**《C 语言程序设计》**

**（第二版）**

**教案**

**北京出版社**

**课时分配表**

|  |  |  |  |
| --- | --- | --- | --- |
| **章序** | **课程内容** | **课时** | **备注** |
| **1** | **认识 C 程序** | **6** |  |
| **2** | **C 程序设计基础** | **7** |  |
| **3** | **顺序结构程序设计** | **5** |  |
| **4** | **选择结构程序设计** | **3** |  |
| **5** | **循环结构程序设计** | **6** |  |
| **6** | **数组** | **5** |  |
| **7** | **用户自定义函数** | **7** |  |
| **8** | **编译预处理** | **5** |  |
| **9** | **指针** | **9** |  |
| **10** | **构造数据类型** | **5** |  |
| **11** | **文件读写技术** | **6** |  |
| **12** | **C 语言综合实训** | **4** |  |
| **13** | **附录** | **4** |  |
| **总计** |  | **72** |  |

**第11课 文件读写技术**

|  |  |  |
| --- | --- | --- |
| **课 题** | 文件读写技术 | |
| **课 时** | 6课时（270 min）。 | |
| **教学目标** | **知识技能目标：**  1．了解文件的打开与关闭。  2．通过学习与练习掌握字符的读写。  **思政育人目标：**  让学生通过学习文件读写技术，能对信息进行有效管理，具备较高的存储意识。 | |
| **教学重难点** | **教学重点：**文件的打开与关闭  **教学难点：**字符的读写 | |
| **教学方法** | 讲授法、问答法、讨论法 | |
| **教学用具** | 电脑、投影仪、多媒体课件、教材 | |
| **教学设计** | 第1节课：考勤（2min）--知识讲解（40min）--作业布置（3min）  第2节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第3节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第4节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第5节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min）  第6节课：知识讲解（40min）--课堂小结（3min）--作业布置（2min） | |
| **教学过程** | **主 要 教 学 内 容 及 步 骤** | **设计意图** |
| **考勤**  **（2min）** | ■【教师】清点上课人数，记录好考勤  ■【学生】班干部报请假人员及原因 | 培养学生的组织纪律性,掌握学生的出勤情况 |
| **知识讲解**  （40min） | **【教师】**文件的打开与关闭  **任务描述**  打开当前目录中的 test.dat 文件，判断并输出文件打开状态信息，然后关闭文件。  **任务分析**  1. 定义一个文件指针 fp 备用；  2. 利用 fopen( ) 函数打开文件；  3. 判断文件是否打开成功，并输出文件状态信息；  4. 关闭文件。  **预备知识**  **一、文件型指针**  一个指针变量指向一个文件，这个指针称为文件指针。文件进行的各种操作就是通过文件指针完成的。C 语言是通过系统提供的结构类型 FILE 来定义文件指针变量的。结构类型 FILE 已在头文件“stdio.h”中定义了，所以在对文件进行操作时，直接使用 FILE 即可。其一般格式如下：  FILE \* 指针变量名 ;  例如，定义一个文件指针 f1 的形式如下：  FILE \*f1;  表示 f1 是指向 FILE 结构的指针变量，通过 f1 即可找到存放某个文件信息的结构变量，然后按结构变量提供的信息找到该文件，实施对文件的操作。  **二、文件的打开与关闭**  对磁盘文件的操作步骤是：“先打开，再读写，最后关闭。”  1. 文件的打开——fopen( )  所谓打开文件，实际上是建立文件的各种有关信息，并使文件指针指向该文件，以便进行其他操作。  打开文件的函数是 fopen( )，其一般形式为：  文件指针 =fopen( 文件名 , 文件使用方式 );  【功能】按照“文件使用方式”打开 < 文件名 > 指定的文件，并使“文件指针”指向该文件。  【注意】  （1）在打开一个文件时，如果出错，fopen 将返回一个空指针值 NULL。在程序中可以用这一信息来判别是否完成打开文件的工作，并作相应的处理。常用以下程序段打开文件：  if ((fp=fopen("text.txt","r"))==NULL)  {  printf("error on opening text.txt");  getchar();  exit(0);  }  这段程序的意义是：如果返回的指针为空，表示不能打开 text.txt 文件，则给出提示信息“error on opening text.txt”，下一行 getchar( ) 的功能是从键盘输入一个字符，但不在屏幕上显示。