻辑上独立的代码段
- 对函数使用者来说，把它看作“黑盒”，只需知道要传送给函数的数据（输入），和函数执行后能得到什么结果（输出）
- 函数可以定义局部对象，使函数在逻辑上作为程序的一个相对独立单位，不受主函数或其它函数的程序对象命名的影响

函数基础知识（续）

- 函数可带形参，函数执行时，操作对象、求值方式等可随不同调用的需要而改变
- 函数为程序的层次构造和开发提供支持，使设计新程序能在已有函数基础上构造功能更强的函数和程序
- 一个C程序以 main()函数作为程序的主函数。程序运行时，从它开始执行
- 在C语言中，函数不能嵌套定义，一个函数并不从属于另一个函数
- 除不能调用 main() 函数外，其它函数可以相互调用

函数库

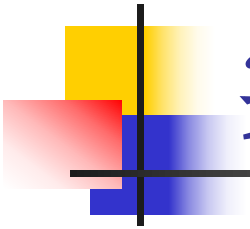
- 把一些公用的、基本的计算功能所对应的函数集中起来，构成一个库，我们称之为**函数库**，相应的函数成为**库函数**
- 函数库中的函数具有预先定义的、标准的**输入、输入接口**
- C语言中定义了一些基本的标准函数
- 编程环境工具厂商（Microsoft、Borland等）往往提供更多的函数供编程者使用

函数库（续）

- C语言使用**头文件**（header file，即***.h文件**）对函数库中的函数进行定义和说明
- 函数库中的函数经编译后，绑定在一起件，构成一个**库文件**（library file，即***.lib文件**）
- C程序调用C语言或者编程环境提供的函数时，要在程序中包含（include）相应的**头文件**；在产生执行文件时，需要与**库文件**中相应的目标函数代码连接
- 为了使用方便，C语言按功能分类，提供了大量函数库，每个函数库都有自己的头文件

库函数的使用

- 使用相应库函数的程序都要在使用之前写上包含相应头文件的预处理命令
- 常用的头文件
  - **stdio.h　(输入输出库函数)**
  - **math.h、stdlib.h、float.h　(数学库函数)**
  - **time.h　(时间库函数)**
  - **ctype.h　(字符分类和转换库函数)**
  - **string.h　(内存缓冲区和字符串处理库函数)**
  - **graphics.h　(图形处理库函数)**
  - **malloc.h、stdlib.h　(内存动态分配库函数)**
  - **signal.h、process.h　(进程控制库函数)**



实例

- 实例1：时间函数使用
- 实例2：随机函数使用（1）
- 实例3：随机函数使用（2）

```c
#include  <stdio.h>
#include  <time.h>
int main()
{ struct tm *tmNow;     /* 定义一个时间结构指针变量 */
  long secsNow;         /* 定义以秒为单位的记录时间的变量 */
  char *strNow;         /* 时间字符串表示的字符指针 */
  time(&secsNow);       /* 调用函数time(),得到当前时间 */
  strNow = ctime(&secsNow);   /* 获得时间的字符串表示 */
  printf("自1970年1月1日至现在的时间(以秒为单位):%ld\n", secsNow);
  printf("当前时间的另一种表示: %s\n", strNow);
  tmNow = localtime(&secsNow); /* 获得结构形式的时间 */
  printf("存于tmNow所指结构中的时间有年中的日、年、月、日、时、分、秒:\n");
  printf("%d  %d-%02d-%02d  %02d:%02d:%02d\n",
         tmNow->tm_yday,tmNow->tm_year+1900,
         tmNow->tm_mon,tmNow->tm_mday,
         tmNow->tm_hour,tmNow->tm_min,tmNow->tm_sec);
  strNow = asctime(tmNow); /* 得到字符串表示的时间 */
  printf("当前时间的另一种表示: %s\n", strNow);
}
```

