非常好的C语言章节习题集带答案 - 百度文库

百度文库

搜索文档或关键词

普通分享 >

[习题集](//wk.baidu.com/search?word=习题集&fr=viewTags)

非常好的C语言章节习题集带答案.

VIP专享文档

2019-04-10

106页

用App查看

## 第 1 章 认识 C 语言

二、习题

（一）、是非题

1. 程序是指挥计算机进行各种信息处理任务的一组指令序列。

A. 对 B. 错

2. 机器语言与硬件平台相关，但汇编语言和硬件平台无关。

A. 对 B. 错

3. 编译型高级语言明显优于解释型高级语言。

A. 对 B. 错

4. C 语言把高级语言的基本结构和低级语言的实用性紧密结合起来，不仅适合编写应用软件，而且适于编写系统软件。

A. 对 B. 错

5. 面向对象的程序设计方法明显优于面向过程的程序设计方法。

A. 对 B. 错

6. 计算机算法要有一个明确的起点和确定的步骤序列。

A. 对 B. 错

7. main 函数是 C 程序的入口，由计算机系统负责调用。

A. 对 B. 错

8. 在 C 语言中，同一行上可以写一条或多条语句，但一条语句不能写在多行上。

A. 对 B. 错

9. 在 C 语言中，扩展名为 .h 的文件称为头文件，常用于组织 C 标准函数库中的函数。

A. 对 B. 错

10. 注释语句会增加编译结果的复杂性，因此要尽量减少注释语句的数量。

A. 对 B. 错 ( 注释语句在预处理阶段被删除 , 不影响编译 )

11. 声明变量的含义是通知编译系统为变量分配合适大小的存储空间。

A. 对 B. 错

12. C 语言不允许使用关键字作为变量的名称，但可以使用保留字为变量命名。

A. 对 B. 错

13. 符号常量提高了程序的可读性，但降低了程序维护的方便性。

A. 对 B . 错

14. 变量声明把一个具体的标识符名称和计算机内存中的一个特殊的位置联系起来，同时确定了该位置存储的信息类型。

A. 对 B. 错

15. C 语言允许在同一条语句中定义多个相同类型的变量，其间用分号进行分隔 。

A. 对 B. 错

16. 在 ISO/ANSI 1999 规定中，允许将变量声明放在程序中的任何位置。

A. 对 B. 错

17. 不同类型的数据在内存中所占存储单元的大小不同，内部存储方式不同，取值范围不同，甚至能够参与的运算种类也不相同 。

A. 对 B. 错

18. 有符号和无符号整数的区别仅在于对数据最高位的解释不同 。若最高位解释为数据位，则为 signed 类型；若最高位解释为符号位，则为 unsigned 类型。

A. 对 B. 错

19. 默认形式的实数常量的数据类型为 float 类型。

A. 对 B. 错

20. 实型变量能表示的数字的有效位数是有限的。

A. 对 B. 错

21.sizeof( ) 是 C 语言提供的标准库函数，用于测试数据类型在系统中所占的字节数。

A. 对 B. 错

22. 使用转义字符可输出 ASCII 码表中的任一字符。

A. 对 B. 错

23.char 类型数据内部采用整数存储方式，因此它和 int 型数据可以互为通用。

A. 对 B. 错

24. 在 C 语言中，保存字符串“ B ”实质上是保存字符 ' B ' 和 ' \0 ' 两个符号。

A. 对 B. 错

25. 在 C 语言中，允许对两个字符型数据进行加法运算。

A . 对 B. 错

（二）、选择题

1 . C 语言是一种 \_\_\_\_\_\_ 。

A. 汇编语言 B. “解释型”语言 C. “编译型”语言 D. 面向对象语言

2 . 下列说法中，正确的是 \_\_\_\_\_\_ 。

A. 机器语言与硬件相关，但汇编语言与硬件无关

B. 不同的计算机类型，其能理解的机器语言相同

C. 汇编语言采用助记符提高程序的可读性，但同样属于低级语言

D. 汇编源程序属于低级语言程序，计算机可以直接识别并执行

3. 下面有关高级语言的说法中，正确的是 \_\_\_\_\_\_ 。

A. 高级语言编写的程序可读性好，执行效率也最高

B. 高级语言程序必须翻译成机器语言程序，计算机才能执行

C. 解释方式和编译方式相比，具有占用内存少、执行速度快的特点

D.C 语言是一种解释型高级语言

4. 下面有关 C 语言特点的说法中，错误的是 \_\_\_\_\_\_ 。

A.C 语言编写的代码较为紧凑，执行速度也较快

B.C 语言不仅适合编写各种应用软件，还适于编写各种系统软件

C. C 语言是一种模块化和结构化的语言

D.C 语言编写的程序通常不具备移植性

5. 下面有关程序设计技术的说法中，错误的是 \_\_\_\_\_\_ 。

A. 机器语言和汇编语言编程通常采用面向计算机的程序设计方式

B. 面向过程的程序设计方法通常采用结构化和模块化设计方法

C. 面向对象的程序设计方法采用客观世界的描述方式

D. C 语言是一种兼顾面向过程和面向对象的高级程序设计语言

6. 下面有关 C 程序操作过程的说法中，错误的是 \_\_\_\_\_\_ 。

A. C 源程序经过编译，得到的目标文件即为可执行文件

B.C 源程序的链接实质上是将目标代码文件和库函数等代码进行连接的过程

C.C 源程序不能通过编译，通常是由于语法错误引起的

D. 导致不能得到预期计算结果的主要原因是程序算法考虑不周

7. 一个 C 语言程序是由 \_\_\_\_\_\_\_\_ 组成。

A. 主程序 B. 子程序 C. 函数 D. 过程

8.C 源程序中不能表示的数制是 \_\_\_\_\_\_ 。

A. 二进制 B. 八进制 C. 十进制 D. 十六进制

9. 一个 C 语言程序总是从 \_\_\_\_\_\_\_\_\_\_ 开始执行。

A. 主程序 B. 子程序 C. 主函数 D. 函数

10. 以下叙述中正确的是 \_\_\_\_\_\_\_ 。

A. 在 C 程序中， main 函数必须位于程序的最前面

B.C 程序的每一行中只能写一条语句

C. 在对一个 C 程序进行编译的过程中，可发现注释中的拼写错误

D. C 语言本身没有输入输出语句

11. 以下叙述中正确的是 \_\_\_\_\_\_\_\_\_ 。

A.C 程序的基本组成单位是语句 B.C 程序中的每一行中只能写一条语句

C. C 语句必须以分号结束 D.C 语句必须在一行内写完

12. 以下叙述中错误的是 \_\_\_\_\_\_\_\_\_ 。

A. 函数是 C 程序的基本组成单位 B. 函数体一般由一组 C 语句序列组成

C. printf 是 C 语言提供的输出语句 D. 函数通常分为库函数和用户自定义函数两种

13. 以下程序的输出结果是 \_\_\_\_\_\_ 。

#include " stdio.h "

void main()

{ int a=2,b=3,c=4;

printf( "%d,%d,%d\n",c,b,a ); }

A.4 3 2 B.2,3,4 C. 4,3,2 D.2 3 4

14. C 语言中，最基本的数据类型是 \_\_\_\_\_\_\_\_\_ 。

A. 整型、实型、逻辑型 B. 整型、实型、字符型

C. 整型、字符型、逻辑型 D. 整型、实型、逻辑型、字符型

15. 下面有关变量声明的说法中，正确的是 \_\_\_\_\_\_\_\_\_ 。

A.C 语言中不用先声明变量，需要时直接使用即可

B. 每个变量的存储空间大小由数据类型和编译环境共同决定

C. 在 VC++6.0 环境下，为 int 型变量分配的存储空间大小为 2 个字节

D. 变量声明时，不能进行赋值操作

16. 若 x 和 y 为整型变量，对于 scanf( " a=%d,b=%d " ,&x,&y); 语句，可使 x 和 y 的值分别为 10 和 20 的正确输入方法是 \_\_\_\_\_\_\_\_\_ 。

A.10 20 B.10, 20 C.a=10 b=20 D. a=10,b=20

17. 下列标识符中，不合法的用户标识符为 \_\_\_\_\_\_\_\_\_ 。

A.aBa B.\_11 C.a\_1 D. a&b

18. 下列标识符中，合法的用户标识符为 \_\_\_\_\_\_\_\_\_ 。

A. month B.5xy C.int D.your name

19 . \_\_\_\_\_\_\_\_\_ 是 C 语言提供的合法的数据类型关键字。

A.Boolean B. signed C.integer D.Char

20 . 不合法的字符常量是 \_\_\_\_\_\_\_\_\_ 。

A . '\678' B . '\"' C . ' ' D . '\4'

21 . 不正确的字符串常量是 \_\_\_\_\_\_\_\_\_ 。

A. 'abc' B."12'12" C."0" D." "

22. 关于下面的程序，正确的说法是 \_\_\_\_\_\_\_\_\_ 。

#include stdio.h

void main()

{ float a=b=2, result;

result=a/b;

printf("result=%f\n", result); }

A. 程序可正常编译，结果为 result=1.000000 B. 共有 1 处语法错误

C. 共有 2 处语法错误 D. 共有 3 处语法错误

23. 下面变量声明的语句中，错误的是 \_\_\_\_\_\_\_\_\_ 。

A . char c=B; B . int a=3 C. char c=65; D . float area=0;

24. 关于下面的程序，正确的说法是 \_\_\_\_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int x,y,z;

x=y=z=1;

printf("x=%d,y=%d,z=%d\n",x,y,z); }

A. 程序无误，能正常通过编译

B.#include <stdio.h> 有误 ， 需改为 #include "stdio.h"

C.int x,y,z; 有误，需改为 int x; y;z;

D.x=y=z=1; 有误，需改为 x=1; y=1;z=1;

25. 已知字母 A 的 ASCII 码为十进制数 65 ，且 c2 为字符型，则执行语句 c2='A'+'6'-'2'; 后， c2 中的值为 \_\_\_\_\_\_\_\_\_\_ 。

A . 69 B . C C . D D . E

26 . 为了计算 s=10! ，则定义变量 s 时应该使用的数据类型是 \_\_\_\_\_\_\_\_ 。

A . int B . unsigned

C . long D . 以上三种类型均可

27 . 以下选项中合法的标识符是 \_\_\_\_\_\_\_\_ 。

A . 1\_1 B . 1-1 C . \_11 D . 1\_

28 . 下列关于 C 语言用户标识符的叙述中正确的是 \_\_\_\_\_\_\_\_ 。

A . 用户标识符中可以出现下画线和中划线（减号）

B . 用户标识符中不可以出现中划线，但可以出现下画线

C . 用户标识符中可以出现下画线，但不可以放在用户标识符的开头

D . 用户标识符中可以出现下画线和数字，它们都可以放在用户标识符的开头

29 . 以下选项中，能用作数据常量的是 \_\_\_\_\_\_\_\_ 。

A . 0119 B . o126 C . 2.5e2.5 D . 119L

30 . 以下关于 short 、 int 和 long 类型数据占用内存大小的叙述中正确的是 \_\_\_\_\_\_\_\_ 。

A . 均占 4 个字节 B . 根据数据的大小决定所占内存的字节数

C . 由用户自己定义 D . 由 C 语言编译系统决定

31. 若已定义 char c= ' \ 010 ' ，则变量 c 所占的字节数为 \_\_\_\_\_\_\_\_\_\_ 。

A . 1 B . 2 C . 3 D . 4

32. 要定义双精度实型变量 a 和 b ，并初始化为数值 7 ，则正确的语句是 \_\_\_\_\_\_\_\_\_\_ 。

A . double a,b;a=b=7; B . double a=b=7;

