

## 第9章 结构

### 9.1 教学要点

本章主要介绍结构的基本概念与定义（含嵌套结构）、结构变量的定义和使用、结构数组的应用、结构指针的概念和使用等知识。其中，重点介绍结构的定义与成员引用方式、结构数组的应用、结构指针的基本概念与使用，而嵌套结构的定义、结构指针作为函数参数是难点。

9.1 节通过综合示例程序“输出平均分最高的学生信息”，引出结构的概念和主要知识点，教师在讲授时，先介绍和分析该示例程序的总体框架结构，结合程序把各知识点引出即可，不需要对源代码进行详细分析，可以让学生课后自己细读。然后重点介绍结构的概念，并与数组进行对比，并举例说明结构定义（包括嵌套结构）的具体方法。结构变量主要介绍定义、初始化以及使用方式。在介绍结构变量的定义时，要举例讲解三种定义方式。介绍结构变量初始化时，要讲清楚结构变量数据在内存中的存储情形。在介绍结构变量的使用时，重点介绍三种情况：结构成员引用方式、相同类型结构变量的相互赋值、结构变量作为函数参数。

9.2 节通过示例程序“学生成绩排序”，介绍结构数组的概念、定义、初始化、成员引用以及应用编程。要结合示例重点向学生讲清楚结构数组的概念以及结构数组成员的引用方式。

9.3 节通过示例程序“修改学生成绩”，重点介绍结构指针的概念，以及通过结构指针变量间接访问数据的方式，特别是通过对比介绍结构指针指向运算符“->”的使用方法。此外，还要结合示例详细介绍结构指针作为函数参数的使用方法，并分析其特点。

讲授学时：4 学时，实验学时同讲授学时。

本章的知识能力结构图见图 9.1。

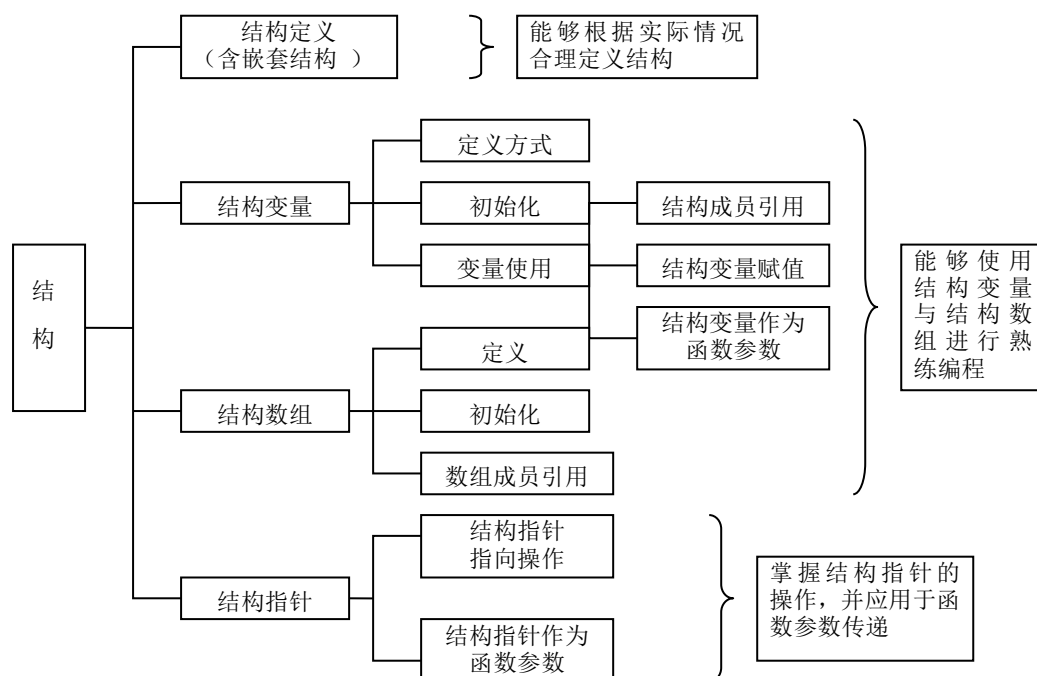






图 9.1 知识能力结构图

## 9.2 讲稿

1	 <b>Chap 9 结构</b>  <b>9.1 输出平均分最高的学生信息</b> <b>9.2 学生成绩排序</b> <b>9.3 修改学生成绩</b>	本章分 3 节。 通过示例程序引导的方式分别介绍结构相关的知识。
2	 <b>本章要点</b> <ul style="list-style-type: none"><li>■ 什么是结构？结构与数组有什么差别？</li><li>■ 有几种结构的定义形式，它们之间有什么不同？</li><li>■ 什么是结构的嵌套？</li><li>■ 什么是结构变量和结构成员变量？如何引用结构成员变量？</li><li>■ 结构变量如何作为函数参数使用？</li><li>■ 什么是结构数组？如何定义和使用结构数组？</li><li>■ 什么是结构指针？它如何实现对结构分量的操作？</li><li>■ 结构指针是如何作为函数的参数的？</li></ul>	提出本章的学习要点。
3	 <b>9.1输出平均分最高的学生信息</b>  <b>9.1.1 程序解析</b> <b>9.1.2 结构的概念与定义</b> <b>9.1.3 结构的嵌套定义</b> <b>9.1.4 结构变量的定义和初始化</b> <b>9.1.5 结构变量的使用</b>	通过示例程序“输出平均分最高的学生信息”，介绍结构的概念与一般定义方式，结构的嵌套定义方法，结构变量的定义、初始化和使用。
4	 <b>9.1.1 程序解析</b>  <b>例9-1 输出平均分最高的学生信息</b> <ul style="list-style-type: none"><li>■ 假设学生的基本信息包括学号、姓名、三门课程成绩以及个人平均成绩。输入 n 个学生的成绩信息，计算并输出平均分最高的学生信息。</li></ul> <div>运行结果</div> <pre>Input n: 3 Input the student's number, name and course scores No. 1: 101 Zhang 78 87 85 No. 2: 102 Wang 91 88 90 No. 3: 103 Li 75 90 84 num: 102, name: Wang, average: 89.67</pre>	介绍示例程序“输出平均分最高的学生信息”。 在课堂上无需详细的代码分析，只要运行演示该程序，并分析程序的整体结构和结构相关语句即可，可以让学生在课后自己认真细读此示例程序。