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
void main()
{  int k; long now;
   srand(time(&now) % 60); /* 用时间初始化随机
                数发生函数的初态，使初态总不相同 */
   for(k = 0; k < 10; k++)  /* 产生10个随机数输出 */
      printf("%d\n", rand()); /* 调用随机函数 */
}
```

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
void main()
{  int i;
   srand(time(NULL)); /*使用随机数生成函数rand()前，需要先用srand()函数设置 产生随机数种子 */
                      /*  void srand(unsigned int seed)   */
   for(i=1; i<=10000; i++) {
     printf("%10d", 1+rand()%6);
     if(i%5==0)
      printf("\n");
   }
}
```

函数定义

- 函数定义的一般形式为

```
类型标识符　函数名(形式参数说明表)
{
   说明和定义部分
   语句序列
}
```

函数定义（续）

- 类型标识符用于标识函数执行结果返回值的类型
  - 当函数执行不返回值时，习惯用void来标记
  - 当函数返回int型值时，类型标识符int可以省略
- 函数名是一个标识符，一个C程序有且只有一个main()函数，其它的函数名可以随意命名

函数定义（续）

- 函数名之后括号内的形式参数说明表是按需要而定
  - 没有形参的函数，也就没有形参说明表，常用void代之，但函数名之后的一对圆括号不可省略
  - 当函数有多个形参时，形参说明之间用逗号分隔，每个形参说明指定形参的**类型**和**形参名**

函数定义（续）

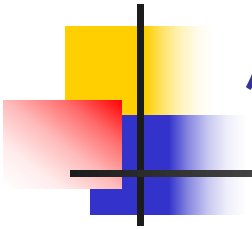
- 最外层花括号“{”和“}”括住的部分是函数体
  - 在函数体的前面部分可有函数需要的程序对象的说明和定义
  - 函数体内定义的变量是局部变量，只能在函数体内引用它们
  - 说明和定义之后是由语句序列组成的执行代码

例子1

- 求两个数中最小值的函数min()

```
double min(double x, double y) /* 返回 double 型值，有两个形参 x, y，都为 double 型的 */
{ /* 函数返回x和y中的小者的值 */
      return x < y ? x : y;
}
```

- return语句的执行将结束函数的执行
- C语言的return语句有两种形式：
  - return;用于不返回值的函数体中
  - return 表达式；用于有返回值的函数体中

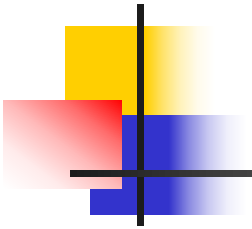


例子2

- 求两个正整数最大公因子的函数gcd()
- 两个正整数a和b的最大公因子有性质：
  - gcd(a, b) = gcd(a-b, b)，如a > b；
  - gcd(a, b) = gcd(a, b-a)，如a < b；
  - gcd(a, b) = a，　　如a = b。

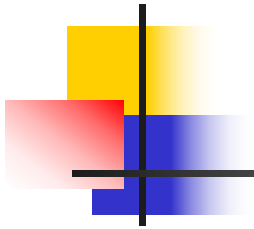
第1解法

```
int gcd(int a, int b)
 {  while( a != b)
       if(a > b) a -= b;
       else b -= a;
    return a;
 }
```



第2解法

- 步骤:
  - [求余数] 求a除b的余数r
  - [判结束] 如r等于0，b为最大公约数
  - [替换] 用b置a，r置b，并回到步骤[求余数]



```
int gcd(int a, int b)
 {  int r;
    while(1) {
      if((r = a % b) == 0) break;
      a = b;  b = r;
     }
    return b;
 }
```

```
int gcd(int a, int b)
 {  int r = a;
    do {
        a = b;
        b = r;
        r = a % b;
    } while (r);
    return b;
 }
```

函数定义（续）

- C语言也允许在函数名后的圆括号内只给出各形参的名，随后才指定各形参的类型，但这种写法在C++中已不允许
  - double min(x, y)
  - double x, y;
  - {
  - return x < y ? x : y;
  - }
- C语言还允许函数体为空的函数
  - dummy() /* 或 dummy(void) */
  - {
  - }