C . double a=b=7; D . double a=7,b=7;

33. 已知字符 a 的 ASCII 码值为 97 ，则以下程序的输出结果是 \_\_\_\_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ char ch='c';

printf("%c%d\n",ch,ch); }

A . c99 B . c98 C . c97 D . 编译出错

34 . 以下程序运行后的输出结果是 \_\_\_\_\_\_\_\_\_ 。

#include “ stdio.h ”

void main()

{ char c;

c='B'+32;

printf("%c\n",m); }

A . B B . b C . B32 D . b32

35 . 以下程序运行后的输出结果是 \_\_\_\_\_\_\_\_\_ 。

#include " stdio.h "

void main()

{ char c1,c2;

c1='a'; c2='b';

c1=c1-32;

c2=c2-32;

printf("%c %c\n",c1,c2);

printf("%d %d\n",c1,c2); }

A. A B B.A B C.a b D.a b

65 66 97 98 65 66 97 98

36. 如果将一个函数的返回值类型说明为 void ，则表示 \_\_\_\_\_\_\_\_\_ 。

A. 该函数可以返回任意类型的值

B. 该函数不能返回任何值

C. 该函数可以返回基本类型的值

D. 该函数是 main 函数，是程序运行的起点

37. 对于 int 型常量，不正确的是 \_\_\_\_\_\_\_\_\_\_ 。

A .029 B.-25 C.0x2A D.-0X28

38. 下列整型常量，错误的是 \_\_\_\_\_\_\_\_\_\_ 。

A.025 B.285L C. -285u( u 表示无符号 , 怎么还有负号 ) D.285lu

39. 以下有关整型变量的定义中，错误的是 \_\_\_\_\_\_\_\_\_\_\_ 。

A.short int x; B.long x; C. unsigned x; D.short x;

40. 下列实型常量，错误的是 \_\_\_\_\_\_\_\_\_\_ 。

A.3.2f B.-2.2E-2F C.-12.0 D. e3

41. 在 VC++6.0 环境下， sizeof(int) 和 sizeof(float) 的值分别为 \_\_\_\_\_\_\_\_\_\_ 。

A.2,2 B.2,4 C. 4,4 D.4,8

42. 对于浮点型数据，下列说法中不正确的是 \_\_\_\_\_\_ 。

A. 浮点数在存储时存在舍入误差

B. 程序设计时，不能直接对两个浮点型数据进行相等比较

C. 将一个很小的浮点数和一个很大的浮点数进行相加，结果可能不正确

D. C 语言中的浮点型变量有 float 和 double 两种

43. 若 ch 为字符型变量，则不能使 ch 表示字符 ' B ' 的语句是 \_\_\_\_\_\_\_\_\_\_ 。

A . ch=B; B . ch=66; C . ch='\102'; D . ch='\x42';

44. 以下程序运行后的输出结果是 \_\_\_\_\_\_\_\_\_ 。

#include "stdio.h"

int f(int x,int y)

{ return 2\*x+y; }

void main()

{ int a=3,b=2,t=0;

t= f(b,a);

printf("%d\n",t); }

A. 0 B. 5 C. 7 D. 8

45. 以下程序运行后的输出结果是 \_\_\_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ char ch= ' a ' ;

int x=5;

float s=2.5F;

printf("%d,%d,%d\n",sizeof(ch),sizeof(x),sizeof(s)); }

A.1,2,4 B. 1,4,4 C.1,4,8 D.a,5,2.5

（三）、编程题

1. 已知长方形的长和宽分别为 x 和 y ，计算其周长和面积。

2. 编写一个程序，其功能为：从键盘上输入两个整型数据，分别存放在整型变量 a 和 b 中，然后输出表达式 a/b 和 a%b 的值。

3. 编写一个程序，其功能为：从键盘上输入一个浮点数，然后分别输出该数的整数部分和小数部分。

4 . 编写一个程序，其功能为：从键盘上输入一个小写字母，显示这个小写字母及它所对应的大写字母以及它们的 ASCII 码值。

提示：大写字母 A ～ Z 的 ASCII 码值为 65 ～ 90 ，小写字母 a ～ z 的 ASCII 码值为 97 ～ 122 。可见，对应的大小写字母的 ASCII 码值相差 32 ，所以大写字母转换成小写字母就是将其 ASCII 值加上 32 ，小写字母转换成大写字母就是将其 ASCII 值减去 32 。

5. 输入一个华氏温度 F ，根据公式 c=5(F-32)/9 计算输出对应的摄氏温度。要求：输入要有提示，输出要有说明。

四、习题答案

（一）、是非题

1 .A 2.B 3.B 4.A 5.B 6.A 7.A 8.B 9.A 10.B

11 .A 12.B 13.B 14.A 15.B 16.A 17.A 18.B 19.B 20.A

21 .B 22.A 23.B 24.A 25.A

（二）、选择题

1 .C 2.C 3.B 4.D 5.D 6.A 7.C 8.A 9.C 10.D

11 .C 12.C 13.C 14.B 15.B 16.D 17.D 18.A 19.B 20.A

21 .A 22.C 23.A 24.A 25.A 26.C 27.C 28.B 29.D 30.D

31 .A 32.D 33.A 34.B 35.A 36.B 37.A 38.C 39.C 40.D

41 .C 42.D 43.A 44.C 45.B

## 第 2 章 顺序结构

二、习题

（一）是非题

1. 表达式 1/4+2.75 的值是 3 。

A. 对 B. 错 ( 表达式的值为？ )

2.C 语言中浮点型变量可以进行自增或自减操作。

A. 对 B. 错

3. 如果整型变量 a 、 b 、 c 已经正确定义，那么 c=b=a=012 是一个正确的表达式。

A. 对 B. 错

4. 算术运算符和赋值运算符的结合方向均由左自右。

A. 对 B. 错

5. 在进行赋值转换时，变量的类型被自动转换为赋值号右边的表达式值的类型。

A. 对 B. 错

6. 语句 char c="A"; 定义了字符型变量 c ，并将其初始化为 A 。

A. 对 B. 错

7. 运算符“ % ”的操作数不允许为单精度和双精度浮点型，允许为字符型和整型。

A . 对 B. 错

8. 运算符 ++ 的优先级高于运算符 + 。

A. 对 B. 错

9. 逗号运算符的求值顺序是从左到右的，逗号表达式的值为第一个表达式的值。

A. 对 B. 错

10. “ = ”运算符用于判断两个数是否相等。

A. 对 B. 错

11 . C 语言本身不提供输入输出语句，但可以通过输入输出函数来实现数据的输入输出。

A. 对 B. 错

12. 在使用函数 scanf 输入数据时必须与函数参数指定的输入格式一致。

A. 对 B. 错

13.getchar() 函数用于输入单个字符， putchar() 函数用于输出单个字符。

A. 对 B. 错

14.123 、 029 、 -0xEFF 均是合法的整型常量。

A. 对 B. 错

15. 假定已有定义 char c ；， sizeof c 中的 sizeof 是容量运算符，而 sizeof(char) 中由于使用了括号形式，称为容量函数。

A. 对 B. 错

16. 假定已有整型变量 x 的定义，则 (x+1)++ 是一个合法的表达式。

A. 对 B. 错

17. 假定已有整型变量初始化语句 int x=2; ，那么表达式 x++,x \*=2+1 的值为 9 。

A. 对 B. 错

18. 假定已有变量定义语句： int m=3,n=2; ，那么 (float)(m/n) 表达式的值是 1.5 。

A. 对 B. 错

19.C 语言中，表达式 1/2\*2 的值为 0 。

A. 对 B. 错

20. 若 scanf 函数的格式说明中有一个空格，在输入时也一定要在对应位置输入一个空格。

A. 对 B. 错

（二）选择题

1. 数字字符 0 的 ASCII 值为 48 ，则以下程序运行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char a='1',b='2'; printf("%c,",b++); printf("%d\n",b-a); }

A.3,2 B.50,2 C. 2,2 D.2,50

2. 以下程序运行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int m=12,n=34;

printf("%d%d",m++,++n);

printf("%d%d\n",n++,++m); }

A. 12353514 B.12353513 C.12343514 D.12343513

3. 若整型变量 a 、 b 、 c 、 d 中的值依次为 1 、 2 、 3 、 4 。则表达式 a+b/d\*c 的值是 \_\_\_\_\_\_ 。

A. 1 B.2.5 C.0.25 D.2

4. 以下程序运行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a,b,c;

　 a=10; b=20; c= a%b+a/b;

　 printf("%d %d %d\n",a,b,c); }

A.10200 B. 10 20 10 C.102011 D.10 20 1

5. 以下程序的功能是：给 r 输入数据后计算半径为 r 的圆面积 s 。程序在编译时出错，出错的原因是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int r; float s;

scanf("%d",&r);

s=\*p\*r\*r;

printf("s=%f\n",s); }

A. 注释语句书写位置错误 B. 存放圆半径的变量 r 不应该定义为整型

C. s=\*p\*r\*r; 语句中使用了非法变量 D. 输出语句中格式描述符非法

6. 设有定义： int k=1,m=2; float f=7; ，则以下选项中错误的表达式是 \_\_\_\_\_\_ 。

A.k=k+=k B.-k++ C. k%int(f) D.f=k%m

7. 设有定义： int a=2,b=3,c=4; ，则以下选项中值为 0 的表达式是 \_\_\_\_\_\_ 。

A.a%b%c B. a/b/c C.a=b=c D.a,b,c

8. 有以下程序段 :int k=0,a=1,b=2,c=3; k=a+=b-=c; ，执行该程序段后， k 的值是 \_\_\_\_\_\_ 。

A. 0 B.1 C.2 D.3

9. 以下程序运行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char c; int n=100;

float f=10; double x;

x=f\*=n/=(c=50);

printf("%d %f\n",n,x); }

A.2 20 B. 2 20.000000 C.100 10 D.50 10.000000

10. 已知字母 A 的 ASCII 码为 65 ，则以下程序运行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char a,b;

a='A'+'5'-'3'; b=a+'6'-'2';

printf("%d%c\n",a,b); }

A.6771 B. 67G C.CG D.C71

11. 表达式 3.6-5/2+1.2+5%2 的值是 \_\_\_\_\_\_ 。

A.3.3 B. 3.8 C.4.3 D.4.8

12. 若变量 x 、 y 已正确定义并赋值，以下符合 C 语言语法的表达式是 \_\_\_\_\_\_\_\_ 。

A. ++x,y=x-- B.x+1=y ( 反了 ) C.x=x+10=x+y ( x+10=x+y 赋值报错 ) D.double(x)/10

13. 以下程序运行后的输出结果是 \_\_\_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int x,y,z;

　 x=y=1;

　 z=x++,y++,++y;

　 printf("%d,%d,%d\n",x,y,z); }

A.2,3,3 B.2,3,2 C. 2,3,1 D.2,2,1

14. 以下选项中，值为 1 的表达式是 \_\_\_\_\_\_\_\_ 。

A.1-'0' B. 1-'\0' C. '1'-0 D. '\0'-'0'