5	<div> <div></div> <div>9.1.1 程序解析</div> </div> <pre> #include&lt;stdio.h&gt; struct student{     int num;          /* 学号 */     char name[10];    /* 姓名 */     int computer, english, math; /* 三门课程成绩 */     double average;   /* 个人平均成绩 */ }; </pre>	展示结构类型的定义。
6	<div> <div></div> <div></div> </div> <pre> int main(void) {     int i, n;     struct student s1,max; /* 定义结构变量 */     printf("Input n: ");     scanf("%d", &amp;n);     printf("Input the student's number, name and course scores\n");     for(i = 1; i &lt;= n; i++){         printf("No.%d: ", i);         scanf("%d%s%d%d%d",&amp;s1.num,s1.name,&amp;s1.math,&amp;s1.english,&amp;s1.computer);         s1.average=(s1.math + s1.english + s1.computer) / 3.0;         if(i == 1) max = s1; /* 结构变量操作 */         if(max.average &lt; s1.average)             max = s1;     }     printf("num:%d, name:%s, average:%.2lf\n", max.num, max.name, max.average);     return 0; } </pre>	介绍程序中结构变量的定义与使用语句。结构的概念和相关定义语法规则在后面展开介绍。
7	<div> <div></div> <div>9.1.2 结构的概念与定义</div> </div> <div> <div>■ 使用结构来表示学生信息：</div> <div> <pre> struct student{     int num;          /* 学号 */     char name[10];    /* 姓名 */     int computer, english, math; /* 三门课程成绩 */     double average;   /* 个人平均成绩 */ }; </pre> </div> </div> <div> <div>■ 结构是C语言中一种新的构造数据类型，它能够把有内在联系的不同类型的数据统一成一个整体，使它们相互关联</div> <div>■ 结构又是变量的集合，可以按照对基本数据类型的操作方法单独使用其变量成员。</div> </div> <div> <div>结构与数组比较：</div> <div> <div>•都是构造类型，是多个变量的集合</div> <div>•数组成员类型相同，结构成员类型不同</div> </div> </div>	介绍结构的概念，指出：结构是C语言中一种新的构造数据类型，能够将不同类型的数据统一成一个整体，每个成员都是一个变量，可以单独操作。 分析结构与数组的区别：数组成员类型必须相同，而结构成员类型可以不同。
8	<div> <div></div> <div>9.1.2 结构的概念与定义</div> </div> <div> <div>■ 结构类型定义的一般形式为：</div> <div> <pre> struct 结构名 {     类型名 结构成员名1;     类型名 结构成员名2;     ...     类型名 结构成员名n; }; </pre> </div> </div> <div> <div>关键字struct和它后面的结构名一起组成一个新的数据类型名</div> <div>结构的定义以分号结束，C语言中把结构的定义看作是一条语句</div> </div>	要说清楚结构类型定义中各部分的含义。关键字 struct 和它后面的结构名一起组成一个新的数据类型名。结构的定义以分号结束，C语言中把结构的定义看作是一条语句。

9	<div><div><div>9.1.2 结构的概念与定义</div><div><div><div>■ 例如，平面坐标结构： struct point {     float x;     float y; };</div><div><div>•虽然x、y的类型相同，也可以用数组的方式表示，但采用结构进行描述，更贴近事物本质，从而增加了程序的可读性，使程序更易理解</div><div>•结构比较适合用于描述具有多个属性的实体或对象</div></div></div></div></div></div>	<div><div>举例说明结构的定义，并对比分析采用结构方式来描述事物，表述更加准确，更贴近事物本质，从而增强程序的可读性，易于理解。</div><div>进一步分析和总结什么情况下更适合用结构来描述事物。</div></div>																				
10	<div><div><div>9.1.3 结构的嵌套定义</div><div><div><div>■ 在我们的实际生活中，一个较大的实体可能由多个成员构成，而这些成员中有些又有可能是由一些更小的成员构成。</div><div>■ 在学生信息中可以再增加一项：“通信地址”，它又可以再划分为：城市、街道、门牌号、邮政编码。</div></div><div><table><tr><td>学号</td><td>姓名</td><td colspan="4">通信地址</td><td>计算机</td><td>英语</td><td>数学</td><td>平均成绩</td></tr><tr><td></td><td></td><td>城市</td><td>街道</td><td>门牌号</td><td>邮编</td><td></td><td></td><td></td><td></td></tr></table></div></div></div></div>	学号	姓名	通信地址				计算机	英语	数学	平均成绩			城市	街道	门牌号	邮编					<div><div>举例说明什么情况下需要使用嵌套的结构。</div></div>
学号	姓名	通信地址				计算机	英语	数学	平均成绩													
		城市	街道	门牌号	邮编																	
11	<div><div><div>9.1.3 结构的嵌套定义</div><div><div><div>■ 由此，我们可以对其结构类型进行如下重新定义：</div><div><div><div>struct address{     char city[10];     char street[20];     int code;     int zip; };</div><div>struct nest_student{     int num;     char name[10];     struct address addr;     int computer, english, math;     double average; };</div></div><div><div>•在定义嵌套的结构类型时，必须先定义成员的结构类型，再定义主结构类型。</div></div></div></div></div></div></div>	<div><div>需要特别提醒学生，在定义嵌套的结构类型时，必须先定义成员的结构类型，再定义主结构类型。</div></div>																				
12	<div><div><div>9.1.4 结构变量的定义和初始化</div><div><div><div>■ 在C语言中定义结构变量的方式有三种：</div><div><div>1.单独定义：先定义一个结构类型，再定义一个具有这种结构类型的变量</div><div><div>struct student{     int num;     char name[10];     int computer, english, math;     double average; }; struct student s1,s2;</div></div></div></div></div></div></div>	<div><div>介绍结构变量定义的三种方式。</div><div>特别指出 struct student 是类型名，student 是结构名，s1、s2 是结构变量名。</div></div>																				

