

《手把手教你学C语言》

图书基本信息

书名：《手把手教你学C语言》

13位ISBN编号：9787111553071

出版时间：2016-11

作者：吴明杰,曹宇,吴丹

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

书籍目录

前言

第1章 为什么要学习C语言

1

1.1 C的起源和发展

1

1.1.1 计算机语言发展的三个阶段

1

1.1.2 语言运行速度的比较

3

1.1.3 C语言的演变过程

3

1.2 C的特点

4

1.2.1 C语言的优点

4

1.2.2 C语言的缺点

5

1.3 C的应用领域

6

1.4 C的重要性

6

1.5 本章总结

7

第2章 怎样学习C语言

8

2.1 学习C语言的心得

8

2.2 学习C语言的目标

9

2.3 常见问题答疑

9

2.3.1 学习Java之前为什么建议先学C语言

9

2.3.2 没学过计算机专业课程能够学懂C语言吗

10

2.3.3 英语和数学不好能学好C语言吗

10

2.4 本章总结

11

第3章 Microsoft Visual C++ 6.0的使用

12

3.1 为什么要学习VC++ 6.0

12

3.2 如何创建编程文件

12

3.3 编写一个最简单的程序

16

3.4	要养成时刻保存的习惯	18
3.5	编译 – 链接 – 执行	18
3.6	怎样运行第二个程序	20
3.7	编译 – 链接 – 执行时保存路径下的文件夹有什么变化	21
3.8	如何编写多文件程序	23
3.9	如何用VC++ 6.0调试程序	28
3.10	本章总结	30
第4章	从一个程序走进C语言	31
第5章	预备知识	35
5.1	CPU、内存、硬盘、显卡、主板、显示器之间的关系	35
5.1.1	电影是如何运行的	35
5.1.2	CPU为什么不能直接操作硬盘却能直接操作内存	35
5.1.3	内存的速度为什么比硬盘的速度快	36
5.1.4	为什么不将内存造得跟硬盘一样大	36
5.1.5	CPU是如何操作内存的	36
5.1.6	主板的作用	37
5.2	HelloWorld程序是如何运行起来的	37
5.3	字节	38
5.3.1	什么是字节	38
5.3.2	字节换算	38
5.3.3	小结	38
5.4	进制	39
5.4.1	什么是进制	39
5.4.2	进制转换口算法	40
5.4.3	进制转换公式法	

41	
5.4.4	人类为什么最习惯用十进制
44	
5.4.5	计算机为什么用的是二进制
44	
5.4.6	小结
44	
5.5	数据类型
45	
5.5.1	数据类型的分类
45	
5.5.2	基本数据类型及其所占的字节数
46	
5.6	常量
48	
5.6.1	整型常量
49	
5.6.2	浮点型常量
49	
5.6.3	字符型常量
51	
5.7	常量是以什么样的二进制代码存储在计算机中的
51	
5.8	补码
52	
5.8.1	原码和反码
52	
5.8.2	补码的两个核心问题
53	
5.8.3	int型变量所能存储的范围
55	
5.8.4	int型和char型变量是如何相互赋值的
56	
5.9	什么是ASCII
57	
5.10	变量
58	
5.10.1	如何定义变量
58	
5.10.2	变量的本质
61	
5.10.3	为什么要使用变量
62	
5.10.4	变量的命名规则
62	
5.10.5	为什么必须要初始化变量
63	
5.10.6	小结
66	