在这里，该语句的作用是等待，只有当用户从键盘按任意键时，程序才继续执行，因此用户可利用这个等待时间阅读出错提示。按键后执行 exit（0）退出程序。  exit（0）是终止程序运行的函数，其功能是强迫当前程序的继续运行，自动关闭已经打开的文件。  （2）用“r”打开一个文件时，该文件必须已经存在，且只能对该文件读。用“w”打开的文件只能向该文件写入。若打开的文件不存在，则以指定的文件名建立该文件，若打开的文件已存在，则将该文件删去，重建一个新文件。  （3）若要向一个已存在的文件追加新的信息，只能用“a”方式打开文件。但此时该文件必须是存在的，否则将会出错。  （4）把一个文本文件读入内存时，要将 ASCII 码转换成二进制码，而把文件以文本方式写入磁盘时，也要把 ASCII 码转换成二进制码，因此文本文件的读写要花费较多的转换时间。对二进制文件的读写不存在这种转换。  2. 文件的关闭——fclose( )  文件一旦使用完毕应及时关闭，以避免发生文件的数据丢失等错误。关闭文件的函数是 fclose，其调用的一般形式是：  fclose( 文件指针变量 );  【功能】关闭文件指针变量所指的文件，同时自动释放分配给文件的内存缓冲区。假如能正确关闭指定的文件，则返回 0 值，否则返回非 0 值。  例如：  fclose(fp);  正常完成关闭 fp 所指文件时，fclose 函数返回值为 0，若返回非 0 值，则表示有错误发生。可以通过返回值监控文件的打开与关闭是否正常进行。  **任务实施**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* 程序 : program11\_1.c \*/  /\* 功能 : 打开当前目录中的 test.dat 文件 , 判断输出打开状态信息后关闭它。 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #include <stdio.h>  void main()  { FILE \*fp; /\* 定义文件指针 fp \*/  fp=fopen("test.dat","r"); /\* 以只读方式打开 test.dat 文件 \*/  if(fp==NULL) /\* 判断文件是否打开成功 , 并输出状态信息 \*/  printf(" 文件打开失败 !");  else  printf(" 文件打开成功 !");  fclose(fp); /\* 关闭打开的文件 \*/  }  【注意】  1. 运行该程序时，当前目录中必须有 test.dat 文件，否则打开文件失败。  2. 打开文件 fopen( ) 和关闭文件 fclose( ) 都是通过函数来实现的，这两个系统函数定义在头文件 stdio.h 中，所以在程序开头要包含“#include <stdio.h>”。  **【学生】**思考、讨论。 | **展示文件的打开与关闭，让学生更加仔细的阅读，从而激发学生的学习欲望。** |
| **作业布置**（3min） | **【教师】**布置课后作业  简述文件的打开与关闭的意义。 | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**  （40min） | **【教师】**字符的读写  **任务描述**  任务 11\_2\_1　读取文件 f1.txt，在屏幕上输出；  任务 11\_2\_2　输入一行字符，将它存入文件 input，再读取文件内容并显示出来。  **预备知识**  字符的读写是靠字符读写函数实现的。字符读写函数是以字符为单位的读写函数，每次可从文件读出或向文件写入一个字符。  **一、字符读取函数 fgetc**  fgetc 函数调用的形式为：  fgetc( 文件指针 fp);  【功能】从文件指针 fp 所指的文件的当前位置读取单个字符。假如函数运行正确，则返回单个字符，否则返回 EOF（值为 -1）。  在 fgetc 函数调用中，读取的文件必须是以读或读写方式打开的，在文件内部有一个位置指针，用来指向文件的当前读写字节，在文件打开时，该指针总是指向文件的第一个字节。使用 fgetc 函数后，该位置指针将向后移动一个字节。因此可连续多次使用fgetc 函数，连续读取多个字符。  **二、字符写入函数 fputc**  fputc 函数调用的形式为：  fputc( 字符量 ch, 文件指针 fp);  其中的字符量 ch 可以是字符常量或字符变量。  【功能】将字符量写入 fp 所指向的文件的当前位置。假如函数运行正确，则返回刚写入文件的字符，否则返回 EOF。  被写入的文件可以用写（w）、读写（w+）、追加（a）方式打开。用写或读写方式打开一个已存在的文件时将清除原有的文件内容，如需保留原有文件内容，希望写入的字符存放在源文件末尾，必须以追加方式打开文件。写入字符从文件开头开始，每写入一个字符，文件内部的位置指针向后移动一个字节。  **任务实施**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* 程序 : program 11\_2\_1.c \*/  /\* 功能 : 读取文件 f1.txt, 在屏幕上输出。 