13	<div><div></div><div><h3>9.1.4结构变量的定义和初始化</h3><p>2. 混合定义：在定义结构类型的同时定义结构变量</p><pre>struct student{     int num;           /* 学号 */     char name[10];      /* 姓名 */     int computer, english, math; /* 三门课程成绩 */     double average;     /* 个人平均成绩 */ }s1, s2;</pre><p>3. 无类型名定义：在定义结构变量时省略结构名</p><pre>struct {     int num;           /* 学号 */     char name[10];      /* 姓名 */     int computer, english, math; /* 三门课程成绩 */     double average;     /* 个人平均成绩 */ } s1, s2;</pre></div></div>	要给学生分析一下，第2种和第3种定义方式的差别，即省略结构名的定义方式具有一定的局限性。																		
14	<div><div></div><div><h3>9.1.4结构变量的定义和初始化</h3><p>■ 结构变量的初始化</p><pre>struct student s1 = {101, "Zhang", 78, 87, 85};</pre><table><tr><td>num</td><td>name</td><td>computer</td><td>english</td><td>math</td><td>average</td></tr><tr><td>↓</td><td>↓</td><td>↓</td><td>↓</td><td>↓</td><td>↓</td></tr><tr><td>101</td><td>Zhang</td><td>78</td><td>87</td><td>85</td><td></td></tr></table></div></div>	num	name	computer	english	math	average	↓	↓	↓	↓	↓	↓	101	Zhang	78	87	85		介绍结构变量的初始化格式。 分析初始化后，结构变量数据在内存中的存放情况。
num	name	computer	english	math	average															
↓	↓	↓	↓	↓	↓															
101	Zhang	78	87	85																
15	<div><div></div><div><h3>9.1.5 结构变量的使用</h3><p>1. 结构变量成员的引用</p><p>■ 在C语言中，使用结构成员操作符“.”来引用结构成员，格式为：</p><p>结构变量名 . 结构成员名</p><pre>s1.num = 101; strcpy(s1.name, "Zhang"); nest_s1.addr.zip = 310015;</pre></div></div>	介绍结构成员运算符“.”的使用方式，并举例说明。																		
16	<div><div></div><div><h3>9.1.5 结构变量的使用</h3><p>2. 结构变量的整体赋值</p><p>■ 具有相同类型的结构变量可以直接赋值。赋值时，将赋值符号右边结构变量的每一个成员的值都赋给了左边结构变量中相应的成员。</p><pre>struct student s1 = {101, "Zhang", 78, 87, 85}, s2; s2 = s1;</pre></div></div>	特别指出：只有具有相同类型的结构变量才可以相互赋值，并说明赋值后的效果。																		


17	<div> <div></div> <div> <h3>9.1.5 结构变量的使用</h3> <h4>3. 结构变量作为函数参数</h4> <ul style="list-style-type: none"> <li>■ 如果一个C程序的规模较大，要以函数的形式进行功能模块的划分和实现</li> <li>■ 如果程序中含有结构数据，则就可能需要用结构变量作为函数的参数或返回值，以在函数间传递数据。</li> <li>■ 例：  <pre>double count_average( struct student s ); main(): s1.average = count_average ( s1 );</pre> </li> </ul> </div> <div> <p><b>特点：</b>可以传递多个数据且参数形式较简单</p> <p><b>缺点：</b>对于成员较多的大型结构，参数传递时进行的结构数据复制使得效率较低</p> </div> </div>	<p>说明结构变量作为函数参数的可能和必要性。</p> <p>分析结构变量作为函数参数的优缺点。</p>
18	<div> <div></div> <div> <h2>9.2 学生成绩排序</h2> <ul style="list-style-type: none"> <li>■ 9.2.1 程序解析</li> <li>■ 9.2.2 结构数组操作</li> </ul> </div> </div>	<p>通过示例程序“学生成绩排序”介绍结构数组相关的知识。</p>
19	<div> <div></div> <div> <h3>9.2.1 程序解析</h3> <p>例9-2 输入n (n&lt;50) 个学生的成绩信息，按照学生的个人平均成绩从高到低输出他们的信息。</p> <pre>struct student students[50], temp; /* 定义结构数组 */  /* 输入 */ ...</pre> </div> </div>	<p>介绍示例程序功能</p> <p>运行演示程序，介绍总体流程</p> <p>介绍结构数组 students 的定义语句</p> <p>说明该示例程序中借用了例 9-2 中的自定义函数 count_average(), 并介绍其调用方式</p>
20	<div> <div></div> <div> <h3>9.2.1 程序解析</h3> <pre>/* 结构数组排序，选择排序法 */ for( i = 0; i &lt; n-1; i++) {     index = i;     for( j = i+1; j &lt; n; j++)         if (students[j].average &gt; students[index].average) /* 比较平均成绩 */             index = j;     temp = students[index];          /* 交换数组元素 */     students[index] = students[i];     students[i] = temp; } /* 输出排序后的信息 */ printf("num\t name\t average\n"); for( i = 0; i &lt; n; i++)     printf("%d\t %s\t %.2f\n", students[i].num, students[i].name, students[i].average);</pre> </div> </div>	<p>简要介绍结构数组的排序</p> <p>简要介绍结构数组元素的输出</p>