函数调用

- 函数被定义以后，凡要实现函数功能的地方，就可简单地通过函数调用来完成
- 函数调用的一般形式为
  - 函数名（实在参数表）
- 实在参数，简称实参。函数调用时，实参按它们出现的顺序与函数定义中的形参一一对应，并要求实参类型与其对应的形参类型相一致

函数调用（续）

- 函数调用有两种方式
  - 传值调用（call by value）
    - 把实参的值传给被调用函数的参数（形参）。这时，被调用函数对参数的改变，不影响调用函数实参的原始值
  - 传引用调用（call by reference）
    - 把实参的地址传给被调用函数的参数（形参）地址。这时，被调用函数对参数的改变，将影响到调用函数实参的原始值

函数调用（续）

- 对double min(double x, double y)的函数调用 w = min(u, v);
  - 函数调用min(u, v)就是对函数min()的调用，它提供了两个实参u和v，分别对应形参x和y
- 如果调用无形参的函数，这时函数的调用形式变为
  - 函数名()
  - 其中函数名之后的一对圆括号是不能省略的

函数调用（续）

- 按函数调用在程序中的作用，有两种不同类型的应用
  - **函数调用只是利用函数所完成的功能**。此时，将函数调用作为一个独立的语句。这种应用不要求或无视函数的返回值
    - 如程序中经常使用的调用格式输入函数scanf()和格式输出函数printf()等。
  - **函数调用是利用函数的返回值**：或用这返回值继续进行表达式的计算，或输出函数返回值等

函数调用的执行过程

1. 为形参分配内存空间
2. 计算实参表达式的值，并将值赋给对应的形参
3. 为函数的局部变量分配内存空间
4. 执行函数体内的语句序列
5. 函数体执行完，或执行了**return**语句后，释放为这次函数调用分配的全部内存空间
6. 将函数值（如果有）返回到函数调用处继续执行

```
#include  <stdio.h>
double x, y, d, min(double, double);
int main()
{
  printf("Enter x,y.\n"); scanf("%lf%lf",&x,&y);
  d = min(x, y);
  printf("MIN(%.3f, %.3f) = %.3f\n", x, y, d);
  return 0;
}
double min(double a, double b)
{  double temp;
   temp = a > b ? b : a;
   return temp;
}
```

对函数调用的说明

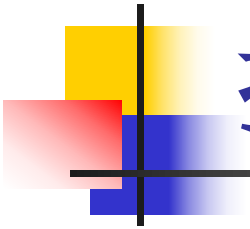
- 当函数执行return语句或执行完函数体的语句序列后，函数的这次调用就结束，随之将控制返回到函数调用处继续执行
- 函数的返回值是通过执行return 语句时，计算return之后的表达式值而获得的。如果函数不提供返回值，则return语句不包含表达式。
- 如果函数有返回值，则应有确定的类型，并在函数定义时指明。若函数定义时不指明返回值类型，且函数有返回值，C语言约定该函数的返回值类型为int型

对函数调用的说明（续）

- 为了明确指明函数不提供返回值，建议在函数定义时，在函数名之前写上**void**，并在这样的函数体内，所有的**return**语句都不带表达式
- 当函数执行不带表达式的**return**语句返回时，函数并不是一定不带回值，而是返回一个不确定的值。这时，不应该利用函数返回值进行再计算，否则会产生错误结果
- 函数定义中的**return**语句的表达式类型应与函数定义中指明的返回值类型相一致。如果**return**语句中的表达式类型与函数定义指明的返回值类型不一致时，对于基本类型情况，则以函数的返回值类型为准，系统会自动进行类型转换

实参向形参单向传递数据

- 在函数未被调用时，函数定义中的形参和函数体中定义的局部变量并不占用存储单元
- 在函数定义中，必须为函数的形参指定数据类型
- 函数体中所使用的形参的初值是由函数调用时对应的实参表达式给定的
- C语言规定，实参表达式对形参的数据传递是“值传递”的，即单向传递
- 对于有多个实参的函数调用情况，C语言不规定实参的求值次序



提要

- C程序简介
- 函数基础知识
- **文件的简单用法**
- 简单程序设计实例

文件的简单用法

- 介绍这部分内容的目的
  - 学习编写从文件输入数据和把结果输出到文件的程序
  - 了解使用文件程序的结构
  - 掌握文件的一般使用方法
  - 熟悉一些和文件操作有关的库函数的用法

涉及一些较深的概念先暂且接受!