15. 设有定义： int k=0; ，以下选项的 \_\_\_\_\_\_\_\_ 与其他三个表达式的值不相同。

A. k++ B.k+=1 C.++k D.k+1

16. 执行以下程序后的输出结果是 \_\_\_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a=10;

a=(3\*5,a+4);

printf("a=%d\n",a); } // 考察逗号运算符

A.a=10 B. a=14 C.a=15 D.a=19

17. 若变量均已正确定义并赋值，以下合法的 C 语言赋值语句是 \_\_\_\_\_\_\_\_ 。

A. x=y=5; B.x=n%2.5; C.x+n=i; D.x=5=4+1;

18. 设变量已经正确定义并赋值，以下正确的表达式是 \_\_\_\_\_\_\_\_ 。

A.x=y\*5=x+z B.int(15.8%5) C. x=y+z\*5,++y D.x=25%5.0

19. 以下不能正确表示代数式 2ab/cd 的 C 语言表达式是 \_\_\_\_\_\_\_\_ 。

A.2\*a\*b/c/d B.a\*b/c/d\*2 C.a/c/d\*b\*2 D. 2\*a\*b/c\*d

20. 设变量 a 和 b 已正确定义并赋初值。请写出与 a-=a+b 等价的赋值表达式 \_\_\_\_\_\_\_\_ 。

A.a=a-a+b B.a-a=b C. a=a-(a+b) D.a=-a+b

21. 设有定义 :int x=2; ，以下表达式中，值不为 6 的是 \_\_\_\_\_\_\_\_ 。

A.x\*=x+1 B.x++,2\*x C.x\*=(1+x) D. 2\*x,x+=2

22. 表达式 (int)((double)9/2)-(9)%2 的值是 \_\_\_\_\_\_\_\_ 。

A.0 B. 3 C.4 D.5

23. 若有定义 int x=10; ，则表达式 x-=x+x 的值为 \_\_\_\_\_\_\_\_ 。

A.-20 B .-10 C.0 D.10

24. 若有定义 double a=22;int i=0,k=18; ，则以下有错的语句是 \_\_\_\_\_\_\_\_ 。

A.a=a++,i++; B.i=(a+k)/(k-i); C. i=a%11; D.i=!a;

25. 以下程序运行后的结果是 \_\_\_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a=2,b=2,c=2; printf("%d\n",a/b%c); }

A.0 B. 1 C.2 D.3

26. 若有定义 int a;long b; double x,y; ，则以下选项中正确的表达式是 \_\_\_\_\_\_\_\_ 。

A. a%( int )(x-y) B.a=b=x,y; C.(a\*y)%b D.y=x+y=x

27. 表达式 a+=a-=a=9 的值是 \_\_\_\_\_\_\_\_ 。

A.-9 B. 0 C.9 D.18

28. 若有定义 int a=3,b=2,c=1; ， 以下选项中错误的赋值表达式是 \_\_\_\_\_\_\_\_ 。

A. a=(b=4)=3 B.a=b=c+1; C.a=(b=4)+c; D.a=1+(b=c=4);

29. 若有定义 int x=12,y=8,z ；， 在其后执行语句 z=0.9+x/y ；， 则 z 的值为 \_\_\_\_\_\_\_\_ 。

A. 1 ( 右边表达式的值为 1.9 ，赋值给 z 舍弃小数 ) B.1.9 C.2 D.2.4

30. 以下程序运行后输出的结果是 \_\_\_\_\_\_\_\_ 。

#include<stdio.h>

main()

{ int a=0,b=0,c=0;

c=(a-=a-5);(a=b,b+=4);

printf("%d,%d,%d", a,b,c); }

A. 0,4,5 B.4,4,5 C.4,4,4 D.0,0,0

31. 设变量均已正确定义并且赋值，以下与其他三组输出结果不同的一组语句是 \_\_\_\_\_\_\_\_ 。

A.x++; printf("%d\n",x); B.n=++x; printf("%d\n",n);

C.++x; printf("%d\n",x); D. n=x++; printf("%d\n",n);

32. 以下程序运行后输出的结果是 \_\_\_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a=1,b=0; printf("%d,",b=a+b); printf("%d",a=2\*b); }

A.0,0 B.1,0 C.3,2 D. 1,2

33. 以下程序运行后输出的结果是 \_\_\_\_\_\_\_\_ 。

#include <stdio.h>

main(){ int k=011; printf("%d\n"k++); }

A.12 B.11 C.10 D. 9

34. 有以下程序：

#include <stdio.h>

main()

{ int m,n,p;

scanf("m=%dn=%dp=%d",&m,&n,&p);

printf("%d%d%d\n",m,n,p); }

若想使变量 m 中的值为 123 ， n 中的值为 456 ， p 中的值为 789 ，则正确的输入是 \_\_\_\_\_\_ 。

A. m=123n=456p=789 B.123,456,789

C.m=123,n=456,p=789 D.123 456 789

35. 以下程序运行后输出的结果是 \_\_\_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a,b,d=25; a=d/10%9; b=a+-1; printf("%d,%d\n",a,b); }

A.6,1 B. 2,1 C.6,0 D.2,0

36. 以下叙述中正确的是 \_\_\_\_\_\_ 。

A. 调用 printf 函数时，必须要有输出项

B. 使用 putchar 函数时，必须在之前包含头文件 stdio.h

C. 在 C 语言中，整数可以以十二进制、八进制或十六进制的形式输出

D. 调用 getchar 函数读入字符时，可以从键盘上输入字符所对应的 ASCII 码

37. 程序如下，下叙述中正确的是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char a1='M',a2='m'; printf("%c\n",(a1, a2)); } // 逗号运算符是最后的值

A. 程序输出大写字母 M B. 程序输出小写字母 m

C. 格式说明符不足，编译出错 D. 程序运行时产生出错信息

38. 以下程序运行时若输入： a< 回车 > ，则叙述正确的是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char c1='1',c2='2'; c1=getchar(); c2=getchar();putchar(c1); putchar(c2); }

A. 变量 c1 被赋予字符 a ， c2 被赋予回车符

B. 程序将等待用户输入第 2 个字符

C. 变量 c1 被赋予字符 a ， c2 中仍是原有字符 2

D. 变量 c1 被赋予字符 a ， c2 中将无确定值

39. 设有定义： int a; float b; ，执行 scanf("%2d%f",&a,&b); 语句时，若从键盘输入： 876 543.0 ， a 和 b 的值分别是 \_\_\_\_\_\_\_\_ 。

A.876 和 543.0 B. 87 和 6.0 C.87 和 543.0 D.76 和 543.0

40. 若在定义语句： int a,b,c; 之后，接着执行以下选项中的语句，则能正确执行的语句是 \_\_\_\_\_\_\_\_ 。

A.scanf("%d",a,b,c); B. scanf("%d%d%d",&a,&b,&c);

C.scanf("%d%d%d",&a, b, c); D.scanf("%d%d%d",a,b,c) ;

41. 已知字符 'A' 的 ASCII 码值是 65 ， 字符变量 c1 的值是 'A' ， c2 的值是 'D' 。执行语句 printf("%d,%d",c1,c2-2); 后，输出结果是 \_\_\_\_\_\_\_\_ 。

A.A,B B.A,68 C .65,66 D.65,68

42. 程序段 :int x=12; double y=3.141593;printf("%d%8.6f",x,y); 的输出结果是 \_\_\_\_\_\_\_\_ 。

A. 12 3.141593 B.123.141593 C.12,3.141593 D.123.1415930

43. 若要求从键盘读入含有空格字符的字符串，应使用函数 \_\_\_\_\_\_\_\_ 。

A.getc() B. gets() C.getchar() D.scanf()

44. 有以下程序，其中 %u 表示按无符号整数输出，程序运行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ unsigned int x=0xFFFF; printf("%u\n",x); }

A.-1 B. 65535 C.32767  D.0XFFFF

（三）、编程题

1. 编写一个程序，从键盘输入圆的半径，计算并输出该圆的周长、面积和体积。

2. 输入四个季度的销售额（整型），计算并输出各季度自年初起的累计销售额，计算并输出第一季度外的三个季度的环比增长率（环比增长率是本期与相比上一期的增长率，针对本题目，即为本季度对上季度的增长率）。

3. 从键盘输入四门课程的（整数）分数，输出成绩信息，计算并输出平均分数。要求输入输出有一定提示，输出的平均分数保留二位小数。

四、习题答案

（一）是非题

1 .B 2.A 3.A 4.B 5.B 6.B 7.A 8.A 9.B 10.B

11 .A 12.A 13.A 14.B 15.B 16.B 17.A 18.B 19.A 20.B

（二）选择题

1 .C 2.A 3.A 4.B 5.C 6.C 7.B 8.A 9.B 10.B

11 .B 12.A 13.C 14.B 15.A 16.B 17.C 18.C 19.D 20.C

21 .D 22.B 23.B 24.C 25.B 26.A 27.B 28.A 29.A 30.A

31 .D 32.D 33.D 34.A 35.B 36.B 37.B 38.A 39.B 4 0 .B

41.C 42.A 43 . C 44.B

## 第 3 章 选择结构

二、习题

（一）、是非题

1. 关系运算符是双目运算符，其功能是将两个运算对象进行大小比较。

A. 对 B. 错

2. 对于与运算 “ && ”，只有当两个运算对象都为真时运算结果才为真。

A. 对 B. 错

3. 使用条件表达式可以构成各种各样的两路选择结构。

A. 对 B. 错

4. 条件可以是任何类型的表达式，如逻辑型，关系型，数值型等，单个已赋过值的变量或常量也可以作为表达式的特例。

A. 对 B. 错

5. else 子句不能单独作为语句使用，它是 if 语句的一部分，必须与 if 配对使用。

A. 对 B. 错

6. C 语言规定， else 总是与它上面最远的 if 配对。

A. 对 B. 错

7. 在 switch 语句中，每一个 case 后的常量表达式的值不能相同，因为选择结构中只允许一个分支所对应的语句组被执行 。

A . 对 B. 错

8. 在执行 switch 选择结构时，从匹配表达式的相应 case 处入口，一直执行到 break 语句或到达 switch 的末尾为止

A. 对 B. 错

9. 在 C 语言中，运算符 “ = ”与 “ == ”的含义都是等于。

A. 对 B. 错

10. 当进行比较的运算对象是字符型数据时，则按字符的 ASCII 码值从左到右逐个对应比较，先大为大。

A. 对 B. 错

11. 判断字符变量 ch 的值是否为数字字符的表达式为 (ch>='0') && (ch<='9') 。

A. 对 B. 错

12. 条件运算符 “ ? : ”是 C 语言中唯一的一个三目运算符，该运算符可以嵌套使用。

A. 对 B. 错

13. 在 C 语言中用 0 表示逻辑值 “ 真 ” ，用非 0 表示逻辑值 “ 假 ” 。

A. 对 B. 错

14. 在 C 语言中，所有的逻辑运算符的优先级都一样。

A. 对 B. 错

15. 在 C 语言中，逻辑运算符的优先级高于算术运算符和关系运算符。

A. 对 B. 错

16. 数学表达式 “ |x|>4 ”对应的 C 语言 表达式是 “ x<-4||x>4 ”。

A. 对 B. 错

17. 与表达式 “ !(x>0) ”等价的表达式是 “ x<=0 ”。

A. 对 B. 错

18. 与表达式 “ !0 ”等价的表达式是 “ 1 ”。

A. 对 B. 错

19. 当 a=1,b=2,c=3 时，执行语句“ if(a>c) b=a;a=c;c=b; ”后，变量 c 中的值为 3 。

A. 对 B. 错

a=1,b=2,c=3;

if(a>c) b=a;

a=c;

c=b;

b=2

20. 当 a=1,b=2,c=3 时，执行语句“ if(a>c) {b=a;a=c;c=b;} ”后，变量 c 中的值为 2 。

A. 对 B. 错

a=1,b=2,c=3;

if(a>c)