21	<div><div></div><div>9.2.2 结构数组操作</div></div> <ul style="list-style-type: none"><li>一个结构变量只能表示一个实体的信息，如果有许多相同类型的实体，就需要使用结构数组。</li><li>结构数组是结构与数组的结合，与普通数组的不同之处在于<b>每个数组元素都是一个结构类型的变量</b>。</li></ul>	<p>介绍结构数组的概念。</p> <p>结构数组是结构与数组的结合，每个数组元素都是一个结构类型的变量。</p> <p>结构数组可以用于描述多个相同结构类型的个体。</p>																								
22	<div><div></div><div>9.2.2 结构数组操作</div></div> <ul style="list-style-type: none"><li>结构数组的定义方法与结构变量类似 <b>struct student students[50];</b></li></ul> <div><div>结构数组<b>students</b>，它有<b>50</b>个数组元素，从<b>students[0]</b>到<b>students[49]</b>，每个数组元素都是一个结构类型<b>struct student</b>的变量</div></div>	<p>结构数组的定义格式。</p> <p>结构数组中每个数组元素都是一个结构变量，每个变量又都包含多个成员。</p>																								
23	<div><div></div><div>9.2.2 结构数组操作</div></div> <ul style="list-style-type: none"><li>结构数组的初始化 struct student students[50] = { { 101,"zhang", 76, 85, 78 }, {102, "wang", 83, 92, 86} };</li></ul> <div><div>students[0]</div><div>students[1]</div><div>...</div><div>students[49]</div></div> <table><tr><td>101</td><td>Zhang</td><td>76</td><td>85</td><td>78</td><td></td></tr><tr><td>102</td><td>Wang</td><td>83</td><td>92</td><td>86</td><td></td></tr><tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	101	Zhang	76	85	78		102	Wang	83	92	86		...	...	...	...	...								<p>介绍结构数组的初始化方式。</p> <p>分析初始化后，结构数组数据在内存中的存放情况。</p>
101	Zhang	76	85	78																						
102	Wang	83	92	86																						
...	...	...	...	...																						
24	<div><div></div><div>9.2.2 结构数组操作</div></div> <ul style="list-style-type: none"><li>结构数组元素的成员引用，其格式为： <b>结构数组名[下标]. 结构成员名</b></li><li>使用方法与同类型的变量完全相同： students[i].num = 101; strcpy(students[i].name, "zhang"); students[i] = students[k]</li></ul>	<p>结合示例介绍结构数组元素的成员引用方法。</p> <p>说明：一个结构数组元素就是一个结构变量。</p>																								

25	<div> <div></div> <div> <h3>9.3 修改学生成绩</h3> <h4>9.3.1 程序解析</h4> <h4>9.3.2 结构指针的概念</h4> <h4>9.3.3 结构指针作为函数参数</h4> </div> </div>	<p>通过示例程序“修改学生成绩”介绍结构指针的概念、操作方式、结构指针作为函数参数等知识。</p>
26	<div> <div></div> <div> <h4>9.3.1 程序解析</h4> <p><b>例9-3</b> 输入n(n&lt;50)个学生的成绩信息，再输入一个学生的学号、课程以及成绩，在自定义函数中修改该学生指定课程的成绩。</p> <pre> int main(void) {     int course, i, n, num, pos, score;     struct student students[50]; /* 定义结构数组 */     ... /* 输入n个学生信息 */     ... /* 输入待修改学生信息 */     /*调用函数，修改学生成绩*/     pos = update_score(students, n, num, course, score);     ... /*输出修改后的学生信息*/ ... } </pre> </div> </div>	<p>介绍示例程序功能要求 运行演示程序，介绍程序的主流程 重点说明自定义函数 update_score()的功能，以及结构数组 students 作为其实参的情况</p>
27	<div> <div></div> <div> <h4>9.3.1程序解析</h4> <p>/* 自定义函数，修改学生成绩 */  int update_score(struct student *p, int n, int num, int course, int score)  {      int i, pos;      for(i = 0; i &lt; n; i++, p++) /* 按学号查找 */          if(p-&gt;num == num)              break;      if(i &lt; n) /* 找到，修改成绩 */      {          switch(course){              case 1: p-&gt;math = score; break;              case 2: p-&gt;english = score; break;              case 3: p-&gt;computer = score; break;          }          pos = i; /* 被修改学生在数组中的下标 */      }      else /* 无此学号 */          pos = -1;      return pos;  }  </p> </div> </div>	<p>重点介绍自定义函数 update_score()的形参，以及通过结构指针 p 对结构数组内容进行访问和操作情况</p>
28	<div> <div></div> <div> <h4>9.3.2结构指针的概念</h4> <ul style="list-style-type: none"> <li>■ 指针可以指向任何一种变量，而结构变量也是C语言中的一种合法变量，因此，指针也可以指向结构变量，这就是结构指针。</li> <li>■ 结构指针就是指向结构类型变量的指针</li> </ul> </div> </div>	<p>结构指针的概念：结构指针就是指向结构类型变量的指针。</p>