定义文件变量

- 在程序的开始处定义文件指针变量，和存储文件名的字符数组

```
#include  <stdio.h>
FILE *fp; /* 定义文件指针变量fp */
char fname[40]; /* 存储文件目录路径和文件名的字符数组 */
```

输入文件名

```
printf("输入文件名(包括目录路径、扩展名)\n");

scanf("%s%*c",fname);/* 输入文件名及回车符 */
```

打开文件

- 程序从正文文件输入数据

读打开时，要求被打开文件已存在

```
if ((fp = fopen(fname, "r")) == NULL){
    printf("%s文件不能打开\n", fname);
    return;
}
```

- 程序向正文文件输出结果

```
fp=fopen(fname,"w");   /* 为写打开文件*/
```

  - 若被打开文件不存在，则建立一个新文件；若被打开文件已存在，则该文件中的数据被删除

关闭文件

- 文件使用结束后，要及时关闭

```
fclose(fp);  /* 以后fp又可用于打开文件 */
```

文件输入输出

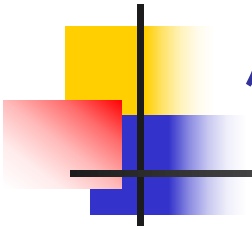
- 调用函数fgetc（）从文件输出下一个字符
  - ch=fgetc(fp); /* 将输入字符存于变量ch */
- 调用函数fscanf()从文件按指定格式输出数据
  - fscanf(fp,"%d%d",&k,&j); /* 从文件输出整数 */
- 调用函数fputc（）向文件输入一个字符
  - fputc(ch, fp); /* 将变量ch中的字符输出到文件*/
- 调用函数fprintf()向文件按指定格式输入数据
  - fprintf(fp,"%d %d\n", k, j); /* 向文件输入整数 */

从文件逐一输出字符

```
int c;  /* 不能为char类型 */
    FILE *fp;
     …  /* 说明有关变量和设置初值等 */
    if ((fp = fopen(文件名, "r")) == NULL) {
       printf("不能打开文件 %s。\n", 文件名);
       return;
    }
    while((c = fgetc(fp)) != EOF)  {
      …  /* 对刚读入的字符信息 c 作某种处理  */
    }
    fclose(fp);
    … /* 输出处理结果 */
```

字符逐一输入形成新文件

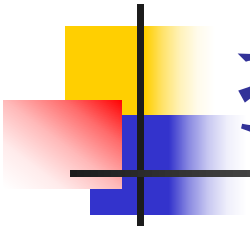
```
int c;  /* 也可以是char类型 */
    FILE *fp;
     …  /* 说明有关变量和设置初值等 */
    fp = fopen(文件名,"w");
    while(还有字符) {
       …  /* 生成字符(或字节)存于变量c */
      fputc(c, fp); /* 将生成的字符输出 */
    }
    fclose(fp);
      … /* 输出程序结束报告 */
```



例子

- 将键盘输入的字符流复制到指定的文件
  - 逐行复制从键盘输入字符到指定文件，直至输入空行结束

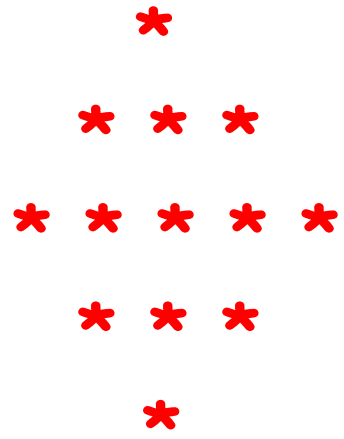
```
#include  <stdio.h>
FILE *fp;
int main()
{ int ch; char fname[40];
  printf("输入文件名!\n"); scanf("%s%*c", fname);
  fp=fopen(fname,"w"); /* 以写方式打开正文文件 */
  while((ch=getchar())!='\n'){ /* 逐行处理，至空行结束 */
    do fputc(ch, fp);    /* 行内字符逐一复制 */
    while ((ch = getchar()) != '\n');   /* 处理当前行 */
    fputc(ch, fp);  /* 输出换行符 */
  }
  fclose(fp);
  printf("程序复制键盘输入字符结束。\n");
  return 0;
}
```