{b=a;a=c;c=b;}

C=3

（二）、选择题

1 . 判断 char 型变量 ch 是否为小写字母的正确表达式是 \_\_\_\_\_\_ 。

A.'a'<=ch<='z' B.(ch>='a') & (ch<='z')

C. (ch>='a') && (ch<='z') D.('a'<=ch) AND ('z'>=ch)

2 . 为表示关系 100 ≥ b ≥ 0 ，应使用 C 语言表达式 \_\_\_\_\_\_ 。

A. (100>=b)&&(b>= 0 ) B.(100>=b)and(b>= 0 )

C.100>=b>= 0 D.(100>=B)&&(B>= 0 )

3. 以下运算符中优先级最高的运算符为 \_\_\_\_\_\_ 。

A. ! B.&& C.!= D.%

4 . 设 x 、 y 和 z 是 int 型变量，且 x=3,y=4,z=5, 则下面表达式中值为 0 的是 \_\_\_\_\_\_ 。

A.'x'&&'y' B.x<=y

C.x||y+z&&y-z D. !((x<y)&&!z||1)

!((3<4)&&!5||1)

!(1&&0||1)

!(0||1)

!1

5. 执行以下程序段后 ， x 的值为 \_\_\_\_\_\_ 。

int a=14,b=15,x;

char c='A';

x=(a&&b)&&(c<'B');

A.true B .1 C.false D.0

6 . 下列表达式中 ， \_\_\_\_\_\_ 不是关系表达式。

A.x%2= =0 B. !(x%2)

C.(x/2\*2-x)= =0 D.x%2!=0

7. 执行以下程序后，输出的结果是 \_\_\_\_\_\_ 。

include "stdio.h"

main()

{ int a=2,b=-1,c=2;

if(a<b) if(b<0) c=0; else c+=1;

printf( "%d\n",c ); }

A.0 B.1 C. 2 D.3

8. 执行以下程序后，输出的结果是 \_\_\_\_\_\_ 。

include "stdio.h"

main()

{ int w=4,x=3,y=2,z=1;

printf( "%d\n",(w<x?w:z<y?z:x) ); }

A.4 B.2 C. 1 D.3

9. 执行以下程序段后，输出结果是 \_\_\_\_\_\_ 。

int a=3,b=5,c=7;

if(a>b) a=b;c=a;

if(c!=a) c=b;

printf( "%d, %d, %d\n",a,b,c );

A. 程序段有语法错误 B. 3,5,3

C.3,5,5 D.3,5,7

10 . 下面程序段的输出结果是 \_\_\_\_\_\_ 。

int a=-1,b=4,k;

k=(a++<=0)&&(!b--<=0);

printf("%d,%d,%d",k,a,b);

A.0,0,3 B.0,1,2

C. 1,0,3 D.1,1,2

11 . 若有定义： float x=1.5;int a=1,b=3,c=2; , 则正确的 switch 语句是 \_\_\_\_\_\_ 。

A.switch(x)//() 里为整型 , 字符 , 枚举 B. switch(int(x))

{ case 1.0: printf("\*\n"); { case 1: printf("\*\n");

case 2.0: printf("\*\*\n"); case 2: printf("\*\*\n");

} }

C.switch(a+b) D.switch(a+b)

{ case 1: printf("\*\n"); { case 1: printf("\*\n");

case 2+1: printf("\*\*\n"); case c: printf("\*\*\n");

} }

12 . 下面程序段的输出结果是 \_\_\_\_\_\_ 。

int n= 'c';

switch(n++) // n 先用，后自增 (n=’c’) 。

{default: printf("error"); break;

case 'a':

case 'A':

case 'b':

case 'B': printf("good"); break;

case 'c':

case 'C': printf("pass");

case 'd':

case 'D': printf("warn"); }

A. passwarn B.passerror

C.goodpasswarn D.pass

13 . 下述程序段的输出结果是 \_\_\_\_\_\_\_ 。

int a=2,b=3,c=4,d=5;

int m=2,n=2;

a=(m=a>b)&&(n=c>d)+5;

printf("%d, %d ",n,a);

A. 2,5 B.0,5 C.2,6 D.0,6

14 . 若 a 是数值类型，则逻辑表达式 (a==1)||(a!=1) 的值是 \_\_\_\_\_\_\_ 。

A.0 B. 1 C.2 D. 不能确定

15 . 已知 int x=10,y=20,z=30; ，以下语句执行后 x,y,z 的值是 \_\_\_\_\_\_\_ 。

if(x>y)

z=x; x=y; y=z;

printf("%d,%d,%d",x,y,z);

A.10,20,30 B.20,30,20

C.20,30,10 D. 20,30,30

16 . 当 a=1,b=3,c=5,d=4,x=0 时，执行完下面一段程序后 x 的值是 \_\_\_\_\_\_\_ 。

if(a>b)

if(c<d) x=1;

else if(a>c)

if(b<d) x=2;

else x=3;

else x=6;

else x=7;

A. 7 B.2 C.3 D.6

17 . 设变量 a 、 b 、 c 、 d 和 y 都已经正确定义并赋值，则以下 if 语句的所表示的含义是 \_\_\_\_\_\_\_ 。

if(a<b)

if(c==d) y=0;

else y=1;

A.a<b 且 c=d 时 y=0 ， a ≥ b 时 y=1

B.a<b 且 c=d 时 y=0 ， a ≥ b 且 c ≠ d 时 y=1

C. a<b 且 c=d 时 y=0 ， a<b 且 c ≠ d 时 y=1

D.a<b 且 c=d 时 y=0 ， c ≠ d 时 y=1

18 . 下述程序的输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int x=1,y=0,a=0,b=0;

switch(x)

{ case 1: switch(y)

{ case 0: a++; break;

case 1: b++;break; }

case 2: a++;b++;break;

case 3: a++;b++;

}

printf("\na=%d,b=%d",a,b); }

A.a=1,b=0 B. a=2,b=1

C.a=1,b=1 D.a=2,b=2

19 . 在执行下述程序时，若从键盘输入 6 和 8 ，则输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a,b,s;

scanf("%d%d",&a,&b);

s=a;

if(a<b)

s=b;

s\*=s;

printf("\n%d",s); }

A.36 B.48 C. 64 D. 以上都不对

20 . 以下程序运行时，输入的值在 \_\_\_\_\_\_\_ 范围时才会有输出结果 。

#inlude <stdio.h>

main()

{ int x;

scanf("%d ",&x);

if(x<=3);else

if(x!=10) printf("%d\n ",x); }

A. 不等于 10 的整数 B. 大于 3 且不等于 10 的整数

C. 大于 3 或等于 10 的整数 D. 小于 3 的整数

21 . 下列程序的输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a=0,b=0,c=0;

if( ++a >0||++b>0) ++c; // 短路运算， ++b 未作运算。

printf("%d,%d,%d",a,b,c); }

A.0,0,0 B.1,1, C. 1,0,1 D. 0,1,1

22 . 有定义语句： int a=1,b=2,c=3; ，则以下 \_\_\_\_\_\_\_ 对应的表达式的值不为真。

A . a!= 'a' B . a+b-c!= ' 0 '-48

C . c-a=='C' - 'A' D . c<='7'+' 0 '

23 . 下列程序的输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int i=1,j=2,k=3;

if(i++==1&&( ++j= =3 ||k++==3)) // 短路运算， k=k+3 未作运算。

printf("%d %d %d",i,j,k); }

A.1 2 3 B.2 3 4 C.2 2 3 D. 2 3 3

24 . 当变量 c 的值 不为 2 、 4 、 6 时，值也为 “ 真 ” 的表达式是 \_\_\_\_\_\_\_ 。

A . (c==2)||(c==4)||(c==6)

B . (c>=2&&c<=6)||(c!=3)||(c!=5)

C . (c>=2&&c<=6)&&!(c%2)

D . (c>=2&&c<=6)&&(c%2!=1)

25 . 下列程序的输出结果是 \_\_\_\_\_\_\_ 。

main()

{ int a=5,b=4,c=3,d=2;

if(a>b>c) printf("%d\n",d);

else if((c-1>=d)==1) printf("%d\n",d+1);

else printf("%d\n",d+2); }

A . 2 B . 3 C . 4 D . 编译时有错且无结果

26 . 己有定义： int x=3,y=4,z=5; ， 则表达式 !(x+y)+z-1&&y+z/2 的值是 \_\_\_\_\_\_\_ 。

A . 6 B . 0 C . 2 D . l

！（ 3+4 ） +5-1&&4+5/2

!7+4&&4+2

0+4&&6

1&&1

27 . 以下程序 运行后的输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a=15,b=21,m=0;

switch(a%3) //15%3=0

{ case 0: m++; break; // 执行后，退出第一层。

case 1: m++;

switch(b%2) // 第二层，未执行

{ default: m++;

case 0: m++; break; }

}

printf( " %d\n " ,m); }

A . 1 B . 2 C . 3 D . 4

28 . 在执行下述程序时，若从键盘输入 6 和 2 ，则输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a,b,k;

scanf("%d,%d",&a,&b);

k=a;

if(a<b) k=a%b;

else k=b%a;

printf("\n%d",k); }

A.5 B.3 C. 2 D.0

29 . 在执行下述程序时，若从键盘输入字母 H ，则输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ char ch;

ch=getchar();

switch(ch)

{ case 'H':printf("Hello! \n");

case 'G':printf("Good morning! \n");

default:printf("Bye\_Bye! \n"); }

}

A.Hello! B.Hello!

Good morning!

C. Hello! D.Hello!

Good morning! Bye\_Bye!

Bye\_Bye!