29	<div data-bbox="300 203 347 232" data-label="Image"></div> <h3 data-bbox="331 241 596 275">9.3.2结构指针的概念</h3> <pre data-bbox="375 322 730 398"> struct student s1 = {101, "zhang", 78, 87, 85}, *p; p = &amp;s1; </pre> <div data-bbox="357 470 721 501" data-label="Diagram"> </div>	<p>介绍结构指针的定义和赋值</p> <p>结构图示说明结构指针赋值后的指向情况</p>
30	<div data-bbox="300 620 347 649" data-label="Image"></div> <h3 data-bbox="331 658 596 692">9.3.2结构指针的概念</h3> <ul data-bbox="331 725 753 891" style="list-style-type: none"> <li>■ 结构指针的使用</li> <li>■ (1) 用*p访问结构成员。如： <code>(*p).num = 101;</code></li> <li>■ (2) 用指向运算符“-&gt;”访问指针指向的结构成员。如： <code>p-&gt;num = 101;</code></li> </ul> <div data-bbox="539 857 778 983" data-label="Text"> <p>当p指向结构变量s1时，下面三条语句的效果是一样的：</p> <pre> s1.num = 101; (*p).num = 101; p-&gt;num = 101; </pre> </div>	<p>介绍如何使用结构指针对指向的数据进行操作。</p> <p>重点介绍运算符“-&gt;”的使用方法。</p> <p>对不同的结构指针使用方式进行比较。</p>
31	<div data-bbox="300 1037 347 1066" data-label="Image"></div> <h3 data-bbox="331 1075 683 1108">9.3.3结构指针作为函数参数</h3> <ul data-bbox="331 1120 798 1191" style="list-style-type: none"> <li>■ 结构指针的操作是非常灵活的，如果将结构指针作为函数的参数，可以完成比基本类型指针更为复杂的操作。</li> </ul> <div data-bbox="331 1198 798 1368" data-label="Text"> <p>■ 例9-3</p> <pre> main(): pos = update_score(students, n, num, course, score); 自定义函数: int update_score(struct student *p, int n, int num, int course, int score) </pre> <div data-bbox="582 1171 794 1263" data-label="Text"> <p>函数update_score运行完毕返回主函数后，主函数中的结构数组students中的值已被修改</p> </div> </div>	<p>结合例 9-3，详细分析结构指针作为函数参数的编程方法。包括形参如何书写，实参如何设定，参数传递含义。</p> <p>要特别指出：函数 update_score 运行完毕返回主函数后，主函数中的结构数组 students 中的成员值已经被结构指针通过间接访问的方式修改了。</p>
32	<div data-bbox="300 1453 347 1482" data-label="Image"></div> <h3 data-bbox="331 1491 683 1525">9.3.3结构指针作为函数参数</h3> <ul data-bbox="331 1559 778 1751" style="list-style-type: none"> <li>■ 与结构变量作为函数参数相比，用结构指针作为函数参数的效率更高。</li> <li>■ 就例 9-3 而言，在函数update_score()中需要修改主函数中结构数组students的数据，根据第8章介绍的知识，在此处也只能使用指针作为函数参数的方式才能通过间接访问操作来实现程序功能。</li> </ul>	<p>与结构变量作为函数参数进行比较，总结结构指针作为函数参数的特点和优势。</p> <p>说明：为实现例 9-3 修改学生成绩的功能，不能采用结构变量作为函数参数的方式，而只能用结构指针作为函数参数的方式。</p>

33	<div> <div>  <h2>本章总结</h2> <ul style="list-style-type: none"> <li>■ 结构的概念与定义（含嵌套结构）</li> <li>■ 结构变量 <ul style="list-style-type: none"> <li>□ 定义</li> <li>□ 初始化</li> <li>□ 使用（成员引用、相互赋值、作为函数参数）</li> </ul> </li> <li>■ 结构数组 <ul style="list-style-type: none"> <li>□ 定义、初始化、结构数组成员引用</li> </ul> </li> <li>■ 结构指针 <ul style="list-style-type: none"> <li>□ 概念</li> <li>□ 结构指针操作</li> <li>□ 结构指针作为函数参数</li> </ul> </li> </ul> </div> <div> <ul style="list-style-type: none"> <li>• 能够根据实际情况合理定义结构</li> <li>• 能够使用结构变量与结构数组进行熟练编程</li> <li>• 掌握结构指针的操作，并应用于函数参数传递</li> </ul> </div> </div>	回顾和总结本章的教学要点，提出能力要求。
----	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------

## 9.3 练习与习题参考答案

### 9.3.1 练习参考答案

9-1 定义一个能够表示复数的结构类型，一个复数包括实数与虚数两个部分。

解答：

```
struct complex {
    float real;
    float imaginary;
};
```

9-2 人的出生日期由年、月、日组成，请在例 9-1 中的学生信息结构中增加一个成员：出生日期，用嵌套定义的方式重新定义该结构类型。

解答：

```
struct date {
    int year;
    int month;
    int day;
};
struct nest_student {
    int num;
    char name[10];
    struct date birthday; /* 定义出生日期 */
    int computer, english, math;
    double average;
};
```

