



中等职业教育国家规划教材配套教材  
全国中等职业教育教材审定委员会审定

# C语言编程基础

## 上机指导与练习

展 涛 主 编

<http://www.phei.com.cn>

计算机  
及应用专业



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

本书配有电子教学参考资料包

TP312  
847

中等职业教育国家规划教材配套教材

# C 语言编程基础上机 指导与练习

展 涛 主 编

庞春 限 公 司

電子工業出版社

Publishing House of Electronics Industry

北京 · BEIJING

## 内 容 简 介

本书是中等职业教育国家规划教材《C 语言编程基础》的配套上机指导与练习用书。

本书的主要内容有 C 语言的基本数据类型、表达式和运算符、数据的输入和输出语句、控制语句、数组、函数、文件、编译预处理和指针,以及简单程序的设计方法,每部分内容分为上机指导和习题两个模块。书中针对目前职业学校的教学特点,习题难易适中,实用性强,便于教师和学生使用。

本书是中等职业学校计算机及应用专业的学生学习 C 语言的配套教材,也可作为计算机爱好者自学 C 语言的参考用书。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

## 图书在版编目(CIP)数据

C 语言编程基础上机指导与练习/展涛主编. —北京:电子工业出版社,2004.7

中等职业教育国家规划教材配套教材

ISBN 7-5053-9967-5

I. C… II. 展… III. C 语言—程序设计—专业学校—教学参考资料 IV. TP312

中国版本图书馆 CIP 数据核字(2004)第 075796 号

责任编辑:陈健德

特约编辑:师彦武

印 刷:涿州京南印刷厂

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:9.5 字数:243.2 千字

印 次:2004 年 7 月第 1 次印刷

印 数:5000 册

定价:12.00 元

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系。联系电话:(010) 68279077。质量投诉请发邮件至 [zltz@phei.com.cn](mailto:zltz@phei.com.cn),盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

# 前言



本书是中等职业教育国家规划教材《C语言编程基础》的配套上机指导与练习用书,适合中等职业学校计算机及应用专业的学生学习使用。

本书在内容编排顺序上与原教材保持一致,每一章节分为两个部分:一部分是上机指导,另一部分是习题。教材的编写力求适应目前职业学校的教学情况,并体现出以下特点:

(1) 针对原教材中的难点、重点做相应的补充和强调,将其归纳总结,实现系统化,有利于学生对知识的深入掌握。

(2) 上机指导部分在上机题目的设计上力求短小精悍,对学生容易出差错的地方都有对应的练习,增加了一些比较有趣味性的小程序,让学生在实习时有一种解决现实问题的成就感。

(3) 习题的设计紧扣大纲,题型比较灵活,所选习题涵盖了原教材中所有的知识点,既可作为随堂练习,也可作为学生的家庭作业,避免了授课教师东拼西凑找习题之苦,具有很高的实用价值。

(4) 习题难易程度适中,没有偏题和怪题,重在学生基础知识的掌握,并注意帮助学生举一反三,培养应用能力。

作为教材,为了方便老师教学使用,本书还配有教学指南、电子教案、程序设计例题的源代码及习题答案(电子版),请有此需要的教师与电子工业出版社联系,我们将免费提供。电子邮件地址为: [ve@phei.com.cn](mailto:ve@phei.com.cn)。