30 . 设变量 x 和 y 均已正确定义并赋值，则以下 if 语句中，在编译时将产生错误信息的是 \_\_\_\_\_\_\_ 。

A.if(x++); // 空语句 B.if(x>y&&y!=0) ;

C. if(x>0) x- - // 少了“；”号 D.if(y<0){;}

else y++; else x++;

31 . 以下程序的运行后的输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a=1,b=0;

if(--a) b++;

else if(a==0) b+=2;

else b+=3;

printf(" %d\n ",b); }

A. 0 B.1 C. 2 D.3

32 . 下列条件语句中，输出结果与其他语句不同的是 \_\_\_\_\_\_\_ 。

A . if(a) printf( " %d\n ",x ); else printf( " %d\n ",y );

B . if(a==0) printf( " %d\n ",y ); else printf( " %d\n ",x );

C . if(a!=0) printf( " %d\n ",x ); else printf( " %d\n ",y );

D . if(a==0) printf( " %d\n ",x ); else printf( " %d\n ",y );

33 . 以下选项中，能表示逻辑值“假”的是 \_\_\_\_\_\_\_ 。

A.1 B.0.000001 C. 0 D.100.0

34 . 以下程序运行时，若从键盘输入 9 ，则输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a=1;

scanf(" %d ",&a);

if(a++<9)

printf(" %d\n ",a);

else

printf(" %d\n ",a); }

A. 10 B.11 C.9 D.8

35 . 若变量 c 的值为非 0 ，则能正确地将 c 的值赋给变量 a 和 b 的表达式是 \_\_\_\_\_\_\_ 。

A.(a=c)||(b=c) B.(b=a)=c

C. (a=c) && (b=c) D.a=c=b

36 . 设有定义 int i; ，则表达式 (i=1,i=10)? i++||++i:++i 的值是 \_\_\_\_\_\_\_ 。

A.10 B. 1 C.2 D.3

37 . 设 a=2 、 b=7.5 、 c=-3.6 ，则 \_\_\_\_\_\_\_ 中的两个表达式的逻辑值相等 。

A .'a'>'c'||'x'<'y' 与 b-a<=-c/2&&c==-3.6

B .'a'>'c'&&x'<'y' 与 a-7<b-6.5

C . 'a'>'c'||'x'>'y' 与 'a'<'c'&&a>b

D .'a'>'c'&&'x'>'y' 与 'a'<'c'||a<c

38 . 当 y 处于闭区间 [x,z] 时，值为真的表达式是 \_\_\_\_\_\_\_ 。

A. ( y<= z)&&(y>=x) B.(x>=y)and(y>=z)c

C.(x>=y>=z) D.(x>=y)&(y>=z)

39 . 以下程序 执行后的输出结果是 \_\_ \_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a=5,b=4,c=3,d=1;

d=(a>b>c); / /a>b 为真 , 判断结果为 1 ， 1 再和 c 比较 1>c ，为假。表达式值为 0.

printf("%d\n",d); }

A.3 B. 2 C.1 D. 0

40 . 以下程序 执行后的输出结果是 \_\_\_\_\_\_\_ 。

#inlude <stdio.h>

main()

{ int a=2,b=7,c=5;

switch(a>0)

{ case 1:switch(b<10)

{ case 1: printf("@");break;

case 0: printf("!");break; }

case 0:switch(c==5)

{ case 0: printf("\*");break;

case 1: printf("#");break;

default: printf("%");break; }

default: { case 1: printf();break; printf("&");}

printf(" \ n"); }

A.!\*# B.@!# C.@%# D. @#&

四、习题答案

（一）、是非题

1 .A 2.A 3.B 4.A 5.A 6.B 7.A 8.A 9.B 10.A

11 .A 12.A 13.B 14.B 15.B 16.A 17.A 18.A 19.B 20.A

（二）、选择题

1 .C 2.A 3.A 4.D 5.B 6.B 7.C 8.C 9.B 10.C

11 .B 12.A 13.A 14.B 15.D 16.A 17.C 18.B 19.C 20.B

21 .C 22.B 23.D 24.B 25.B 26.D 27.A 28.C 29.C 30.C

31 .C 32.D 33.C 34.A 35.C 36.B 37.C 38.A 39.D 40.D

（三）、编程题（ 略 ）

## 第 4 章 循环结构

二、习题

（一）、是非题

1. C 语言中， do-while 语句构成的循环只能用 break 语句退出。

A. 对 B. 错

2. 在循环外的语句不受循环的控制，在循环内的语句也不受循环的控制。

A. 对 B. 错

3. 从语法角度看， for( 表达式 1; 表达式 2; 表达式 3) 语句中的 3 个表达式均可省略。

A. 对 B. 错

4. for 、 while 和 do while 循环结构的循环体均为紧接其后的第一个语句（含复合语句）。

A. 对 B. 错

5. 由 i=-1;while(i<10) i+=2;i++; 可知，此 while 循环的循环体执行次数为 6 次。

A. 对 B. 错

6. 循环 for( ; ; ) 的循环条件始终为真。

A. 对 B. 错

7. do … while 语句的循环体至少执行 1 次， while 和 for 循环的循环体可能一次也不执行。

A. 对 B. 错

8. break; 语句执行时退出本层循环， continue; 语句结束本次循环。

A. 对 B. 错

9. for 循环、 while 循环和 do while 循环结构之间可以相互转化。

A . 对 B. 错

10. break; 语句执行时退出到包含该 break; 语句的所有循环外。。

A. 对 B. 错

11. 二重循环 for(i=0;i<5;i+=2) for(j=0;j<5;j++) 的循环体的执行次数为 15 。

A . 对 B. 错

外循环 i 执行 3 次 , 内循环 j 执行 5 次。 3\*5=15 次。

12. 执行 i=-1; while(i<10) i+=2;i++; 后， i 的值 为 11 。

A. 对 B . 错

13. int i=1;while(1) i++; 则在任何条件下，该循环语句都会执行无穷多次

A. 对 B. 错

14. while 和 do … while 的功能完全相同。

A. 对 B. 错

15. while 后的表达式只能是逻辑或关系表达式。

A. 对 B. 错

16. 语句 while(E); 中的表达式 E 等价于 E!=0 。

A. 对 B. 错

17. 只能在循环体内和 switch 语句体内使用 break 语句 。

A. 对 B. 错

18. 程序段 int k=10; while(k=0)k=k-1; 中， while 循环执行 10 次 。

A. 对 B. 错

19. 若 i 为整型变量，则 for(i=2;i= =0;)printf("%d",i--); 循环执行次数是 0 次 。

A. 对 B. 错

20. 执行语句 for(i=1;i++<4; ); 后，变量 i 的值是 4 。

A. 对 B. 错

（二）、选择题

1. 下列程序的输出结果是 \_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int i,a=0,b=0;

for(i=1;i<10;i++)

{ if(i%2==0) {a++; continue;}

b++;}

printf("a=%d,b=%d",a,b); }

A. a=4,b=4 B. a=4,b=5 C. a=5,b=4 D. a=5,b=5

2 . 已知 ： int t=0; while(t=1) { ... } ， 则以下叙述正确的是 \_\_\_\_\_\_ 。

A. 循环表达式的值为 0 B. 循环表达式的值为 1

C. 循环表达式不合法 D. 以上说法都不对

3. 设有以下程序段 ，则 \_\_\_\_\_ 。

int x=0,s=0; while(!x!=0) s+=++x; printf("%d",s);

while(!0!=0)

while(1!=0)

while(1) 执行 s=s+(++x) s=1

A. 运行程序段后输出 0 B. 运行程序段后输出 1

C. 程序段中的表达式是非法的 D. 程序段执行无限次

4 . 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int y=9;

for( ; y>0;y--)

if(y%3==0) printf("%d",--y); }

A . 741 B . 963 C . 852 D . 875421

在循环中，能被 3 整除的数有 9,6,3 ，输出前自减 1 ，所以得 8,5,2 。

5. 以下描述中正确的是 \_\_\_\_\_\_ 。

A . 由于 do-while 的循环体只能有一条语句，所以循环体内不能使用复合语句

B . do-while 由 do 开始，用 while 结束，在 while( 表达式 ) 后面不能写分号

C . 在 do-while 循环中，是先执行一次循环体，再进行判断

D . 在 do-while 循环中，根据情况可以省略 while

6. 以下程序段的运行结果是 \_\_\_\_\_\_ 。

i=0; do printf("%d,",i); while(i++); printf("%d\n",i);

A . 0,0 B . 0,1 C . 1,1 D . 程序进入无限循环

7. 若程序执行时的输入数据是 2473 < 回车 > ，则下述程序的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int cs;

while((cs=getchar())!='\n')

{switch(cs-'2')

{ case 0:

case 1: putchar(cs+4);

case 2: putchar(cs+4); break;

case 3: putchar(cs+3);

default: putchar(cs+2); }

}

}

A. 668977 B. 668966 C. 6677877 D. 6688766

8. 以下程序的输出结果是 \_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int a,i;a=0;

for(i=1;i<5;i++)

switch(i)

{ case 0: case 3:a+=2;

case 1: case 2:a+=3;

default:a+=5 ; }

printf("%d\n",a); }

A . 31 B . 13 C . 10 D . 20

9. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int i,j,m=55;

for(i=1;i<=3;i++)

for(j=3;j<=i;j++) m=m%j;

printf("%d\n",m); }

A .0 B . 1 C . 2 D . 3

10 . 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int i;

for(i='a';i<'f';i++,i++)printf("%c",i-'a'+'A');

printf("\n"); }

A. ACE B.BDF C.ABD D.CDE

11 . 以下程序运行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int k=5,n=0;

do {switch(k)

{ case 1: case 3:n+=1;k--;break;

default:n=0;k--;

case 2: case 4:n+=2;k--;break; }

printf("%d",n);

} while(k>0 && n<5); }

A . 235 B . 0235 C . 02356 D . 2356

12 . 下面程序的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int n=9; while(n>6){n--; printf("%d",n);} }

A . 987 B . 876 C . 8765 D . 9876

13. 下述程序的输出结果是 \_\_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int x=0,y=0,i;

for(i=1;;++i)

{ if(i%2==0) {x++;continue;}

if(i%5==0) {y++;break;}

}

printf (" %d,%d",x,y); }

A . 2,1 B . 2,2 C . 2,5 D . 5,2

14 . 下述程序的输出结果是 \_\_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int x=8;

for( ; x>0; x--)

{ if(x%3) {printf("%d,",x--); continue; }

printf("%d,",--x); } }

A . 7,4,2, B . 8,7,5,2, C . 9,7,6,4, D . 8,5,4,2,

15 . 以下不会构成无限循环的语句或语句组是 \_\_\_\_\_\_\_ 。

A . n=0; do{++n;}while(n<=0); B . n=0; while(1){n++;}

C . n=10; while(n); { n--;} D . for(n=0,i=1; ;i++) n+=i;

16. 以下程序的运行结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int k=0,m=0,i,j;

for(i=0;i<2;i++)