9-3 例9-1中，如果要计算的是三门课程的课程平均分，应该如何改写程序？

解答：

```
int main(void)
{
    int i, n;
```

```

double math_ave, english_ave, computer_ave;
struct student s1; /* 定义结构变量 */
printf("Input n: ");
scanf("%d", &n);
printf("Input the student's number, name and course scores\n");
math_ave = english_ave = computer_ave = 0;
for(i = 1; i <= n; i++){
    printf("No.%d: ", i);
    scanf("%d%s%d%d%d",&s1.num,s1.name,&s1.math,&s1.english,&s1.computer);
    math_ave = s1.math;
    english_ave = s1.english;
    computer_ave = s1.computer;
}
printf("math_ave:%.2lf, english_ave:%.2lf, computer_ave:%.2lf\n", math_ave, english_ave,
computer_ave);
return 0;
}

```

9-4 定义一个包含 5 名学生信息的结构数组，并对该结构数组的所有元素进行初始化。

解答：

```

struct student{
    int num;
    char name[10];
    int computer, english, math;
    double average;
};
struct student s[5]={ {30101, "张一",69,75,84 }, {30132, "李二",80,85,90}, {40231, "王三",71,91,74}, {40754, "赵四",65,76,87}, {50426, "刘五",81,92,73} };

```

9-5 参考例 9-2，输入并保存 10 个学生的成绩信息，分别输出平均成绩最高和最低的学生信息。

解答：

```

#include<stdio.h>
struct student{ /*学生信息结构定义*/
    int num; /* 学号 */
    char name[10]; /* 姓名 */
    int computer, english, math ; /* 三门课程成绩 */
    double average; /* 个人平均成绩 */
};
struct student students[10]; /* 定义结构数组 */
int main(void)
{
    int i, max,min;
    /* 输入 10 个学生的记录*/
    for(i = 0; i < 10; i++){
        printf("No %d: ", i+1);
        scanf("%d%s%d%d%d", & students[i].num, students[i].name, &students[i].math,
&students[i].english, &students[i].computer);
        students[i].average=(students[i].math+students[i].english+students[i].computer)/3.0;
    }
}

```

```

    }
    /* 查找平均成绩最高、最低分学生的数组下标值 */
    max=min=0;
    for( i = 1; i < 10; ++i ){
        if(students[i].average > students[max].average)  max=i;
        if(students[i].average < students[min].average)  min=i;
    }
    /*输出平均成绩最高和最低的学生信息*/
    printf("最高分学生信息：学号:%d,姓名:%s,数学:%d,英语:%d,计算机:%d,平均分 :%.2lf \n", students[max].num, students[max].name, students[max].math, students[max].english, students[max].computer, students[max].average);
    printf("最低分学生信息：学号:%d,姓名:%s,数学:%d,英语:%d,计算机:%d,平均分 :%.2lf \n", students[min].num, students[min].name, students[min].math, students[min].english, students[min].computer, students[min].average);
    return 0;
}

```

9-6 定义一个 struct student 类型的结构指针，用其实现一个学生信息的输入和输出。

解答：

```

struct student{          /*学生信息结构定义*/
    int num;              /* 学号 */
    char name[10];        /* 姓名 */
    int computer, english, math; /* 三门课程成绩 */
    double average;       /* 个人平均成绩 */
}s, *p;
p = &s;
scanf("%d%s%d%d%d", &p->num, p->name, &p->math, &p->english, &p->computer);

```

9-7 改写例 9-3 中的函数 update\_score()，将第一个形参改为结构数组形式。

解答：

```

int update_score(struct student s[ ], int n, int num, int course, int score)
{
    int i,pos;
    for(i = 0; i < n; i++) /* 按学号查找 */
        if(s[i].num == num)
            break;
    if(i < n) /* 找到，修改成绩 */
    {
        switch(course){
            case 1: s[i].math = score; break;
            case 2: s[i].english = score; break;
            case 3: s[i].computer = score; break;
        }
        pos = i; /* 被修改学生在数组中的下标 */
    }
}

```

```

    }
    else                                /* 无此学号 */
        pos = -1;
    return pos;
}

```

### 9.3.2 习题参考答案

#### 一、选择题

1	2	3	4	5
D	B	C	B	A

#### 二、填空题

1. 成员（分量）      指向
2. stud[i].name
3. &time.hour, &time.minute, &time.second  
time.second    time.minute++    time.hour == 24
4.    1    2    A    B
5.    23, wang, 98.5, wang

#### 三、程序设计

1. 时间换算：用结构类型表示时间内容（时间以时、分、秒表示），输入一个时间数值，再输入一个秒数  $n$  ( $n < 60$ )，以  $h:m:s$  的格式输出该时间再过  $n$  秒后的时间值（超过 24 点就从 0 点开始计时）。试编写相应程序。

解答：

```

#include <stdio.h>
int main(void)
{
    int n;
    int repeat, ri;
    struct time{
        int hour, minute, second;
    }time;
    scanf("%d:%d:%d", &time.hour, &time.minute, &time.second);
    scanf("%d",&n);
    time.second += n;
    if(time.second >= 60){
        time.minute += time.second / 60;
        time.second %= 60;
        if(time.minute >= 60){
            time.hour += time.minute / 60;
            time.minute %= 60;
            if(time.hour >= 24)
                time.hour %= 24;
        }
    }
}

```

```

    }
}
if(time.hour<10) printf("0%d:",time.hour);
else printf("%d:",time.hour);
if(time.minute<10) printf("0%d:",time.minute);
else printf("%d:",time.minute);
if(time.second<10) printf("0%d",time.second);
else printf("%d",time.second);
return 0;
}