本书由展涛主编,庞春蕾、王辉参加了本书的编写工作,全书由展涛统稿。

由于作者水平有限,经验不足,书中难免存在不少缺点和错误,敬请读者批评指正。

编者

2004年7月



# 目 录



|                   |      |
|-------------------|------|
| 第1章 概 述           | (1)  |
| 知识要点              | (1)  |
| 上机指导              | (1)  |
| 典型例题              | (3)  |
| 习题1               | (3)  |
| 第2章 基本数据类型和简单程序设计 | (6)  |
| 知识要点              | (6)  |
| 上机指导              | (7)  |
| 典型例题              | (8)  |
| 习题2               | (9)  |
| 第3章 运算符与表达式       | (14) |
| 知识要点              | (14) |
| 上机指导              | (14) |
| 典型例题              | (17) |
| 习题3               | (17) |
| 第4章 数据的输入与输出      | (27) |
| 知识要点              | (27) |
| 上机指导              | (27) |
| 趣味小程序             | (28) |
| 典型例题              | (29) |
| 习题4               | (31) |
| 第5章 控制语句          | (36) |
| 5.1 条件选择语句 if     | (36) |
| 知识要点              | (36) |
| 上机指导              | (36) |
| 典型例题              | (37) |
| 习题5               | (38) |
| 5.2 开关语句 switch   | (43) |
| 知识要点              | (43) |
| 上机指导              | (43) |
| 典型例题              | (44) |
| 习题6               | (45) |
| 5.3 for 循环语句      | (47) |

|                                   |       |
|-----------------------------------|-------|
| 知识要点 .....                        | (47)  |
| 上机指导 .....                        | (48)  |
| 趣味小程序 .....                       | (49)  |
| 典型例题 .....                        | (50)  |
| 习题 7 .....                        | (50)  |
| 5.4 while 语句和 do-while 循环语句 ..... | (55)  |
| 知识要点 .....                        | (55)  |
| 上机指导 .....                        | (55)  |
| 典型例题 .....                        | (56)  |
| 习题 8 .....                        | (57)  |
| <b>第 6 章 数 组</b> .....            | (63)  |
| 6.1 数值型数组 .....                   | (63)  |
| 知识要点 .....                        | (63)  |
| 上机指导 .....                        | (64)  |
| 典型例题 .....                        | (65)  |
| 习题 9 .....                        | (66)  |
| 6.2 字符型数组 .....                   | (73)  |
| 知识要点 .....                        | (73)  |
| 上机指导 .....                        | (73)  |
| 典型例题 .....                        | (74)  |
| 习题 10 .....                       | (75)  |
| <b>第 7 章 函 数</b> .....            | (80)  |
| 知识要点 .....                        | (80)  |
| 上机指导 .....                        | (81)  |
| 典型例题 .....                        | (82)  |
| 习题 11 .....                       | (83)  |
| <b>第 8 章 文 件</b> .....            | (91)  |
| 知识要点 .....                        | (91)  |
| 习题 12 .....                       | (92)  |
| <b>第 9 章 编译预处理</b> .....          | (95)  |
| 知识要点 .....                        | (95)  |
| 上机指导 .....                        | (96)  |
| 典型例题 .....                        | (96)  |
| 习题 13 .....                       | (97)  |
| <b>第 10 章 指 针</b> .....           | (99)  |
| 知识要点 .....                        | (99)  |
| 上机指导 .....                        | (101) |
| 典型例题 .....                        | (103) |
| 习题 14 .....                       | (104) |

模拟试题 1 ..... (113)

模拟试题 2 ..... (118)

模拟试题 3 ..... (123)

模拟试题 4 ..... (128)

模拟试题 5 ..... (133)

模拟试题 6 ..... (138)

# 第1章 概述



## 知识要点

- C 语言的标志符和保留字。
- C 语言程序的结构特点和书写格式。
- C 语言程序的编辑及运行方法。



## 上机指导

### 1. 实验目的

- (1) 熟悉 Turbo C 2.0 的软件环境，掌握使用 Turbo C 进行编辑、编译、连接和运行的方法。
- (2) 通过运行简单的 C 程序，掌握 C 语言源程序的特点。

### 2. 实验内容

- (1) 安装并运行 Turbo C 2.0 程序，进入 C 语言程序设计环境。
- (2) 编辑源程序文件。

选择“File”→“Load”菜单命令，加载已经存在的 C 语言源程序文件。

选择“File”→“New”菜单命令建立新文件。

请输入下面的 C 语言源程序，在编辑完成后，用“File”→“Save”菜单命令或 F2 键保存编辑结果到新文件中。

```
int a=4,b=2;
main()
{
    int e=1,f=2;
    printf("%d\n",max(a,b));
    printf("%d\n",max(e,f));
}
int max(int x,int y)
{
    return x>y?x:y;
}
```