{ for(j=0;j<3;j++) k++; k-=j; }

m=i+j;

printf("k=%d,m=%d",k,m); }

A . k=0,m=3 B . k=0,m=5 C . k=1,m=3 D . k=1,m=5

17 . 若 int i,j; ，则 for(i=j=0;i<10&&j<8;i++,j+=3) 控制的循环体的执行次数是 \_\_\_\_\_\_\_ 。

A.9 B.8 C. 3 D.2

18 . 下述程序的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int i=6;

while(i--) printf("%d",--i);

printf("\n"); }

A.531 B. 420 C.654321 D . 死循环

19 . 下述程序的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int a=0,b=0,c=0,i;

for(i=0;i<4;i++)

switch(i)

{ case 0:a=i++;

case 1:b=i++;

case 2:c=i++;

case 3:i++; }

printf("%d,%d,%d,%d\n",a,b,c,i);}

A.0,1,3,4 B.1,2,3,4 C. 0,1,2,5 D.0,2,3,4

20 . 下面程序的运行结果是 \_\_\_\_\_\_\_ 。

a=1;b=2; c=2; while(a<b<c) { t=a;a=b;b=t;c--;}

printf("%d,%d,%d",a,b,c);

A . 1,2,0 B . 2,1,0 C . 1,2,1 D . 2,1,1

21 . 以下叙述中正确的是 \_\_\_\_\_\_\_ 。

　　 A.break 语句只能用于 switch 语句体中

　　 B.continue 语句的作用是使程序的执行流程跳出包含它的所有循环

　　 C. break 语句只能用在循环体内和 switch 语句体内

　　 D. 在循环体内使用 break 语句和 continue 语句的作用相同

22 . 以下程序运行后的输出结果是 \_\_\_\_\_\_\_ 。

　 #include "stdio.h"

void main()

{ int k=5,n=0;

do { switch(k)

{ case 1: case 3: n+=1; break;

default: n=0;k--;

case 2: case 4: n+=2;k--;break; }

printf("%d", n);}while(k>0&&n<5); }

A. 2345 　　 B.0235 　　 C.02356 　　 D.2356

23 . 下列程序的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int i,j;

for(i=1;i<4;i++)

{ for(j= i; j<4;j++) printf("%d\*%d=%d ",i,j,i\*j);

printf("\n");} }

　 A.1\*1=1 1\*2=2 1\*3=3 B. 1\*1=1 1\*2=2 1\*3=3

2\*1=2 2\*2=4 2\*2=4 2\*3=6

3\*1=3 3\*3=9

C . 1\*1=1 D . 1\*1=1

1\*2=2 2\*2=4 2\*1=2 2\*2=4

1\*3=3 2\*3=6 3\*3=9 3\*1=3 3\*2=6 3\*3=9

24 . 执行以下程序时 输入 1234567890< 回车 > ，则其中 while 循环体将执行 \_\_\_\_\_\_\_ 次。

　　 #include "stdio.h"

void main()

{ char ch; while((ch=getchar())=='0') printf("#"); }

A . 10 B . 0 C . 2 D . l

25 . 下列程序的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int k=5;

while(-k) printf("%d",k-=3); printf("\n"); }

A.1 B.2 C.4 D. 死循环

26. 以下程序执行后的输出结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int i;

for(i=1;i<=40;i++)

{ if(i++%5==0)

if(++i%8==0) printf("%d",i); }

printf("\n"); }

A . 5 B . 24 C . 32 D . 40

27 . 有以下程序，若运行时从键盘输入： 18 , 11< 回车 > ，则程序输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int a,b;

printf("Enter a,b:");scanf("%d,%d",&a,&b);

while(a!=b)

{ while(a>b) a-=b;

while(b>a) b-=a; }

printf("%3d%3d\n",a,b); }

A . 1 1 B . 1 2 C . 1 3 D . 1 4

28 . 要求通过 while 循环不断读入字符 ， 当读入字母 N 时结束循环。若变量已正确定义，以下正确的程序段是 \_\_\_\_\_\_\_ 。

A. while((ch=getchar())!= ' N ' )printf("%c",ch) ；

B.while(ch=getchar()!= ' N ' )printf("%c",ch) ；

C.while(ch=getchar()== ' N ' )printf("%c",ch) ；

D.while((ch=getchar())== ' N ' )printf("%c",ch) ；

29 . 以下程序运行后的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int y=10; while(y--);printf("y=%d\n",y); }

A . y=0 B . y= -1 C . y=1 D . while 构成无限循环

30 . 有以下程序段 ：

int n,t=1,s=0; scanf("%d",&n);

do{ s=s+t; t=t-2; }while (t!=n);

为使此程序段不陷入死循环，从键盘输入的数据应该是 \_\_\_\_\_\_\_ 。

A. 任意正奇数 B. 任意负偶数 C. 任意正偶数 D. 任意负奇数

31 . 以下程序运行后的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int i,j,x=0;

for(i=0;i<2;i++)

{ x++;

for(j=0;j<=3;j++){ if(j%2)continue;x++; }

x++;

}

printf("x=%d\n",x); }

A . x=4 B . x=8 C . x=6 D . x=12

32 . 设变量已正确定义 ，则以下能正确计算 f=n! 的程序段是 \_\_\_\_\_\_\_ 。

A . f=0; B . f=1;

for(i=1;i<=n;i++) for(i=1;i<=n;i--)

f\*=i; f\*=i;

C . f=1; D . f=1;

for(i=n;i>1;i++) for(i=n;i>=2;i--)

f\*=i; f\*=i;

33 . 以下程序运行后的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int k=5,n=0;

while(k>0)

{ switch(k)

{ default : break;

case 1 : n+=k;

case 2 :

case 3 : n+=k; }

k--; }

printf("%d\n",n); }

A.0 B.4 C.6 D. 7

34 . 以下程序的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int a=1,b;

for(b=1;b<=10;b++)

{ if(a>=8)break;

if(a%2==1){a+=5;continue; }

a-=3; }

printf("%d\n",b); }

A . 3 B . 4 C . 5 D . 6

35 . 以下程序输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{int i;

for(i=0;i<3;i++)

switch(i)

{ case 0:printf("%d",i);

case 2:printf("%d",i);

default:printf("%d",i); } }

A.022111 B.021021 C. 000122 D.012

36 . 以下程序输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int i=0,s=0;

for( ; ; )

{ if(i==3||i==5) continue;

if(i==6) break;

i++; s+=i; } ;

printf("%d\n",s); }

A.10 B.13 C.21 D. 程序进入死循环

37 . 若变量已正确定义 ，要求程序段完成求 5! 的计算，不能完成此操作的程序段是 \_\_\_\_\_\_\_ 。

A . for(i=1,p=1;i<=5;i++) p\*=i; B . for(i=1;i<=5;i++){ p=1;p\*=i;}

C . i=1;p=1;while(i<=5){p\*=i; i++;} D . i=1;p=1;do{p\*=i;i++;}while(i<=5);

38 . 以下程序 执行后的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int x=0,y=5,z=3;

while(z-->0&&++x<5) y=y-1;

printf("%d,%d,%d\n",x,y,z); }

A . 3,2,0 B . 3,2,-1 C . 4,3,-1 D . 5,-2,-5

39 . 以下程序 执行后的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

void main()

{ int i,n=0;

for(i=2;i<5;i++)

{ do{ if(i%3) continue; n++; } while(!i);

n++; }

printf("n=%d\n",n); }

A . n=5 B . n=2 C . n=3 D . n=4

40 . 下面程序的功能是输出以下形式的金字塔图。在下划线处应填入的是 \_\_\_\_\_\_\_ 。

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

#include "stdio.h"

void main( )

{ int i,j;

for(i=1;i<=4;i++)

{ for(j=1;j<=4-i;j++) printf("");

for(j=1;j<=\_\_\_\_\_\_\_;j++) printf("\*");

printf("\n"); }

}

A . i B . 2\*i-1 C . 2\*i+1 D . i+2

四、习题答案

（一）、是非题

1 .B 2.B 3.A 4.A 5.A 6.A 7.A 8.A 9.A 10.B

11 .A 12.B 13.A 14.B 15.B 16.A 17.A 18.B 19.A 20.B

（二）、选择题

1 .B 2.B 3.B 4.C 5.C 6.B 7.A 8.A 9.B 10.A

11 .A 12.B 13.A 14.D 15.A 16.B 17.C 18.B 19.C 20.A

21 .C 22.A 23.B 24.B 25.D 26.C 27.A 28.A 29.B 30.D

31 .B 32.D 33.D 34.B 35.C 36.D 37.B 38.B 39.D 40.B

## 第 5 章 数组

二、习题

（一）、是非题

1 . 数组名代表数组的首地址。

A. 对 B. 错

2. 字符型数组可以用来存储字符串，因此可以用字符串初始化数组，例如： char str[]="C Programming"; 。

A. 对 B. 错

3. 函数 strcpy(str1,str2) 是将字符串 str2 连接到字符串 str1 之后。

A. 对 B. 错

4 . 一个数组可以存储不同类型的数据。

A. 对 B. 错

5 . 数组在定义时可以不指定数组的长度，可根据程序中实际使用的元素个数决定。

A. 对 B. 错

6 . 在 C 语言中，只有当两个字符串所包含的字符个数相同时，才能比较大小。例如，字符串“ That ”与“ The ”就不能进行大小比较。

A. 对 B. 错

7 . 若有定义： int a[10]; ，使用数组元素时，下标的最小值是 1 ，最大值是 10 。

A. 对 B. 错

8 . 在 C 语言中引用数组元素时，其数组的下标可以是任何类型的表达式。

A. 对 B. 错

9 . 任何一维数组的名称都是该数组存储单元的开始地址，且其每个元素按照顺序连续占用存储空间。

A. 对 B. 错

10 . 一维数组的元素实质上就是一个变量，代表内存中的一个存储单元。

A. 对 B. 错

1 1 . 若有说明： int a[3][4]={0}; ，则只有元素 a[0][0] 可得到初值 0 。

A. 对 B. 错

12 . 若有说明： int a[][3]={1,2,3,4,5,6,7}; ，则 a 数组第一维的大小不能确定。

A. 对 B. 错

13 . 若一维数组在定义时进行了初始化，则可以省略数组长度，系统会根据初始值个数确定该数组长度。

A. 对 B. 错

14 . 数组初始化时，若初始值的个数少于数组定义的长度， C 语言会自动将剩余的元素初始化为初始化列表中最后一个初始值。

A. 对 B. 错

15 . 可以对字符型数组进行整体输入、输出。

A. 对 B. 错

16. 可以对实型数组进行整体输入、输出。

A. 对 B. 错

17. 设有 定义： double a[4][5]; ，则数组 a 中元素的个数是 9 。

A. 对 B. 错

18. 设有 定义： int a[3][4]={{1,2,5,3},{2,4,7,9},{3,6,5,8}}; ，则表达式 a[1][2]+a[2][3] 的值 9 。