5.11 各类型数据之间的混合运算

66

5.12 代码规范化

68

5.12.1 代码如何写才能规范

68

5.12.2 代码规范化的好处

68

5.12.3 代码规范化的七大原则

69

5.12.4 小结

72

5.13 本章总结

72

第6章 printf的用法

73

6.1 printf的格式

74

6.2 输出控制符

76

6.3 %x、%X、%#x、%#X的区别

76

6.4 如何输出“%d”、“\”和双引号

77

6.5 本章总结

78

第7章 scanf的用法

79

7.1 概述

79

7.2 使用scanf的注意事项

82

7.2.1 参数的个数一定要对应

82

7.2.2 输入的数据类型一定要与所需要的数据类型一致

83

7.2.3 在使用scanf之前使用printf提示输入

85

7.3 本章总结

86

第8章 运算符和表达式

87

8.1 算术运算符

87

8.2 关系运算符

89

8.3 逻辑运算符

90

8.4 赋值运算符

93	
8.5	运算符的优先级
94	
8.6	本章总结
95	
第9章	选择结构程序设计
96	
9.1	流程控制
96	
9.1.1	什么是流程控制
96	
9.1.2	流程控制的分类
96	
9.2	选择执行的定义和分类
97	
9.3	if语句
97	
9.3.1	if的最简单用法
97	
9.3.2	if的控制范围问题
99	
9.3.3	if...else的用法
100	
9.3.4	if...else if...else的用法
103	
9.3.5	练习—求分数的等级
104	
9.3.6	练习—三个整数从小到大排序
107	
9.3.7	if的常见问题解析
110	
9.4	switch语句
113	
9.5	本章总结
115	
第10章	循环控制
117	
10.1	循环执行的定义和分类
117	
10.2	for循环
117	
10.2.1	for语句的格式
117	
10.2.2	自增和自减
121	
10.2.3	for循环编程练习
122	
10.2.4	for和if的嵌套使用
122	

10.2.5	强制类型转换	125
10.2.6	浮点数的存储所带来的问题	130
10.2.7	for循环的嵌套使用	134
10.2.8	for循环嵌套编程练习	136
10.3	while循环	146
10.3.1	while的执行顺序	146
10.3.2	while和for的比较	147
10.3.3	如何看懂一个程序——“试数”	148
10.3.4	do...while	151
10.3.5	break和continue	156
10.4	清空输入缓冲区	158
10.4.1	输入缓冲区	158
10.4.2	%d和%c读取缓冲区的差别	159
10.4.3	用scanf吸收回车	161
10.4.4	getchar()	163
10.4.5	fflush(stdin)	166
10.5	本章总结	168
第11章 数组		
11.1	一维数组的使用	169
11.1.1	一维数组的定义	169
11.1.2	一维数组的初始化	170
11.1.3	一维数组元素的引用	173
11.1.4	如何将数组a赋给数组b	174
11.1.5	如何编程获取数组的长度	175
11.1.6	练习	

177	
11.1.7	宏定义：#define
179	
11.1.8	小结
182	
11.2	数组倒置算法
182	
11.3	数组查找算法
184	
11.3.1	顺序查找
184	
11.3.2	折半查找
185	
11.4	数组插入、删除算法
188	
11.4.1	插入算法
188	
11.4.2	删除算法
189	
11.5	数组排序算法
190	
11.5.1	冒泡排序
190	
11.5.2	插入排序
193	
11.5.3	选择排序
196	
11.5.4	快速排序
198	
11.5.5	四种排序算法的比较
204	
11.6	二维数组的使用
205	
11.6.1	二维数组的定义
205	
11.6.2	二维数组的初始化
206	
11.6.3	二维数组如何输出
207	
11.6.4	练习—杨辉三角
208	
11.6.5	是否存在多维数组
213	
11.7	本章总结
213	
第12章	函数
215	
12.1	概述
215	

- 12.1.1 什么是函数
215
- 12.1.2 C程序的组成和编译单位
216
- 12.1.3 库函数和自定义函数
216
- 12.1.4 函数的调用
216
- 12.2 为什么需要函数
217
- 12.3 有参函数
217
 - 12.3.1 有参函数定义的一般形式
218
 - 12.3.2 形参和实参
219
 - 12.3.3 主调函数中对被调函数的声明
220
 - 12.3.4 定义函数时应指定返回值类型
221
 - 12.3.5 函数的返回值
222
 - 12.3.6 return是如何将值返回给主调函数的
222
 - 12.3.7 函数的命名规则
223
 - 12.3.8 练习—判断一个自然数是否是素数
223
- 12.4 函数的递归调用
227
 - 12.4.1 什么是递归
227
 - 12.4.2 使用递归必须要满足的两个条件
228
 - 12.4.3 如何学习递归
229
 - 12.4.4 递归和循环的关系
229
 - 12.4.5 递归的优缺点
229
 - 12.4.6 练习—用递归求n的阶乘
230
 - 12.4.7 练习—用递归实现1+2+3+...+100的和
232
 - 12.4.8 练习—用递归求斐波那契数列
233
 - 12.4.9 练习—用递归求两个数的最大公约数
234
 - 12.4.10 小结