提要

- C程序简介
- 函数基础知识
- 正文文件的简单用法
- 简单程序设计实例

【例1】输入整数n，输出由2*n+1行2*n+1列，以下形式(n = 2)的图案。



- 图案分成两部分，上面由n+1行，下面有n行
- 图案中，同一行上的两个星号字符之间有一个空格符
- 对于上半部分，设第一行的星号字符位于屏幕的中间，则后行图案的起始位置比前行起始位置提前两个位置
- 对于下半部，第一行的起始位置比上半部最后一行起始位置前进两个字符位置，以后各行也相继进两个位置

```
#include <stdio.h>
int main()
{  int n, j, k;
   printf("Enter n!\n"); scanf("%d", &n);
   for(j = 0;j <= n; j++) {
     printf("%*c", 40 - 2*j, ' ');
     for(k = 1; k <= 2*j+1; k++) printf(" *");
     printf("\n");
   }
   for(j = n-1; j >= 0; j--) {
     printf("%*c", 40 - 2*j, ' ');
     for(k = 1; k <= 2*j+1; k++) printf(" *");
     printf("\n");
   }
}
```

"%*c"中的*表示输入宽度由后面的参量值确定

【例2】试编制一个程序输出以下形式的乘法表。

	1	2	3	4	5	6	7	8	9
1	1								
2	2	4							
3	3	6	9						
4	4	8	12	16					
5	5	10	15	20	25				
6	6	12	18	24	30	36			
7	7	14	21	28	35	42	49		
8	8	16	24	32	40	48	56	64	
9	9	18	27	36	45	54	63	72	81

```
#include <stdio.h>
int main()
{ int i, j;
  printf("\n\t   1   2   3   4   5   6   7   8   9\n");
  for(i=1; i <= 9; i++) {
     printf("\t%d", i);
     for(j = 1; j<=i; j++) printf("%4d", i*j);
     printf("\n");
  }
  printf("\n\n\n");
}
```

【例3】编制一个程序，实现输入n(>2) 个整数，输出其中的次最大数。

为求次最大，程序在循环过程中需保留两个数，当前暂时最大数max1和当前暂时次最大数max2。

程序将输入的第一个数暂时保留，待输入第二个数后，确定max1和max2。

从第三个输入数x开始，根据x调整max1和max2。调整过程需考虑以下几种可能：

x>max1，则以max1作为新的max2，x作为新的max1；

max1>x>max2，则以x作为新的max2；

x<max2，则不调整。

```
#include <stdio.h>
int main()
{  int n, i, max1, max2, x, temp;
   printf("输入 n(>=2)!\n"); scanf("%d", &n);
   if (n < 2) return;
   printf("输入第 %d 个整数.  ", 1);
   scanf("%d", &temp);
   printf("输入第 %d 个整数.  ", 2);
   scanf("%d", &x);
   if (temp < x) { max1 = x; max2 = temp;  }
```

```
 else { max1 = temp; max2 = x; }
 for(i = 3; i <= n; i++) {
      printf("输入第 %d 个整数.  ", i);
      scanf("%d", &x);
      if ( x > max1) {
          max2 = max1; max1 = x;
      }
      else if (x > max2) max2 = x;
 }
 printf("次最大是 %d\n\n", max2);
}
```

【例4】编制对给定的整数，判该整数是否是质数的函数，若是质数，函数返回1，否则函数返回0。

判一个整数n是否是一个质数有许多方法：

如n=2，则n是质数；

若n是其它偶数，则n不是质数；

让整数变量k自3开始，每次增2，直至k 的平方超过n为止，若其中某个k能整除n，则n不是质数。若所有这样的k都不能整除n，则n是质数。

```
int isPrime(long n)
{   long k;
    if (n == 2L) return 1;
    if (n % 2 == 0) return 0;
    for(k = 3L; k*k <= n; k += 2L)
        if (n % k == 0) break;
    if (k*k > n) return 1;
    return 0;
}
```