A. 对 B. 错

19. 设有 定义： char s[]={"E\_mail"}; ，则执行 printf("%d\n",strlen(s)); 后输出的值是 6 。

A. 对 B. 错

20. 设有两个整型数组 a 和 b ，可以通过 a=b ，将数组 b 整体赋值给数组 a 。

A. 对 B. 错

（二）、选择题

1 . 设有两字符串“ Good ”、“ Morning ”分别存放在字符数组 str1[10] 、 str2[10] 中，下面语句中能把“ Morning ”连接到“ Good ”之后的是 \_\_\_\_\_\_ 。

A.strcpy(str1,str2); B.strcpy(str1, " Morning " );

C. strcat(str1, " Morning " ); D.strcat( " Good " ,str2);

2 . 以下对一维整型数组 a 定义正确的是 \_\_\_\_\_\_ 。

A.int a(10); B.int n=10,a[n];

C.int n;scanf( " %d " ,&n); int a[n]; D. #define SIZE 10 int a[SIZE];

3. 下列字符串赋值语句中，不能正确把字符串“ Hello World ”赋给数组的语句是 \_\_\_\_\_\_ 。

A.char a1[]={'H','e ','l','l','o',' ','W','o','r','l','d',' \0 '};

B.char a2[15]; strcpy(a2, " Hello World " );

C. char a3[15]; a3= " Hello World " ;

D.char a4[10]={ " Hello World " } ;

4 . 用 scanf 函数输入一个字符串到数组 str 中，下面正确的语句是 \_\_\_\_\_\_ 。

A.scanf( " %s " ,&str); B.scanf( " %c " ,&str[10]);

C.scanf( " %s " ,str[10]); D. scanf( " %s " ,str) ;

5 . 以下能对二维数组 a 进行正确说明和初始化的语句是 \_\_\_\_\_\_ 。

A.int a()(3)={ (1,2 , 3),(2,4,6)}; B.int a[2][]={{3,2,1},{5,6,7}};

C. int a[][3]={{3,2,1},{5,6,7}}; D. int a(2)()={(1,2 , 3),(2,4,6)};

6 . 判断字符串 a 和 b 是否相等，应当使用 \_\_\_\_\_\_ 。

A.if(a==b) B.if(a=b)

C.if(strcat(a,b)) D. if(strcmp(a,b))

7 . 有字符数组 a[80] 和 b[80] ，则正确的输出语句是 \_\_\_\_\_\_ 。

A.puts(a,b); B. puts(a);puts(b);

C.printf( " %s,%s " ,a[],b[]); D.putchar(a,b);

8 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a[5],i;

for(i= 0 ;i<5;i++)

{ a[i]=9\*(i-2+4\*(i>3))%5;

printf("%2d",a[i]); } }

A. - 3-4 0 4 4 B.-3-4 0 4 3 C.-3-4 0 4 2 D.-3-4 0 4 0

9 . 假设 array 是一个有 10 个元素的整型数组， 则下列写法中正确的是 \_\_\_\_\_\_ 。

A. array[0]=10; B.array=0; C.array[10]=0; D.array[-1]=0;

10 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a[3],i,j,k;

for(i=0;i<3;i++) a[i]=0;

k=2;

for(i=0;i<k;i++)

for(j=0;j<k;j++)

a[j]=a[i]+1;

printf("%d\n",a[1]); }

A.0 B.1 C.2 D. 3

11 . 若有定义： int a[5]={1 , 2,3,4 , 5 } ;char b='c',c; ，则下面表达式中数值为 2 的是 \_\_\_\_\_\_ 。

A.a[2] B.a[c-b] C.a[b] D. a[b-'b']

12 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a[3][3]={1,2,3,4,5,6,7,8,9},i;

for(i=0;i<=2;i++) printf("%d ",a[i][2-i]); }

A. 3 5 7 B.3 6 9 C.1 5 9 D.1 4 7

13 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char c[5]={'a','b','\0','c','\0'};

printf("%s",c); }

A.'a' 'b' B. ab C.ab c D.ab\0c\0

14 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char s[12]= "a book!";

printf("%d",strlen(s)); }

A.6 B. 7 C.10 D.12

15 . 若有说明： int a[3][4]; 则对 a 数组元素的正确引用是 \_\_\_\_\_\_ 。

A.a[2][4] B.a[1,3] C. a[1+1][0] D.a(2)(1)

16 . 以下各组选项中，均能正确定义二维实型数组 a 的选项是 \_\_\_\_\_\_ 。

A.float a[3][4]; float a[][4]; float a[3][]={{1},{0}};

B.float a(3,4); float a[3][4]; float a[][]={{0};{0}};

C. float a[3][4]; static float a[][4]={{0},{0}};

D.float a[3][4]; float a[3][];

17 . 若二维数组 a 有 m 列，则计算任一元素 a[i][j] 在数组中位置的公式是 \_\_\_\_\_\_ 。（假设 a[0][0] 位于数组的第一个位置上。）

A.i\*m+j B.j\*m+i C.i\*m+j-1 D. i\*m+j+1

18 . 以下对 C 语言字符数组描述错误的是 \_\_\_\_\_\_ 。

A. 字符数组可以存放字符串

B. 字符数组中的字符串可以整体输入或输出

C. 可以在赋值语句中通过赋值运算符” = ”对字符数组整体赋值

D. 不可以用关系运算符对字符数组中的字符串进行比较

19 . 以下合法的数组定义是 \_\_\_\_\_\_ 。

A.int a[]= " language " ; B.int a[5]={0,1,2,3,4,5};

C.char a= " string " ; D .char a[]={ " 0,1,2,3,4,5 " } ;

20 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

#include <string.h>

main()

{ char a[7]="abcdef",b[4]="ABC";

strcpy(a,b);

printf("%c",a[5]); }

A. 空格 B. \ 0 C. f D. 不确定

21 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

#include <string.h>

main()

{ char a[30];

strcpy(&a[0],"ch");

strcpy(&a[1],"def");

strcpy(&a[2],"abc");

printf("%s\n",a);

system("pause"); }

A.chdefabc B.cda C. cdabc D.abcdef

22 . 设有定义： char a[80];int i=0; ，以下不能将一行带有空格的字符串（不超过 80 个字符）正确读入的是 \_\_\_\_\_\_ 。

A.gets(a);

B.while(( a[i++]=getchar())!=' \ n'); a[i]=' \0 ';

C. scanf( " %s " ,a);

D.do{scanf( " %c " ,&a[i]); }while(a[i++]!=' \ n'); a[i]=' \0 ';

23 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char str[]="SSSWLIA",c;

int k;

for(k=2;(c=str[k])!='\0';k++)

{ switch(c)

{ case 'I':++k;break;

case 'L':continue;

default:putchar(c);continue;

}

putchar('\*');

}

}

A.SSW\* B. SW\* C.SW\*A D.SWA\*

24 . 设 char s[10]= "abcd",t[]="12345"; ，则 s 和 t 在内存中分配的字节数分别是 \_\_\_\_\_\_ 。

A. 6 和 5 B. 6 和 6 C. 10 和 5 D . 10 和 6

25 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char a[]="Hello World";

int i,j;

for(i=j=0; a[i]!='\0';i++)

if(a[i]!='l') a[j++]=a[i];

a[j]='\0';

puts(a); }

A.Hello World B.Heo World

C. Heo Word D. 没有任何输出内容

26 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char str1[]="abcd",str2[]="abcef";

int i,s;

i=0;

while((str1[i]==str2[i])&&(str1[i]!='\0')) i++;

s=str1[i]-str2[i];

printf("%d\n",s); }

A. -1 B.0 C.1 D. 不确定

27 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include

main()

{ char s[]="012xy";

int i,n=0;

for(i=0;s[i]!=0;i++)

if(s[i]>= 'a'&&s[i]<= 'z') n++;

printf("%d\n",n); }

A. 0 B. 2 C.3 D.5

28 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a[]={2,3,5,4},i;

for(i=0;i<4;i++)

switch(i%2)

{ case 0:switch(a[i]%2)

{ case 0:a[i]++;break;

case 1:a[i]--;}

break;

}

for(i=0;i<4;i++) printf("%2d",a[i]);

printf("\n"); }

A. 3 3 4 4 B.2 0 5 0 C.3 0 4 0 D.0 3 0 4

29 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

#include <string.h>

main()

{ char a[10]="abcd"; printf("%d,%d\n",strlen(a),sizeof(a)); }

A.7,4 B. 4,10 C.8,8 D.10,10

30 . 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

#define MAX 10

main()

{ int i,sum,a[]={1,2,3,4,5,6,7,8,9,10};

sum=1;

for(i=0;i<MAX;i++) sum-=a[i];

printf("%d",sum);

system("pause"); }

A.55 B. -54 C.-55 D.54

四、习题答案

（一）、是非题

1 .A 2.A 3.B 4.B 5.B 6.B 7.B 8.B 9.A 10.A

11 .B 12.B 13.A 14.B 15.A 16.B 17.B 18.B 19.A 20.B

（二）、选择题

1 .C 2.D 3.C 4.D 5.C 6.D 7.B 8.A 9.A 10.D

11 .D 12.A 13.B 14.B 15.C 16.C 17.D 18.C 19.d 20.C

21 .C 22.C 23.B 24.D 25.C 26.A 27.B 28.A 29.B 30.B

## 第 6 章 函数

二、习题

（一）是非题

1. 在 C 程序中， main() 函数必须有且只有一个。

A. 对 B. 错

2. 如果某个函数在定义时省略了存储类型，则默认的存储类型是 int 。

A. 对 B. 错

3.C 程序中，定义函数时的参数是“实际参数”，调用函数时的参数是“形式参数”。

A. 对 B. 错

4. 函数的定义可以嵌套，但函数的调用不可以嵌套。

A. 对 B. 错

5. 若函数定义的头部为： double cacl(float x,float y) ，则该函数是外部函数。

A. 对 B. 错

6. 若用数组名作为函数调用的实参，传递给形参的是数组中的第一个元素的值。

A. 对 B. 错

7.C 程序中所有的全局变量都是静态存储的。

A. 对 B. 错

8. 函数中的多个 return 语句，可以实现调用该函数后，多个值同时被返回。

A. 对 B. 错

9. 如果在一个函数中的复合语句内定义了一个变量，则该变量在该函数中都有效。

A. 对 B . 错

10. 有参的宏，其参数参与运算，宏替换要先完成参数的计算，然后再执行替换。

A. 对 B. 错

11. 在函数之间数据传递的方式有三种，即返回值、参数、全局变量。

A. 对 B. 错

12. 在 C 语言中，除了 main() 函数外，其余 所有的 函数都能够被调用。

A. 对 B. 错

13. 在 main() 函数中定义的变量称为全局变量。

A. 对 B. 错

14. 在 C 语言中形参与实参共用内存单元。

A. 对 B. 错

15. 在 C 语言中定义的 auto 变量，若没有对其进行显式初始化，则初值是一个不确定的值。

A. 对 B. 错

（二）选择题

1. 以下说法中正确的是 \_\_\_\_\_\_\_\_ 。

A. C 语言程序总是从第一个定义的函数开始执行

B. 在 C 语言程序中，要调用的函数必须在 main() 函数中定义

C. C 语言程序总是从 main() 函数开始执行

D. C 语言程序中的 main() 函数必须放在程序的开始部分

2. 在 C 语言中，函数值类型在定义时可以缺省，此时函数值的隐含类型是 \_\_\_\_\_\_\_\_ 。

A. void B. int C.float D.double

3. 在 C 语言中，函数返回值的类型最终取决于 \_\_\_\_\_\_\_\_ 。

A. 函数定义时在函数值首部所说明的函数类型

B.return 语句中表达式值的类型

C. 用函数时主调函数所传递的实参类型

D. 函 数定义时形参的类型

4. 以下程序的执行结果为 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

float fun(int x,int y)