235	
12.5	数组名作为函数参数
236	
12.6	变量的作用域和存储方式
238	
12.6.1	局部变量
239	
12.6.2	全局变量
240	
12.6.3	为什么不建议使用全局变量
242	
12.6.4	自动变量 (auto)
242	
12.6.5	静态变量 (static)
242	
12.6.6	寄存器变量 (register)
245	
12.6.7	外部变量 (extern)
246	
12.7	本章总结
248	
	第13章 指针
249	
13.1	指针的重要性
249	
13.2	地址和指针的概念
250	
13.3	指针和指针变量
251	
13.3.1	指针变量的定义
252	
13.3.2	指针变量的初始化
254	
13.3.3	指针常见错误
257	
13.4	指针作为函数参数
260	
13.4.1	互换两个数
260	
13.4.2	函数参数传指针和传数据的区别
262	
13.4.3	定义只读变量 : const
263	
13.4.4	用const修饰指针变量时的三种效果
264	
13.5	指针和一维数组的关系
266	
13.5.1	用指针引用数组元素 266
13.5.2	指针的移动

267	
13.5.3	指针变量的自增运算
271	
13.5.4	两个参数确定一个数组
277	
13.5.5	指针变量占多少字节
279	
13.6	函数、数组、指针相结合的程序练习
280	
13.7	动态内存分配
282	
13.7.1	传统数组的缺点
282	
13.7.2	malloc函数的使用（一）
283	
13.7.3	malloc函数的使用（二）
286	
13.7.4	free函数的使用
289	
13.7.5	练习—动态数组的构建
291	
13.7.6	动态数组长度的扩充和缩小
294	
13.7.7	静态内存分配和动态内存分配小结
296	
13.7.8	多级指针
296	
13.7.9	跨函数使用动态内存
298	
13.8	通过指针引用二维数组
302	
13.8.1	二维数组元素的地址
302	
13.8.2	二维数组的首地址和数组名
303	
13.8.3	练习
307	
13.9	函数指针
309	
13.9.1	什么是函数指针
309	
13.9.2	如何用函数指针调用函数
310	
13.10	本章总结
311	
第14章	字符串
313	
14.1	字符串常量
313	

14.2	不能将一个字符串常量赋给一个字符变量	314
14.3	字符数组	315
14.3.1	字符数组的定义	315
14.3.2	字符数组的初始化	315
14.3.3	初始化内存函数：memset()	318
14.3.4	用scanf输入字符串	320
14.4	字符串与指针	322
14.5	如何用scanf给字符指针变量所指向的内存单元初始化	323
14.6	字符串处理函数	324
14.6.1	字符串输入函数gets()	324
14.6.2	优先使用fgets()函数	326
14.6.3	使用gets()和fgets()前注意吸收回车	329
14.6.4	练习——判断字符串是否是回文字符串	331
14.6.5	练习——将字符串中的小写字母转换成大写字母	336
14.6.6	字符串输出函数puts()	337
14.6.7	字符串输出函数fputs()	339
14.6.8	字符串复制函数strcpy()	341
14.6.9	字符串复制函数strncpy()	343
14.6.10	内存拷贝函数memcpy()	343
14.6.11	sprintf()	345
14.6.12	字符串比较函数strcmp()	347
14.7	本章总结	348
第15章	自定义数据类型——结构体	350
15.1	为什么需要结构体	350
15.2	定义和使用结构体变量	