【例5】编制输入整数，输出小于等于该整数的全部质数的程序。程序首先输出质数2，之后对指定范围内的奇数采用例4的方法判其是否是质数。

```
#include  <stdio.h>
int main()
{  long m, n; int j; /* j控制每行输出10个质数 */
   printf("输入整数\n"); scanf("%ld", &m);
   printf("%6d", 2);  j = 1;
   for(n = 3L; n <= m; n += 2)
     if (isPrime(n)) {
        if (j++ % 10 == 0) printf("\n");
        printf("%6ld", n);
     }
     printf("\n");
 }
```

【例6】输入x，求级数s(x)的近似值。约定求和的精度为0.000001。

$$s(x) = x + \frac{x^3}{3*1!} + \frac{x^5}{5*2!} + \frac{x^7}{7*3!} + \ldots$$

一般地，设级数为

$$s(x) = t_0 + t_1 + t_2 + \text{---} + t_k$$

求级数部分和的算法可描述如下：

```
{ s = 0;  /* 级数的部分和变量s,置初值0 */
  t = 首项值; /* 置通项变量t为级数的首项值 */
  k = 0; /* 置项序号变量k为0 */
  while (fabs(t) >= Epsilon) {
    s += t;  /* 累计当前项tk到部分和 */
    t = f(t,k);/* 由当前项t和k计算下一个当前项的值 */
    k++;     /* 项序号增1 */
  }
}
```

对于本题，首项值为x，级数第k(>=0)项t_k的算式为

$$(-1)^k * x^{(2*k+1)}/((2*k+1)*k!)$$

k+1项t_{k+1}与k项t_k有关系

$$t_{k+1} = -t_k*x*x*(2*k+1)/((2*k+3)*(k+1))$$

t_k是通项t的当前项值，t_{k+1}是通项t的下一个当前项值。由当前项t和k计算t的下一个当前项值t'，可用以下表达式实现：

$$t' = -t*x*x*(2.0*k+1.0)/((2.0*k+3)*(k+1))$$

把以上式子代入上述算法，并令x的值由输入给定，写出程序如下：

```
#include  <stdio.h>
#include  <math.h>
#define  Epsilon  0.000001
int main()
{  int k; double s, x, t;
   printf("Enter x.\n");  scanf("%lf", &x);
    s = 0.0;  /* 级数的部分和变量s置初值0 */
    t = x; /* 置通项变量t为级数的首项值 */
    k = 0; /* 置项序号变量k为0 */
    while (fabs(t) >= Epsilon) {
      s += t; t = -t*x*x*(2.0*k+1)/((2.0*k+3)*(k+1));
      k++;      /* 项序号增1 */
    }
    printf("s(%f) = %f\n", x, s);
}
```

【例7】编制从键盘输入整数序列，并按输入顺序输出到指定的文件中的程序。

程序循环地从键盘输入整数，将整数输出到指定的文件中。当程序发现不能从键盘输入整数时结束循环。

```
#include <stdio.h>
FILE *fp;
int main()
{ int x, k;  char fname[40];
  printf("输入文件名!\n");scanf("%s%*c", fname);
  if ((fp=fopen(fname, "w")) == NULL) {
    printf("不能打开文件 %s。\n", fname);
    return 0; }
  k = 1;
  while(scanf("%d", &x) == 1) {  /* scanf返回值表示输入参数的个数  */
    fprintf(fp, "%d\t", x);
    if (k++ % 5 == 0) fprintf(fp, "\n");
}
  fclose(fp);
  printf("\n输出了 %d个整数.\n", k-1);
  return 1;
}
```