### (3) 编译和连接程序文件。

源程序先经过编译,生成目标文件(.OBJ 格式文件),然后将目标文件进行连接,生成可执行文件(.EXE 格式文件)。

步骤一:

选择“Compile”→“Compile to OBJ”菜单命令,可将源程序编译成目标文件;再选择“Compile”→“Link EXE file”菜单命令,可将目标文件及库文件连接在一起,生成可执行文件。

步骤二:

选择“Compile”→“Make EXE file”菜单命令,可将源程序直接经过编译、连接后生成可执行文件。

如果在编译或连接过程中发生错误,在“Message”窗口中将显示出错信息,修改错误后,可重新对源程序进行编译、连接,直到程序完全正确。

### (4) 运行编译生成的可执行程序文件。

如果程序的编译和连接过程完全正确,则可以选择“Run”菜单命令运行编译生成的可执行程序文件。通过“Run”→“Run”命令可完成编译、连接和运行任务。如果可以确认源程序无错误,可以直接使用该命令。在程序运行完毕后,仍返回到 Turbo C 环境,选择“Run”→“User screen”菜单命令可以查看程序的输出结果,在看完后按任意键返回到 Turbo C 环境。

### (5) 下面是一个最简单的 C 语言程序。

```
main()
{ }
```

这是一个只有主框架的源程序,虽然它没有可执行的代码,但是却可以被很好地编译和执行。

### (6) #include<stdio.h>

```
main()
{
    printf("祝大家学习进步! \n");
}
```

这是最简单的程序之一,每一种编程语言都会提供一个这样的例子,你可以将程序中的斜体字用其他的语句来代替。

### (7) 编辑并运行下面的 C 语言源程序,查看程序的运行结果。

```
main()
{
    int num1, num2;                                /* 变量定义部分 */
    printf("Input the first integer number: ");
    scanf("%d", &num1);
    printf("Input the second integer number: ");    /* 可执行语句部分 */
    scanf("%d", &num2);
    printf("max=%d\n", max(num1, num2));
}
```



```
int max(int x,int y)
```

```
{
```

```
    return x>y?x:y;
```

```
}
```

### 3. 分析与讨论

下面总结 C 程序的结构和书写规则。



### 典型例题

例 1 以下不能定义为用户标志符的是\_\_\_\_\_。

A. scanf

B. void

C. \_3com\_

D. int

**解析:** C 语言的用户标志符由字母或下划线开头, 由字母、数字或下划线组成, 从这个角度看, 本题的 4 个选项都符合。但 D 选项中的 int 属于 C 语言的保留字, 不能再作为用户标志符使用; A 选项中的 scanf 虽然是 C 语言中标准输入函数的名称, 但不属于保留字, 故仍可作为用户标志符使用。



### 习题 1

1. 选择题 (在每小题列出的 4 个选项中, 只有 1 个是正确的)

(1) C 程序的基本单位是\_\_\_\_\_。

A. 标志符

B. 函数

C. 表达式

D. 语句

(2) 一个 C 语言程序总是从\_\_\_\_\_开始执行。

A. 书写顺序的第 1 个函数

B. 书写顺序的第 1 条执行语句

C. 主函数 main()

D. 不确定

(3) C 语言中, 所有的保留字必须\_\_\_\_\_。

A. 大写

B. 小写

C. 大小写均可

D. C 语言中没有保留字

(4) C 语言采用\_\_\_\_\_方式将源程序翻译为二进制代码。

A. 编译

B. 解释

C. 汇编

D. 翻译

(5) 合法的用户定义标志符是\_\_\_\_\_。

A. P-S

B. putchar

C. double

D. 908

(6) 结构化程序应尽量避免使用的语句是\_\_\_\_\_。

A. do-while 语句

B. for 语句

C. goto 语句

D. while 语句

(7) C 语言的保留字是\_\_\_\_\_。

A. char

B. abc

C. printf

D. Int



(8) C 语言的主要特点不包括下面的\_\_\_\_\_。

- A. C 语言生成的目标代码质量低      B. C 语言是结构化程序设计语言  
C. C 语言允许直接对位、字节和地址进行操作      D. 数据类型丰富