351	
15.2.1	声明结构体类型
351	
15.2.2	定义结构体类型变量
352	
15.2.3	结构体变量可进行哪些运算
354	
15.2.4	结构体变量的初始化—定义时初始化
354	
15.2.5	结构体变量的引用
355	
15.2.6	结构体变量的初始化—先定义后初始化
358	
15.2.7	结构体字节对齐
363	
15.3	结构体数组
369	
15.3.1	结构体数组的定义和引用
369	
15.3.2	结构体数组定义时初始化
371	
15.4	结构体指针
372	
15.4.1	指向结构体变量的指针
372	
15.4.2	指向结构体数组的指针
375	
15.5	结构体变量和结构体指针变量作为函数参数
378	
15.6	练习—动态构造存放学生信息的结构体数组
382	
15.7	本章总结
385	
第16章	链表
387	
16.1	为什么要学习链表
387	
16.1.1	数组的缺陷
388	
16.1.2	链表的优点
388	
16.2	链表的缺点 389
16.3	链表相关术语
389	
16.4	链表的定义和分类
390	
16.4.1	链表的定义
390	
16.4.2	链表的分类

390	
16.5	编写一个链表程序
391	
16.6	练习——插入结点
396	
16.7	练习——删除结点
404	
16.8	练习——销毁链表
409	
16.9	链表排序算法
413	
16.9.1	如何互换结点
413	
16.9.2	链表冒泡排序
414	
16.9.3	链表插入排序
418	
16.9.4	链表选择排序
421	
16.9.5	链表快速排序
422	
16.9.6	链表排序小结
425	
16.10	单循环链表
425	
16.11	双向链表
430	
16.12	typedef的用法
434	
16.13	本章总结
437	
	第17章 栈
439	
17.1	内存的分区
439	
17.2	线性表
440	
17.3	栈的定义
441	
17.4	栈的基本操作
441	
17.5	链栈程序演示
442	
17.6	栈的应用
450	
17.7	本章总结
454	
	第18章 队列
455	

18.1	队列的定义	455
18.2	顺序队列	455
18.3	链式队列的基本操作	456
18.4	链式队列程序演示	457
18.5	队列的应用	464
18.6	本章总结	464
第19章	文件操作	465
19.1	文件概述	466
19.1.1	文件的定义	466
19.1.2	文件的分类	466
19.2	文件类型指针变量	467
19.3	文件的打开	469
19.3.1	打开文件	469
19.3.2	文件的使用方式	470
19.3.3	练习——打开文件	472
19.4	文件的关闭	474
19.5	文件读写函数概述	475
19.6	fgetc()和fputc()	475
19.6.1	fputc()	475
19.6.2	fgetc()	477
19.6.3	fgetc()程序举例	478
19.6.4	fgetc()和fputc()结合程序举例	482
19.6.5	如何判断文件是否为空	483
19.7	fgets()和fputs()	484
19.7.1	fgets()	

484	
19.7.2	fputs()
486	
19.8	移动文件位置指针：fseek()和rewind()
488	
19.8.1	fseek()
488	
19.8.2	rewind()
489	
19.9	格式化读写函数：fprintf()和fscanf()
489	
19.10	数据块读写函数：fread()和fwrite()
491	
19.10.1	fwrite()函数
492	
19.10.2	fread()函数
494	
19.10.3	ftell()
500	
19.10.4	如何用fread()和fwrite()删除文件中的数据块
503	
19.10.5	练习——登录程序
509	
19.11	本章总结
513	
第20章	头文件
515	
20.1	程序是如何编译生成可执行文件的
515	
20.2	概述
516	
20.2.1	什么是头文件
516	
20.2.2	头文件的作用
517	
20.2.3	什么样的内容适合放在头文件中
517	
20.3	“#include”和“#include”的区别
518	
20.4	如何自定义头文件
519	
20.4.1	头文件的组成部分
519	
20.4.2	#ifndef/#define/#endif
520	
20.5	本章总结
522	
第21章	位操作运算符
523	

21.1	按位与 (&)	523
21.2	按位或 ()	524
21.3	按位异或 (^)	524
21.4	左移运算符 (<<)	527
21.6	本章总结	528
附录A	gcc编译工具的使用	530
附录B	make自动化编译工具的使用	537
附录C	gdb调试工具的使用	549
附录D	链接库	563
附录E	运算符的优先级和结合性	572
	常用ASCII码表	574

《手把手教你学C语言》

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu000.com