【例8】编制从指定的文件中输出整数，并按输出顺序输出到显示屏上。

程序从文件循环地输入整数，将整数输出到显示屏。当程序发现不能从文件输入整数时结束循环。

```
#include  <stdio.h>
FILE *fp;
int main()
{ int x, k;    char fname[40];
  printf("输入文件名!\n");
  scanf("%s%*c", fname);
  if ((fp=fopen(fname, "r")) == NULL)  {
     printf("不能打开文件 %s。\n", fname);
     return 0;}
  k = 1;
  while(fscanf(fp, "%d", &x) == 1) {
      printf("%d\t", x);
      if (k++ % 5 == 0) printf("\n"); }
  fclose(fp);
  printf("\n从文件 %s输入了 %d个整数.\n", fname, k-1);
  return 1;
}
```

【例9】求Sn=a+aa+aaa+…+aa…a(n个a之值)，其中a是一个十进制数字。

```
#include<stdio.h>
int main()
{ long sn; int i, a, t, n;
   printf("Input n and a:");  scanf("%d %d", &n, &a);
   t=a;sn=0;i=1;
   do{
     sn+=t; i++;t=t*10+a;
   }while(i<n);

   printf("Sn=%l\n", sn);
}
```

【例10】有一分数序列：2/1,3/2,5/3,8/5,13/8,21/13,…

求出这个数列的前n项之和

- 分析：
  - 假设这个序列的第k项为$t_k=a_k/b_k$，有$t_{k+1}=(a_k+b_k)/a_k=1+1/t_k$
  - $a_1=2$; $b_1=1$

```
#include<stdio.h>
int main()
{ int i=1,n, float t, s
   printf("Input n:"); scanf("%d", &n);
   s=0.0;t=2.0
   while(i<=n){
     s+=t;
     t=1.0+1.0/t;i++
   }

   printf("The sum is: %f", s);
}
```

【例11】用迭代法求 $x=\sqrt{a}$，求平方根的迭代公式为

$$x_{n+1}=\frac{1}{2}(x_n+\frac{a}{x_n})$$

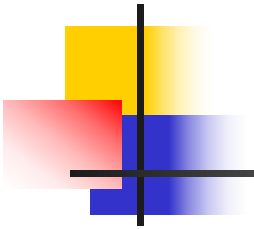
要求前后两次求出的x的差的绝对值小于10-5。

```
#include<stdio.h>
int main()
{float a, x1,x2;
  printf("Input a's value:"); scanf("%f", &a);
  x2=a;
 do{
   x1=x2;    x2=0.5*(x1+a/x1);
  } while(fabs(x2-x1)>=1e-5);

  printf( x=√a =%f", x2) ;
}
```

【例12】打印出所有的“水仙花数”，所谓“水仙花数”是指一个3位数，其各位数字立方和等于该数本身。例如，153是一水仙花数，因为153=$1^3+5^3+3^3$

```
#include<stdio.h>
int main()
{
int i, first, second, third;
for(i=100; i<=999; i++){
    first=i/100; second=(i/10)%10; third=(i%100)%10;
 if (i==first*first*first+second*second*second+third*third*third)
   printf("%d\n", i);}
}
```



【例13】一个数如果恰好等于它的因子之和，这个数就称为“完数”。例如，6的因子为1、2、3，而6=1+2+3，因此6是“完数”。编程序找出1000之内的所有完数，并按下面格式输出其因子：

6's factors are 1,2,3

```
#include<stdio.h>
int main()
{int i, j, sum;
 for(i=1;i<=1000;i++){
    sum=0;
    for(j=1;j<i;j++)
      if(i%j==0)sum+=j;
    if(i==1)
      printf("1's factor is 1")
    else if(sum==i){
      printf("%d's factors are:");
      for(j=1;j<i;j++)
        if(i%j==0)printf("%d,", j);
      printf("\n");
    }
 }
}
```