(9) 编程风格叙述不正确的是\_\_\_\_\_。

- A. 大小写字母用在不同的场合,一般除了符号名和常量名用大写字母外,其他一律用小写字母  
B. 使用有意义的标志符      C. 程序中的注释可有可无      D. 使用括号来改善表达式的清晰度

(10) C 程序描述不正确的是\_\_\_\_\_。

- A. 每个语句和数据定义的最后必须为分号  
B. 一个 C 程序的书写格式要求严格,一行只能写一个语句  
C. C 语言本身没有输入、输出语句  
D. 一个 C 程序总是从 main() 函数开始执行

(11) C 语言结构形式的正确描述是\_\_\_\_\_。

- A. main() 函数必须放在程序之首      B. 每一行只能写一个语句,每个语句之后必须为分号  
C. C 语言程序中可以加注释      D. C 语言的自由度大,可以从任何函数执行

(12) 在每一个 C 程序中都必须包含一个名为\_\_\_\_\_的函数。

- A. main      B. MAIN      C. name      D. function

(13) 关于 C 语言用户标志符的叙述正确的是\_\_\_\_\_。

- A. 用户标志符中可以出现下划线和中划线(减号)  
B. 用户标志符中不可以出现中划线,但可以出现下划线  
C. 用户标志符中可以出现下划线,但不可以放在用户标志符的开头  
D. 用户标志符中可以出现下划线和数字,它们都可以放在用户标志符的开头

(14) 以下叙述正确的是\_\_\_\_\_。

- A. C 程序中注释部分可以出现在程序中任意合适的地方  
B. 花括号“{”和“}”只能作为函数体的定界符  
C. 构成 C 程序的基本单位是函数,所有函数名都可以由用户命名  
D. 分号是 C 语句之间的分隔符,不是语句的一部分

## 2. 填空题

(1) C 语言是结构化的程序设计语言,程序结构一般采用\_\_\_\_\_和\_\_\_\_\_结构。

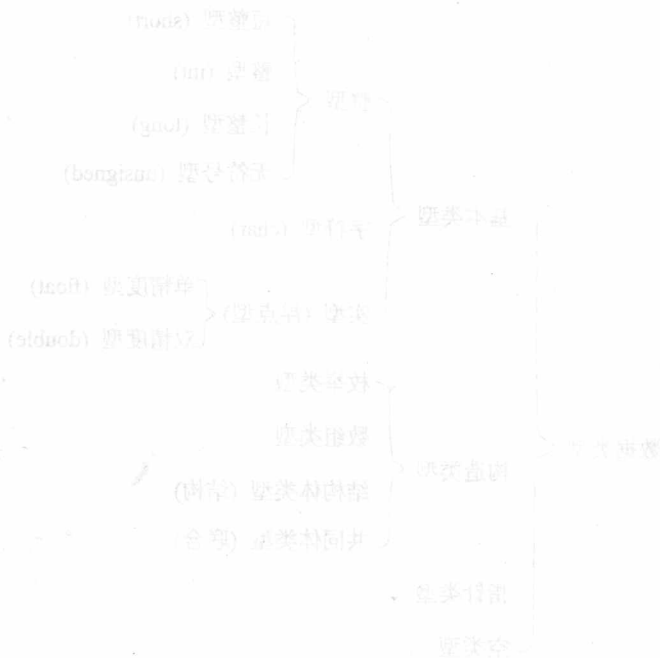
(2) C 语言程序由函数组成,函数由函数的\_\_\_\_\_和\_\_\_\_\_两部分组成。