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #include <stdio.h>  void main()  {  FILE \*fp; /\* 定义文件指针 fp \*/  char ch;  if((fp=fopen("f1.txt","r"))==NULL)  {  printf("contnot open fi le!");  exit(0);  }  ch=fgetc(fp);  while(ch!=EOF)  {  putchar(ch);  ch=fgetc(fp);  }  fclose(fp);  }  程序中首先定义了文件指针 fp，以读文本文件方式打开文件 f1.txt，并使 fp 指向该文件。若打开文件出错，则终止程序，否则，读出一个字符后进入循环，只要读出的字符不是文件结束标志（每个文件末有一个结束标志 EOF）就显示该字符，再读入下一个字符，继续显示。每读一次，文件内部的位置指针向后移动一个字符，文件结束时，该指针指向 EOF。  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* 程序 : program 11\_2\_2.c \*/  /\* 功能 : 输入一行字符 , 将它存入文件 input, 再读取文件内容并显示出来。 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #include <stdio.h>  void main()  { FILE \*fp;  char ch;  if((fp=fopen("input.txt","w+"))==NULL)  {  printf("cannot open fi le!");  exit(0);  }  printf(" 输入一行字符 :\n");  ch=getchar(); /\* 输入一行字符 , 将它存入文件 input \*/  while(ch!= '\n')  {  fputc(ch,fp);  ch=getchar();  }  fclose(fp);  if((fp=fopen("input.txt","r"))==NULL)  {  printf("cannot open fi le!");  exit(0);  }  ch=fgetc(fp); /\* 读取文件 input 的内容 , 并显示出来 \*/  while(ch!=EOF)  {  putchar(ch);  ch=fgetc(fp);  }  fclose(fp);  }  程序运行结果：  输入一行字符 :  Iliketoplayfootball  输出：  Iliketoplayfootball  程序中首先以读写文本文件方式打开文件 input.txt。在提示输入一串字符串后，从键盘读入一个字符串后进入循环，当读入字符不是“\n”（换行符）时，文件内的位置指针会向后移动一个字节，写入完毕，指针已指向文件末尾。若想把文件从头读出，需把指针移向文件头，可以先将文件关闭，再将文件打开，这样文件指针会回到文件头。将位置指针指向文件开头之后，就按照前面的知识将文件内的字符一个一个读出并显示即可。  **【学生】**思考、讨论。 | **通过教师讲解，了解字符的读写的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了字符的读写，让学生要善于保存资料、学会资源共享，具备良好的使用文件的习惯。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | **【教师】**布置课后作业  简述fgetc 函数调用的形式。 | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**字符串的读写  **任务描述**  任务 program 11\_3\_1　读取文件 input 中的一个包含 6 个字符的字符串；任务 program 11\_3\_2　在【program11\_2\_2.c】建立的文件 input.txt 中追加一个字符串。  **预备知识**  **一、字符串读取函数 fgets**  fgets 函数调用的形式为：  fgets( 字符数组名 st,n, 文件指针 fp);  【功能】从 fp 所指向的文件中读取 n-1 个字符，并在最后一个字符后加上串结束标志“\0”，组成字符串后存入字符数组 st（st 指定的内存区）中。假如函数运行正确，则返回数组 st 的首地址，否则返回 NULL。  【注意】在读完 n-1 个字符之前，假如遇到了换行符或 EOF，则只读到该位置便停止，在其后补充字符串结束标志“\0”，组成字符串，存入 st 指定的内存区中。  **二、字符串写入函数 fputs**  fputs 函数调用的形式为：  fputs(st, 文件指针 fp);  其中的 st 可以是字符串常量，也可以是字符数组名，或是指针变量。  【功能】向文件写入一个字符串。写入文件时，字符串结束标志并不写入文件。假如函数运行正确，则返回写入文件的实际字符个数，否则返回 EOF。  例如：  fputs("china",fp);  其意义是把字符串“china”写入 fp 所指的文件中。  **任务实施**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* 程序 : program11\_3\_1.c \*/  /\* 功能 : 读取文件 input 中的一个包含 6 个字符的字符串。 