```

2. 计算两个复数之积：编写程序，利用结构变量求解两个复数之积。

提示：求解  $(a_1+a_2i) \times (b_1+b_2i)$ ，乘积的实部为： $a_1 \times b_1 - a_2 \times b_2$ ，虚部为： $a_1 \times b_2 + a_2 \times b_1$ 。

解答：

```

#include <stdio.h>
struct complex{
    int real;
    int imag;
};
struct complex multiply(struct complex x, struct complex y) {
    struct complex product;
    product.real = x.real * y.real - x.imag * y.imag;
    product.imag = x.real * y.imag + x.imag * y.real;
    return product;
}
int main(void)
{
    struct complex product, x, y;
    scanf("%d%d%d%d", &x.real, &x.imag, &y.real, &y.imag);
    product = multiply(x, y);
    printf("(%d+%di) * (%d+%di) = %d + %di\n", x.real, x.imag, y.real, y.imag,
product.real, product.imag);
    return 0;
}

```

3. 平面向量加法：输入两个二维平面向量  $V_1=(x_1, y_1)$  和  $V_2=(x_2, y_2)$  的分量，计算并输出两个向量的和向量。试编写相应程序。

解答：

```

#include <stdio.h>
struct vector{
    double x;
    double y;
};
int main(void)

```

```

{
    struct vector v1, v2;
    double a,b;
    scanf("%f%f%f%f", &v1.x, &v1.y, &v2.x, &v2.y);
    a=  v1.x+ v2.x;
    b=  v1.y+v2.y;
    if(a>0)
        a=(int)((a+0.05)*10);
    else
        a=(int)((a-0.05)*10);
    a=a/10.0;
    if(b>0)
        b=(int)((b+0.05)*10);
    else
        b=(int)((b-0.05)*10);
    b=b/10.0;
    printf("(%.1f, %.1f)\n",a,b);
    return 0;
}

```

4. 查找书籍：从键盘输入 10 本书的名称和定价并存入结构数组中，从中查找定价最高和最低的书的名称和定价，并输出。试编写相应程序。

解答：

```

#include<stdio.h>
#define NUMBER 10
struct book
{ char name[30];
  float price;
};
int main()
{
    int i,maxl,minl,n;
    struct book test[NUMBER];
    scanf("%d",&n);

    for(i=0; i<n; i++){
        getchar();
        gets(test[i].name);
        scanf("%f",&test[i].price);
    }
    if(n<=0) return 0;
    maxl=minl=0;
    for(i=1; i<n; i++)
    { if(test[maxl].price<test[i].price)    maxl=i;

```

```

        if(test[minl].price > test[i].price)    minl=i;
    }
    printf("%.2f, %s\n", test[maxl].price, test[maxl].name);
    printf("%.2f, %s\n", test[minl].price, test[minl].name);
    return 0;
}

```

5. 通讯录排序：建立一个通讯录，通信录的结构记录包括：姓名、生日、电话号码，其中生日又包括三项：年、月、日。编写程序，定义一个嵌套的结构类型，输入 n(n<10) 个联系人的信息，再按他们的年龄从大到小的顺序依次输出其信息。试编写相应程序。

解答：

```

#include<stdio.h>
struct date{
    int y, m, d ;
};
struct friends_list{
    char name[10];
    struct date birthday;
    char phone[15];
};
int main(void)
{
    int i, n;
    struct friends_list friends[10];
    void sort(struct friends_list s[], int n);

    scanf("%d", &n);
    for(i=0; i<n; i++)
        scanf("%s%d%d%d%s", friends[i].name, &friends[i].birthday.y, &friends[i].birthday.m,
&friends[i].birthday.d, friends[i].phone);
    /*fill_b*/
    sort(friends, n);
    /*fill_n*/
    for(i=0; i<n; i++)
        printf("%s  %d  %s\n", friends[i].name, friends[i].birthday.y*10000+
friends[i].birthday.m*100+ friends[i].birthday.d, friends[i].phone);
    return 0;
}
void sort(struct friends_list s[], int n)
{
    int i, j;
    long b1, b2;
    struct friends_list temp;
    for(i=1; i<n; i++)
        for(j=0; j<n-i; j++){
            b1 = s[j].birthday.y*10000+ s[j].birthday.m*100+ s[j].birthday.d;

```



```

        b2 = s[j+1].birthday.y*10000+ s[j+1].birthday.m*100+ s[j+1].birthday.d;
        if( b1 > b2)
            {temp=s[j];  s[j]=s[j+1];  s[j+1]=temp; }
    }
}

```

6. 按等级统计学生成绩：输入 10 个学生的学号、姓名和成绩，输出学生的成绩等级和不及格人数。每个学生的记录包括学号、姓名、成绩和等级，要求定义和调用函数 `set_grade()`，根据学生成绩设置其等级，并统计不及格人数，等级设置：85~100 为 A，70~84 为 B，60~69 为 C，0~59 为 D。试编写相应程序。

解答：

```

/* 设置学生成绩等级并统计不及格人数（结构指针作为函数参数） */
#include<stdio.h>
#define N 10
struct student{
    int num;
    char name[20];
    int score;
    char grade;
};
int set_grade(struct student *p);
int main(void)
{
    struct student stu[N], *ptr;
    int i, count;
    ptr = stu;
    printf("Input the student's number, name and score: \n");
    for(i = 0; i < N; i++){
        printf("No %d: ", i+1);          /* 提示输入第 i 个同学的信息 */
        scanf("%d%s%d", &stu[i].num, stu[i].name, &stu[i].score);
    }
    count = set_grade(ptr);
    printf("The count (<60): %d\n", count);
    printf("The student grade:\n");
    for(i = 0; i < N; i++)
        printf("%d %s %c\n", stu[i].num, stu[i].name, stu[i].grade);
    return 0;
}
int set_grade(struct student *p)
{
    int i, n = 0;
    for(i = 0; i < N; i++, p++){
        if(p->score >= 85)
            p->grade = 'A';
        else if(p->score >= 70)