(3) 在 C 语言中,一条语句可以写在多行上,用\_\_\_\_\_作续行符。

(4) 在 C 语言中,标志符主要作为\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_的名字使用。



- (5) 为了增强程序的可读性,可以利用\_\_\_\_\_对 C 语言程序中的语句进行注释。
- (6) 最初,设计 C 语言的目的只是为了描述和实现\_\_\_\_\_操作系统。1983 年美国国家标准化协会 (ANSI) 对 C 语言问世以来的各种版本进行了扩充和规范,制定了 ANSI C。
- (7) C 语言的基本符号集采用\_\_\_\_\_码符号集。
- (8) 函数体由符号\_\_\_\_\_开始,用符号\_\_\_\_\_结束,函数体可分为\_\_\_\_\_和\_\_\_\_\_两部分。
- (9) C 语言源程序文件的后缀是\_\_\_\_\_;在经过编译后,生成文件的后缀是\_\_\_\_\_;在经过连接后,生成文件的后缀是\_\_\_\_\_。
- (10) C 语言程序的语句以\_\_\_\_\_结束。
- (11) 为了生成可执行文件,\_\_\_\_\_程序把编译器的输出和各种库函数连接在一起。
- (12) C 语言的程序模块称为\_\_\_\_\_。

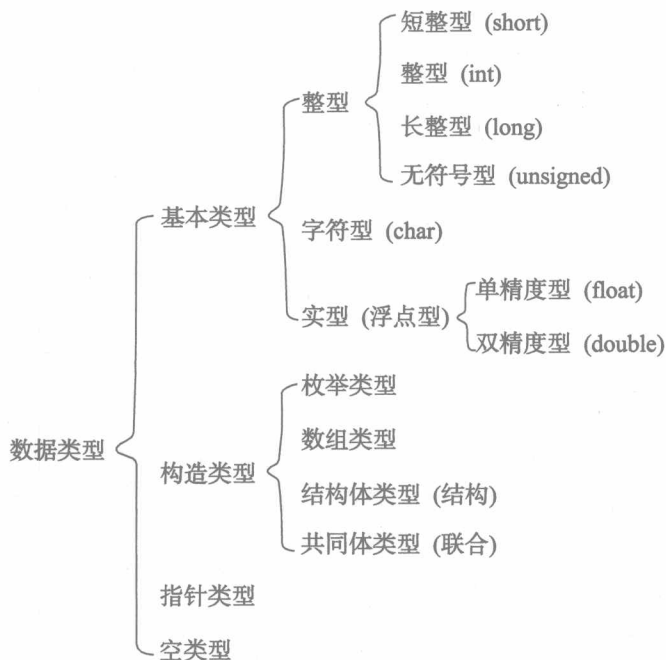


# 第2章 基本数据类型和简单程序设计



## 知识要点

### 1. C 语言的基本数据类型



### 2. 不同类型数据的混合运算

在 C 语言中，整型、实型和字符型数据之间可以进行混合运算（因为字符数据与整型数据可以通用）。

如果一个运算符两侧的操作数的数据类型不同，则系统按“先转换、后运算”的原则，首先将数据自动转换成同一类型，然后在同一类型数据之间进行运算。转换规则为：

```
double ← float
↑
long
↑
int ← char, short
```



(1) 横向向左的箭头, 表示必须进行的转换。char 和 short 型必须转换成 int 型, float 型必须转换成 double 型。

(2) 纵向向上的箭头, 表示不同类型的转换方向。在 int 型与 double 型数据进行混合运算时, 要先将 int 型数据转换成 double 型数据, 然后在两个同类型的数据之间进行运算, 结果为 double 型。

(3) 除自动转换外, C 语言也允许强制转换。

数据类型强制转换的一般格式为: (要转换成的数据类型)(被转换的表达式)。

当被转换的表达式是一个简单表达式时, 外面的一对圆括号可以省略。

例如“(double)a”等价于“(double)(a)”, 表示将变量 a 的值转换成 double 型;“(int)(x+y)”表示将 x+y 的结果转换成 int 型。