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #include <stdio.h>  void main()  {  FILE \*fp;  char st[6];  if((fp=fopen("input.txt","r"))==NULL) /\* 以读文本文件方式打  开 input.txt \*/  {  printf("cannot open fi le!");  exit(0);  }  fgets(st,6,fp); /\* 读取 input.txt 中的字符串 \*/  printf("%s",st); /\* 输出字符串 \*/  fclose(fp);  }  上述程序中首先定义一个字符数组 st，共 6 个字节，以读文本文件方式打开文件input.txt，从中读出 5 个字符送入 str 数组，在数组最后一个单元内加上“\0”，然后在屏幕上输出 st 数组。  程序运行结果：  Ilike  可见，输出的 5 个字符正是【program11\_2\_2.c】建立的 input.txt 文件的前 5 个字符。  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* 程序 : program 11\_3\_2.c \*/  /\* 功能 : 在【program11\_2\_2.c】建立的文件 input.txt 末尾追加一个字符串。 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #include <stdio.h>  void main()  {  FILE \*fp;  char ch,st[20];  if((fp=fopen("input.txt","a"))==NULL) /\* 用追加读写方式打开文件  input.txt\*/  {  printf("cannot open fi le!");  exit(0);  }  printf("input a string:\n");  scanf("%s",st);  fputs(st,fp);  fclose(fp);  if((fp=fopen("input.txt","r"))==NULL)  {  printf("cannot open fi le!");  exit(0);  }  ch=fgetc(fp);  while(ch!=EOF)  {  putchar(ch);  ch=fgetc(fp);  }  fclose(fp);  }  本例用追加读写文本文件的方式打开文件 input.txt。然后输入字符串，并用 fputs 函数把该字符串写入文件 st。注意，程序通过关闭文件后再打开的方法把文件内部位置指针移到文件开头，再进入循环逐个显示当前文件中的全部内容，即可实现所要求的功能。  程序运行结果：  输入：  china ↙  输出：  Iliketoplayfootballchina  **【学生】**思考、讨论。 | **教师通过字符串的读写展示，让学生了解字符串的读写的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了字符串的读写，让学生会利用周围的资源解决实际问题，具备求知精神。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **把【program11\_3\_2.c】的文件打开方式换成“w+”，文件“input.txt”中的内容将会是什么？** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**数据块的读写  **任务描述**  将 3 个学生的考试成绩存入一个文件，并读取出来显示在屏幕上。  **预备知识**  C 语言提供了用于整块数据的读写函数，可用来读写一组数据，如一个数组，一个结构变量的值等。数据块的读写函数在处理文件中的数据时，是以字节数已经确定的一块数据为单位的，每次读取若干块数据。  **一、数据块读取函数 fread**  fread 函数调用的一般形式为：  fread( buffer,size,n,fp);  其中，buffer 为存放数据的内存区的首地址；size 为无符号整数，表示每块数据的字节数；n 也是无符号整数，表示每次读取的块数；fp 为文件指针，指向打开的可读文件。  【功能】从 fp 所指向的文件中读取 n 块字节数为 size 的数据块，将这些数据块存入 buffer 指定的内存区。假如函数运行正确，则返回 n 值，否则返回 NULL 值。  **二、数据块写入函数 fwrite**  fwrite 函数调用的一般形式为：  fwrite(buffer,size,n,fp);  其中，buffer为存放数据的内存区首地址；size为无符号整数，表示每块数据的字节数；n 也是无符号整数，表示每次读取的块数；fp 为文件指针，指向打开的可写文件。  【功能】将 buffer 指向的内存区域的 n 块字节数为 size 的数据块写入 fp 所指向的文件中。假如函数运行正确，则返回 n 值，否则返回 NULL 值。  【注意】  （1）读写数据块函数的对应文件可以是文本文件，也可以是二进制文件。  （2）size 表示数据块长度，一般用“sizeof（数据类型）”来确定。例如，一个构造型数据，其 size 为 sizeof（struc 结构名）。  （3）读写 n 个数据块后，文件中的位置指针会自动后移 n×size 个字节的位置。  **任务实施**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* 程序 : program 11\_4.c \*/  /\* 功能 : 将 3 个学生的考试成绩存入一个文件 , 并读取出来显示在屏幕上。 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #include <stdio.h>  struct stu  {  char name[10];  int num;  fl oat score;  }stu[3],stu1;  void main()  {  FILE \*fp;  int i;  fl oat score;  printf("input data:\n");  for(i=0;i<3;i++)  {  printf(" 输入第 %d 位同学的姓名、学号、成绩 : \n",i+1);  scanf("%s",stu[i].name);  scanf("%d",&stu[i].num);  scanf("%f",&score);  stu[i].score=score;  }  if((fp=fopen("stu.txt","wb+"))==NULL)  {  printf("cannot open fi le!");  exit(0);  }  fwrite(stu,sizeof(struct stu),3,fp);  fclose(fp);  if((fp=fopen("stu.txt","rb"))==NULL)  {  printf("cannot open fi le!");  exit(0);  }  printf("\nname\tnumber\tscore\n");  for(i=0;i<3;i++)  {  fread(&stu1,sizeof(struct stu),1,fp);  printf("%s\t%3d\t%.2f\n",stu1.name,stu1.num,stu1.score);  }  fclose(fp);  }  该程序首先定义了一个结构数组，用于存放学生的数据，然后输入学生的数据，利用 fwrite 函数将学生的数据写入文件 stu.txt 中。写文件完毕之后，将文件的位置指针指向文件的开头，然后利用fread函数将文件中的学生数据一个一个地读出并输出到屏幕上，这样就实现了存储学生成绩并显示的功能。  程序运行结果：  输入第 1 位同学的姓名、学号、成绩 :  iack 438 98.5 ↙  输入第 2 位同学的姓名、学号、成绩 :  rose 439 80.0 ↙  输入第 3 位同学的姓名、学号、成绩 :  mike 440 55.5 ↙  name number score  iack 438 98.50  rose 439 80.00  mike 440 55.50  **【学生】**思考、讨论。 | **教师通过数据块的读写展示，让学生了解数据块的读写的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了数据块的读写，能够统筹思考问题，具备整体规划能力，努力做一名解决问题的程序员。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **简述数据块读取函数 fread。** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**格式化读写  **任务描述**  利用 fscanf 函数和 fprintf 函数实现【任务 4】的 program11\_4.c。  **预备知识**  **一、格式化数据读取函数 fscanf**  调用 fscanf 函数的一般形式为：  fscanf( 文件指针 fp, 格式字符串 , 输入表列 );其中的格式字符串和输入表列与 scanf 函数中的格式字符串的形式一样。  【功能】从 fp 指向的文件中照格式读取数据。假如函数运行正确，则返回读取数据的数目，否则返回 EOF。  **二、格式化数据写入函数 fprintf**  fprintf 函数调用的一般形式为：  fprintf( 文件指针 fp, 格式字符串 , 输出表列 );  其中格式字符串和输出表列与 printf 函数中的格式字符串的形式一样。  【功能】按照格式字符串的指定格式，将输出表列中对应的地址的数据写入 fp 指向  的文件中。假如函数运行正确，则返回读取数据的数目，否则返回 EOF。  **任务实施**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* 程序：program 11\_5.c \*/  /\* 功能：利用 fscanf 函数和 fprintf 函数实现【任务 4】的 program 11\_4.c。 