```

```

        p->grade = 'B';
    else if(p->score >= 60)
        p->grade = 'C';
    else{
        p->grade = 'D';
        n++;
    }
}
return n;
}

```

## 9.4 实验指导教材参考答案

### 一、调试示例

计算职工工资：输入一个正整数  $n$  ( $3 \leq n \leq 10$ )，再输入  $n$  个职员的信息（表 9.1），要求输出每位职员的姓名和实发工资（实发工资=基本工资+浮动工资-支出）。（源程序 error09\_1.c）

表 9.1 工资表

姓名	基本工资	浮动工资	支出
zhao	240.00	400.00	75.00
qian	360.00	120.00	50.00
zhou	560.00	150.00	80.00

错误行号： 10      正确语句： struct emp s[10];  
 错误行号： 16      正确语句： scanf("%s%f%f%f", s[i].name, &s[i].jbgz, &s[i].fdgz, &s[i].zc);

### 二、基础编程题

（1）时间换算：用结构类型表示时间内容（时间以时、分、秒表示），输入一个时间数值，再输入一个秒数  $n$  ( $n < 60$ )，以 h:m:s 的格式输出该时间再过  $n$  秒后的时间值(超过 24 点就从 0 点开始计时)。试编写相应程序。

解答：参见习题程序设计第 1 题

（2）计算平均成绩：建立一个学生的结构记录，包括学号、姓名和成绩。输入整数  $n$  ( $n < 10$ )，再输入  $n$  个学生的基本信息，要求计算并输出他们的平均成绩(保留 2 位小数)。试编写相应程序。

解答：

```

#include<stdio.h>
struct student{          /*学生信息结构定义*/
    int num;              /* 学号 */
    char name[10];        /* 姓名 */
    int score;             /* 成绩 */
}

```

```

};
int main(void)
{
    int i, n;
    double ave=0;
    struct student s1;    /* 定义结构变量 */
    printf("n=");
    scanf("%d", &n);
    for(i = 1; i <= n; i++){
        scanf("%d%s%d", &s1.num, s1.name, &s1.score);
        ave += s1.score;
    }
    ave = ave/n;
    printf("%.2lf\n", ave);
    return 0;
}

```

(3) 计算两个复数之积：利用结构变量求解两个复数之积，输入复数的实部与虚部都为整数。试编写相应程序。

解答：参见习题程序设计第 2 题

(4) 查找书籍：从键盘输入  $n(n < 10)$  本书的名称和定价并存入结构数组中，从中查找定价最高和最低的书籍的名称和定价，并输出。试编写相应程序。

解答：参见习题程序设计第 4 题

(5) 按等级统计学生成绩：输入 10 个学生的学号、姓名和成绩，输出学生的成绩等级和不及格人数。每个学生的记录包括学号、姓名、成绩和等级，要求定义和调用函数 `set_grade()`，根据学生成绩设置其等级，并统计不及格人数，等级设置：85~100 为 A，70~84 为 B，60~69 为 C，0~59 为 D。试编写相应程序。

解答：参见习题程序设计第 6 题

### 三、改错题

找出总分最高的学生：建立一个有  $n (3 < n \leq 10)$  个学生成绩的结构记录，包括学号、姓名和 3 门成绩，输出总分最高学生的姓名和总分。（源程序 `error09_2.cpp`）

错误行号： 8	正确语句： <u>}student[10];</u>
错误行号： 14	正确语句： <u>student[i].sum=0;</u>
错误行号： 24	正确语句： <u>max=student[i].sum;</u>

### 四、拓展编程题

(1) 通信录排序：通信录的结构记录包括：姓名、生日、电话号码，其中生日又包括三项：年、月、日。定义一个嵌套的结构类型，输入  $n(n < 10)$  个联系人的信息，再按他们的年龄从大到小的顺序依次输出其信息。试编写相应程序。

解答：参见习题程序设计第 5 题

(2) 有理数比较：编写函数 CompareRational，比较两个有理数的大小。该函数参数为两个有理数（结构类型）。若第一个有理数小于第二个，返回 -1；若相等，返回 0；若第一个有理数大于第二个，则返回 1。编写程序，接受用户输入的两对整数，分别组成两个有理数，并调用上述函数进行比较，输出比较结果。试编写相应程序。

解答：

```
#include<stdio.h>
struct rational{          /*有理数结构定义*/
    int num;
    int deno;
};
int CompareRational(struct rational r1, struct rational r2){
    double a,b;
    a=1.0*r1.num/r1.deno;
    b=1.0*r2.num/r2.deno;
    if(a>b) return 1;
    else if(a==b) return 0;
    else return -1;
}
int main(void)
{
    struct rational r1,r2;
    char ch;
    int n;
    scanf("%d/%d", &r1.num, &r1.deno);
    scanf("%d/%d", &r2.num, &r2.deno);
    n= CompareRational(r1, r2);
    if(n==1) ch='>';
    else if(n==0) ch='=';
    else ch='<';
    printf("%d/%d %c %d/%d\n", r1.num, r1.deno, ch, r2.num, r2.deno);
    return 0;
}
```

(3) 平面向量加法：输入两个二维平面向量  $V1=(x1, y1)$  和  $V2=(x2, y2)$  的分量，计算并输出两个向量的和向量。试编写相应程序。

解答：参见习题程序设计第 3 题