## 上机指导

### 1. 实验目的

- (1) 了解 C 语言中数据类型的意义。
- (2) 了解 C 语言中数据类型的转换方式。

### 2. 实验内容

- (1) 输入下面程序, 观察它的运行结果。

```
main()
{
    char ch1, ch2;
    ch1 = 'a'; ch2 = 'B';
    printf("ch1 = %c, ch2 = %c\n", ch1 - 32, ch2 + 32); /* 字母的大小写转换 */
    printf("ch1 + 200 = %d\n", ch1 + 200); /* 用字符形式输出一个大于 256 的数值 */
    printf("ch1 + 200 = %c\n", ch1 + 200);
    printf("ch1 + 256 = %d\n", ch1 + 256);
    printf("ch1 + 256 = %c\n", ch1 + 256);
}
```

C 语言允许对字符数据进行算术运算, 此时就是对它们的 ASCII 码值进行算术运算。



### 想一想

用字符形式输出一个大于 256 的数值, 会得到什么结果?

- (2) 下面程序的目的是计算由键盘输入的任意两个整数的平均值。

```
#include <stdio.h>
main()
{
    int x, y, a;
```



```
scanf("%d,%d",&x,&y);
a=(x+y)/2;
printf("The average is:%d", a);
}
```

在调试无语法错误后,分别使用下列测试用例对上述程序进行测试:

- 2, 6
- -2, -6
- 1, 0
- 32800, 33000
- -32800, 33000

① 分析上述哪几组测试用例较好。通过测试,你发现程序有什么问题吗?若有问题,请指出原因及改正方法。

② 运算符 `sizeof` 用以测试一个数据或类型所占用的存储空间的字节数。请编写一个程序,测试各基本数据类型所占用的存储空间大小。

(3) 输入并运行下面的程序。

```
main()
{
    char c1='a',c2='b',c3='c',c4='\101',c5='\116';
    printf("a%c b%c\tc%c\tabc\n",c1,c2,c3);
    printf("\t\tb%c%c",c4,c5);
}
```

要求在运行之前先分析程序,写出运行结果,再与上机结果验证。

### 3. 分析与讨论

- (1) 如何正确地选用数据类型?(提示:结合前面做过的实验及课本进行讨论总结)
- (2) 将一个负整数赋给一个无符号的变量,会得到什么结果?
- (3) 将一个大于 32767 的长整数赋值给整型变量,会得到什么结果?



### 典型例题

例 1 指出下面的常量、变量哪些是非法的,哪些是合法的。如果是常量,请说明数据类型。

- (1) 45 (2) 2+3 (3) x (4) 2e3 (5) E3 (6) 4E2.5 (7) "{X" (8) 'ABC'  
(9) 0xabcl (10) 0e0 (11) 0675 (12) "3>4" (13) 4.8e (14) T (15) '\*'

解析: (1) 45 为整型常量; (2) 2+3 的结果为整型的算术表达式; (3) x 是一个变量; (4) 2e3 是一个实型常量; (5) E3 是一个变量; (6) 4E2.5 是一个非法常量,因为浮点数的指数形式的指数部分只能为整型量; (7) "{X" 是字符串常量; (8) 'ABC' 是非法常量,因为单引号只能括单字符; (9) 0xabcl 是一个十六进制长整型常量; (10) 0e0 是浮点型常量;



- (11) 0675 是一个八进制整型常量; (12) "3>4" 是一个字符串常量; (13) 4.8e 是非法常量;  
(14) T 是一个变量; (15) '\*' 是一个字符常量。

例2 写出下面程序的运行结果。

```
main()
{
    float f=5.75;
    printf("(int)f=%d,f=%f\n",(int)f,f);
}
```