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #include <stdio.h>  struct stu  {  char name[10];  int num;  fl oat score;  }stu[3]，stu1[3];  void main()  {  FILE \*fp;  int i;  fl oat score;  printf("input data\n");  for(i=0;i<3;i++)  {  printf(" 输入第 %d 位同学的姓名、学号、成绩 :\n"，i+1);  scanf("%s"，stu[i].name);  scanf("%d"，&stu[i].num);  scanf("%f"，&score);  stu[i].score=score;  }  if((fp=fopen("stu.txt"，"wb+"))==NULL)  {  printf("cannot open fi le!");  exit(0);  }  for(i=0;i<3;i++)  fprintf(fp，"%s %d %f"，stu[i].name， stu[i].num，stu[i].  score);  fclose(fp);  if((fp=fopen("stu.txt"，"rb"))==NULL)  {  printf("cannot open fi le!");  exit(0);  }  printf("\nname\tnumber\tscore\n");  for(i=0;i<3;i++)  {  fscanf( fp，"%s %d %f"， stu1[i].name， &stu1[i].num，  &stu1[i].score);  printf("%s\t%3d\t%.2f\n"，stu1[i].name，stu1[i].num，  stu1[i].score);  }  fclose(fp);  }  【说明】与 program 11\_4.c 相比，本程序中 fscanf 和 fprintf 函数每次只能读写一个结构数组元素，因此采用了循环语句来读写全部数组元素。在使用fscanf和fprintf函数时，要特别注意格式字符串一定要对应，否则很容易出错。  **【学生】**思考、讨论。 | **教师通过格式化读写展示，让学生了解格式化读写的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了格式化读写，让学生能遵守各种编程规范，具备规范化、标准化的代码编写习惯。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **简述格式化数据如何写入函数 fprintf。** | 通过课后练习，使学生巩固所学新知识 |
| **知识讲解**（45min） | **【教师】**文件定位  **预备知识**  前面介绍的对文件的读写方式都是顺序读写，即只能从头开始顺序地读写各个数据。  但在实际问题中，常要求只读写文件中某一指定部分。为了解决这个问题，可通过移动文件内部的位置指针，将其指示到需要读写的位置再进行读写，这种读写称为随机读写。  实现随机读写的关键是按要求移动位置指针，称为文件的定位。文件定位函数主要有两个，即 rewind 函数和 fseek 函数。  **一、文件的定位函数**  1. rewind 函数  rewind 函数的调用形式为：  rewind( 文件指针 fp);  rewind 函数的作用是使位置指针指向文件的开头，该函数没有返回值。  2. fseek 函数  fseek 函数用来移动文件内部位置指针，其调用形式为：  fseek( 文件指针 fp, 位移量 , 起始点 );其中，“文件指针 fp”指向被移动的文件。“位移量”表示移动的字节数，要求位移量是 long 型数据，以便在文件长度大于 64KB 时不会出错。当用常量表示位移量时要求加后缀“L”。“起始点”表示从何处开始计算位移量，规定的起始点有 3 种：文件开头、当前位置和文件末尾。  例如：  　　fseek(fp,10L,0);  其含义是把位置指针移到离文件首部 10 个字节处。  【注意】fseek 函数一般用于二进制文件，因为在文本文件中要进行转换，故往往计算的位置会出现错误。文件的随机读写在移动位置指针之后，即可用前面介绍的一种读写函数进行读写，由于一般是读写一个数据块，因此常用 fread 和 fwrite 函数。  **二、文件检测函数**  在文件中有一些文件检测函数，用于检测当前文件指针的指向或文件是否出错。  1. 文件结束检测函数 feof  函数调用格式：  feof( 文件指针 fp);  【功能】判断文件是否处于文件结束位置，如果文件结束，则返回值为 1，否则为 0。  2. 测试当前位置函数 ftell  函数调用格式：  ftell( 文件指针 fp);  【功能】判断文件位置指针所指向的位置，如函数运行正确，则返回 fp 所指文件的位置指针相对于文件首部偏移的字节数，否则返回 -1。  3. 读写文件出错检测函数 ferror  函数调用格式：  ferror( 文件指针 fp);  【功能】检查文件在用各种输入输出函数进行读写时是否出错。如 ferror 返回值为0 表示未出错，否则表示有错。  **【学生】**思考、讨论。 | **教师通过文件定位展示，让学生了解文件定位的基本理论知识。** |
| **课堂小结**  （3min） | 【**教师**】**回顾和总结本节课的知识点。**  **这节课我们一起学习了文件定位，具备一定的纠错检错能力，不断提高个人编程水平。** | 通过对所学知识的回顾，培养学生的归纳总结能力 |
| **作业布置**（2min） | 【**教师**】**布置课后作业**  **建立一个学生电话簿的数据文件，并读取其中的数据。** | 通过课后练习，使学生巩固所学新知识 |
| **教学反思** | 加强机房纪律和机器使用道德方面的教育，使学生不仅在班级集中注意力学习，在机房上机时也要按规定严格约束自己。 | |