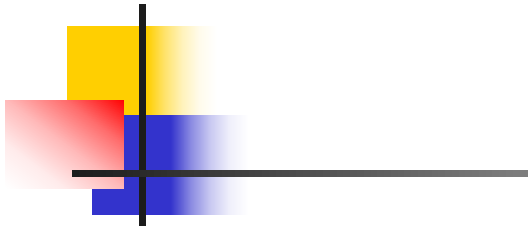
猴子吃桃问题

- 【例14】猴子第一天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了一个。第二天早上又将第一天剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第 10 天早上想再吃时，发现只剩下一个桃子了。编写程序求猴子第一天摘了多少个桃子。
- 关键是：搞清楚第一天桃数和第二天桃子数之间的关系，即第二天桃子数加 1 的 2 倍等于第一天的桃子数。

```
#include <stdio.h>
int main()
{
  Int day,x1,x2; /*定义 day、x1、x2 3 个变量为基本整型*/
  day=9;
  x2=1;
  while(day>0){
    x1=(x2+1)*2; /*第一天的桃子数是第二天桃子数加1后
                   的2倍*/
    x2=x1;
    day--; /*因为从后向前推所以天数递减*/
  }
  printf("the total is %d\n",x1); /* 输出桃子的总数*/
  return 0;
}
```

渔夫打鱼晒网问题

- **【例15】**如果一个渔夫从 2011 年 1 月 1 日开始三天打渔，两天晒网，编程实现当输入 2011年 1 月 1 日以后的任意一天，输出该渔夫是在打渔还是在晒网。



```
#include <stdio.h>
int leap(int a) /*自定义函数leap()用来指定输入的年份是否为闰年*/
{
   if (a%4==0&&a%100!=0||a%400==0) /* 闰年判定条件 */
     return 1; /*是闰年返回1*/
  else
     return 0; /*不是闰年返回O*/
}
```

```
int number(int year,int month,int day)
/*自定义函数 number() 计算输入日期距2011年1月1日共有多少天*/
{
int sum = 0, i, j, k;
int a[12]={31,28,31,30,31,30,31,31,30,31,30,31}; /*数组a存放平年每月的天数*/
int b[12]={31,29,31,30,31,30,31,31,30,31,30,31}; /*数组b存放闰年每月的天数*/
if(leap(year)) /*判断是否为闰年*/
  for(i=0;i<month-1;i++)
  sum+=b[i]; /*是闰年，累加数组b前m-1个月份的天数*/
else
  for(i=0;i<month-1;i++)
  sum+=a[i]; /*不是闰年，累加数组a前m-1个月份的天数*/
for(j=2011;j<year;j++)
  if (leap(j))
    sum+=366; /*2011年到输入的年份是闰年的加366*/
  else
    sum+=365; /*2011年到输入的年份不是闰年的加365*/
  sum+=day; /*将前面累加的结果加上日期，求出总天数*/
  return sum; /*返回计算的天数*/
}
```

```
int main()
{
 int year,month,day,n;
 printf("请输入年月日\n");
 scanf("%d%d%d",&year,&month,&day); /*输入年月日*/
 n=number(year,month,day); /*调用函数 number()*/
 if((n%5)<4&&(n%5)>0) /*余数是1或2或3时在打渔，否则在晒网*/
 printf("%d：%d：%d 打鱼\n",year,month,day);
 else
 printf("%d：%d：%d 晒网\n",year,month,day);
 return 0;
}
```

求亲密数

- 如果整数A的全部因子（包括1，不包括A本身）之和等于B；且整数B的全部因子（包括1，不包括B本身）之和等于A，则将整数A和B称为亲密数

- 求3000以内的全部亲密数

```
#include<stdio.h>
int main(){
  int a, i, b, n;
  printf("There are following friendly--numbers pair smaller than 3000:\n");
  for( a=1; a<3000; a++ ) /*穷举3000以内的全部整数*/
  {
    for( b=0, i=1; i<a; i++) /*计算数a的各因子，各因子之和存放于b*/
      if(!(a%i))  b+=i;
    for( n=0, i=1; i<b; i++ ) /*计算b的各因子，各因子之和存于n*/
      if(!(b%i))  n+=i;

    if(n==a && a<b) /*使每对亲密数只输出一次*/
    printf("%4d--%4d ", a, b); /*若n=a，则a和b是一对亲密数，输出*/
  }
  return 0;
}
```

第3章作业

- 习题：1，5，13，18，20，21，23，24