解析: f 虽强制转为 int 型, 但只在运算中起作用, 而 f 本身的类型并不改变。因此, (int)f 的值为 5 (删去了小数), 而 f 的值仍为 5.75。即强制转换类型得到的是一个所需类型的中间量, 原表达式类型并不发生变化。例如, (double)a 只是将变量 a 的值转换成一个 double 型的中间量, 其数据类型并未转换成 double 型。

输出结果为: (int)f=5, f=5.750000。



## 习题2

1. 选择题 (在每小题列出的4个选项中, 只有1个是正确的)

(1) 若有定义 int a, 变量 a 占用内存的 2 个字节, 则 a 所能存储数据的最小值为\_\_\_\_\_。

A. 0

B. -32767

C. 32768

D. -65535

(2) C 语言中转义字符非法的是\_\_\_\_\_。

A. '\t'

B. 'abc'

C. '\0xaf'

D. '\r'

(3) 变量定义合法的是\_\_\_\_\_。

A. short aL=5678;

B. double b=1+5e2.5;

C. long do=0xfdaL;

D. float 2\_and=1e-3;

(4) 语句 "printf("a\\bre\\'hi\\'y\\\\"bou\\n");" 的输出结果是\_\_\_\_\_。

A. a\\bre\\'hi\\'y\\\\"bou

B. a\\bre\\'hi\\'y\\bou

C. re'hi'you

D. abre'hi'y\\bou

(5) 若有说明语句 "char a='\\285';" 则变量 a \_\_\_\_\_。

A. 包含 1 个字符

B. 包含 2 个字符

C. 包含 3 个字符

D. 说明不合法

(6) 数值最大的是\_\_\_\_\_。

A. 74

B. 074

C. '\\101'

D. '\\x43'

(7) 下列符号串与 45.67 不相同的常量是\_\_\_\_\_。

A. 4.567E1

B. "45.67"

C. .4567E2

D. 4567E-2

(8) 以下叙述正确的是\_\_\_\_\_。



- A. a 是实型变量, 因此 C 语言不允许进行赋值  $a=10$ ;
- B. 在赋值表达式中, 赋值号右边可以是变量, 也可以是常量或表达式
- C. 在赋值表达式中, 赋值号左边可以是变量, 也可以是常量或表达式
- D. 已有  $a=3$ , 当执行了表达式  $a+6$  之后, 使 a 中的值为 9

(9) 下面这段程序中的\_\_\_\_\_属于常量。

```
main()
{
    int sum;
    sum = 10 + sum;
    printf("sum = %d\n", sum);
    printf("result is 25\n");
}
```

- A. 10                      B. sum                      C. main                      D. printf
- (10) 下面实现换行功能的格式符为\_\_\_\_\_。
- A. '\t'                      B. '\n'                      C. '\a'                      D. '\k'

(11) 下列不正确的转义字符是\_\_\_\_\_。

- A. '\\ '                      B. '\ '                      C. '074'                      D. '\0 '
- (12) 下列叙述正确的是\_\_\_\_\_。

- A. C 语言中既有逻辑类型也有集合类型                      B. C 语言中没有逻辑类型但有集合类型
- C. C 语言中有逻辑类型但没有集合类型                      D. C 语言中既没有逻辑类型也没有集合类型
- (13) 若有以下程序段:

```
int m=32767, n=032767;
printf("%d, %o\n", m, n);
```

执行后输出结果是\_\_\_\_\_。

- A. 32767,32767                      B. 32767,032767                      C. 32767,77777                      D. 32767,077777
- (14) 有以下程序段:

```
main()
{
    char a,b,c,*d;
    a='\'; b='\xabc';
    c='\0xab'; d="\0127";
    printf("%c%c%c%c\n",a,b,c,*d);
}
```

在编译时出现错误, 以下叙述正确的是\_\_\_\_\_。

- A. 程序中只有“ $a='\'$ ”语句不正确                      B. “ $b='\xabc'$ ”语句不正确
- C. “ $d="\0127"$ ”语句不正确                      D. “ $a='\'$ ”和“ $c='\0xab'$ ”语句都不正确