{ return(x\*y); }

main()

{ int a=2,b=5,c=8;

printf(" %.0f\n " ,fun((int)fun(a+b,c),a-b)); }

A.- 24 B.24 C. -168 D.168

5. 以下程序运行后的输出结果是 \_\_\_\_\_\_\_\_ 。

#include " stdio.h "

int f(int x);

main()

{ int n=1,m;

m=f(f(f(n)));

printf( " %d\n " ,m); }

int f(int x)

{return x\*2;}

A.1 B.2 C.4 D. 8

6. 如下函数调用语句中，含有的实参个数是 \_\_\_\_\_\_\_\_ 。

fun(arg1,arg2+arg3,(arg4,arg5)); // 实参可以是表达式

A. 3 B.4 C.5 D. 有语法错误

7. 以下程序的输出结果是 \_\_\_\_\_\_\_\_ 。

#include " stdio.h "

main()

{ int w=5;fun(w); }

fun(int m)

{ if(m>0)fun(m-1);

printf( " %d " ,m) ;}

A.5 4 3 2 1 B. 0 1 2 3 4 5 C.1 2 3 4 5 D.5 4 3 2 1 0

8. 在一个 C 源程序文件中所定义的全局变量，其作用域为 \_\_\_\_\_\_\_\_ 。

A. 所在文件的全部范围 B. 所在程序的全部范围

C. 所在函数的全部范围 D. 由具体定义位置和 extern 说明来决定范围

9. 以下程序的运行结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

int a=1;

int f(int c)

{ static int a=2;

c=c+1;

return (a++)+c; }

main()

{ int i,k=0;

for(i=0;i<2;i++)

{ int a=3;

k+=f(a); }

k+=a;

printf("%d\n",k); }

A.13 B. 14 C.15 D.16

10. 以下程序执行后变量 w 的值是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

int fun1(double a)

{ return a\*=a;}

int fun2(double x,double y)

{ double a=0,b=0;

a=fun1(x);b=fun1(y);return(int)(a+b); }

main()

{ double w;

w=fun2(1.1,2.0);

…… }

A.5.21 B.5 C. 5.0 D.0.0

11. 以下程序执行时，给变量 x 输入 10 ，程序的输出结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

int fun(int n)

{ if(n==1) return 1;

else return(n+fun(n-1)); }

main()

{ int x;

scanf("%d",&x);

x=fun(x);

printf("%d\n",x); }

A.45 B.54 C. 55 D. 65

12. 以下叙述中错误的是 \_\_\_\_\_\_\_\_ 。

A.C 程序必须由一个或一个以上的函数组成

B. 函数调用可以作为一个独立的语句存在

C. 若函数有返回值，必须通过 return 语句返回

D. 函数形参的值也可以传回给对应的实参

13. 以下叙述中正确的是 \_\_\_\_\_\_\_\_ 。

A. 预处理命令行必须位于 C 源程序的起始位置

B. 在 C 语言中，预处理命令行都以 "#" 开头

C. 每个 C 程序必须在开头包含预处理命令行： #include " stdio.h "

D.C 语言的预处理不能实现宏定义和条件编译的功能

14. 以下叙述中正确的是 \_\_\_\_\_\_\_\_ 。

A. 局部变量说明为 static ，其生存期将得到延长

B. 全局变量说明为 static ，其作用域将被扩大

C. 任何存储类的变量在未赋初值时，其值都是不确定的

D. 形参可以使用的存储类说明符与局部变量完全相同

15. 在函数调用过程中，如果函数 funA 调用了函数 funB ，函数 funB 又调用了函数 funA ，则 \_\_\_\_\_\_\_\_ 。

A. 称为函数的直接递归调用 B.C 语言中不允许这样的递归调用

C. 称为函数的循环调用 D. 称为函数的间接递归调用

16. 以下程序的输出结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

long fib(int n)

{ if(n>2) return(fib(n-1)+fib(n-2));

else return(2); }

void main()

{ printf("%d\n",fib(3)); }

A . 2 B . 4 C . 6 D . 8

17. 以下所列的各函数首部中，正确的是 \_\_\_\_\_\_\_\_ 。

A . void play(var:Integer,var b:Integer) B . void play(int a,b)

C . void play(int a,int b) D . Sub play(a as integer,b as integer)

18. 以下程序的输出结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

#define f(x) x\*x

void main()

{ int a=6,b=2,c;

c=f(a)/f(b);

printf("%d\n",c); }

A.9 B.18 C. 36 D.6

19. 下列程序的运行结果为 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

#define MA(x) x\*(x-1)

void main()

{ int a=1,b=2;

printf("%d\n",MA(1+a+b)); }

A.5 B.6 C.7 D. 8

20. 以下程序的输出结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

f(int b[],int m,int n)

{ int i,s=0;

for(i=m;i<n;i=i+2)

s=s+b[i];

return s; }

main()

{ int x,a[]={1,2,3,4,5,6,7,8,9};

x=f(a,3,7);

printf("%d\n",x); }

A . 10 B . 18 C . 8 D . 15

21. 下列的结论中只有 \_\_\_\_\_\_\_\_ 是正确的。

A. 所有的递归程序均可以采用非递归算法实现

B. 只有部分递归程序可以用非递归算法实现

C. 所有的递归程序均不可以采用非递归算法实现

D. 以上三种说法都不对

22. 以下程序的运行结果为 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

int x=2;

int fun(int p)

{ static int x=3;

x+=p;

printf("%d ",x);

return (x); }

void main()

{ int y=3; printf("%d \n",fun(fun(x))); }

A.5 10 20 B.5 6 10 C. 5 10 10 D.5 5 10

23. 下列程序的输出结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

int fun(int x,int y,int cp,int dp)

{ cp=x\*x+y\*y; dp=x\*x-y\*y; }

void main()

{ int a=4,b=3,c=5,d=6;

fun(a,b,c,d);

printf("%d %d\n",c,d); }

A . 16 9 B . 4 3 C.25 9 D. 5 6

24. 以下程序输出的结果是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

long fun(int n)

{ long s;

if(n==1||n==2) s=2;

else s=n-fun(n-1);

return s; }

main()

{ printf("%ld\n",fun(3));}

A . 1 B . 2 C . 3 D . 4

25. 阅读下面的程序：

#include "stdio.h"

void main()

{ int swap();

int a,b;

a=3;b=10;

swap(a,b);

printf("a=%d,b=%d\n",a,b); }

swap(int a,int b)

{ int temp;

temp=a; a=b; b=temp; }

下面的说法中，正确的是 \_\_\_\_\_\_\_\_ 。

A. 在 main() 函数中调用 swap() 后，能使变量 a 和 b 的值交换

B. 在 main() 函数中输出的结果是： a=3,b=10

C. 程序第 2 行的语句 int swap(); 是对 swap() 函数进行调用

D.swap() 函数的类型是 void

四、习题答案

（一）、是非题

1 .A 2.A 3.B 4.B 5.A 6.B 7.A 8.B 9.B 10.B

11 .B 12.B 13.B 14.B 15.A

（二）、选择题

1 .C 2.B 3.A 4.C 5.D 6.A 7.B 8.D 9.B 10.C

11 .C 12.D 13.B 14.A 15.D 16.B 17.C 18.C 19.B 20.A

21 .B 22.C 23.D 24.A 25.B

第 7 章 结构体

二、习题

（一）、是非题

1. 结构体变量所占用空间是全部成员占用空间之和。

  A. 对 B. 错

2. 共用体可以作为结构体的成员，但结构体不能作为共用体的成员。

  A. 对 B. 错

3. 结构体中各个成员的数据类型必须是相同的。

  A. 对 B. 错

4. 有以下定义的结构体：

typedef struct student

{ int num; char name[10]; int age; } stu1;

则， struct student 是结构体变量名。

  A. 对 B. 错

5. 枚举类型中，枚举值是常量，不能再在程序中对其赋值。

  A. 对 B. 错

6.typedef 的作用是给数据类型取一个别名。

  A. 对 B. 错

7. 如果两个结构体变量有相同的类型，则可以把其中一个结构体变量的值整体赋值给另一个结构体变量。

  A. 对 B. 错

8. 一个共用体变量可以有多个数据成员，这些数据成员共享同一个存储空间，但如果空间不够用可以临时由系统分配。

  A. 对 B. 错

9. 结构体变量可以整体赋值，但不能整体比较。

  A. 对 B. 错

10. 在定义了枚举变量后，系统才为该变量分配内存空间。

  A. 对 B. 错

（二）、选择题

1. 在定义一个共用体变量时，系统分配给它的内存是 \_\_\_\_\_\_ 。

A. 各成员所需内存之和 B. 第一个成员所需内存

C. 成员中占用内存最大者 D. 任意一个成员所需内存

2. 以下结构体的定义语句中，正确的是 \_\_\_\_\_\_ 。

A . struct student {int num; char name[10];int age;} ; stu;

B . struct {int num; char name[10];int age;} student ; struct student stu;

C. struct student {int num; char name[10];int age;}stu;

D.struct student {int num; char name[10]; int age;}; student stu;

3. 如有定义的枚举类型：

enum week\_day{Wed=3,Thu,Fri,Sat,Sun,Mon,Tue}everyday;

则， Mon 的值为 \_\_\_\_\_\_ 。

A.1 B. 2 C. 8 D. 5

4. 有如下定义的结构体类型：

struct data

{ int year; int month; int day; }workday;

对其中成员 month 的正确引用方式是 \_\_\_\_\_\_ 。

A.data.month B.data.year.month C.month D. workday.month

5. 已知对学生记录的描述为：

struct student

{ int num;

char name[20],sex;

struct{ int year,month,day; }birthday;

} ;

struct student stu;

设变量 stu 中的“生日”是“ 1995 年 11 月 12 日”，对“ birthday ”正确赋值的程序是 \_\_\_\_\_\_ 。

A . year=1995;month=11;day=12;

B . stu.year=1995;stu.month=11;stu.day=12;

C . birthday.year=1995;birthday.month=11;birthday.day=12;

D . stu.birthday.year=1995;stu.birthday.month=11;stu.birthday.day=12;

6. 下列关于枚举类型的描述不正确的是 \_\_\_\_\_\_ 。

A. 可以在定义枚举类型时对枚举元素进行初始化

B. 在赋值时，不可以将一个整数赋值给枚举变量

C. 枚举变量不可以进行关系运算

D. 枚举变量只能取对应枚举类型的枚举元素表中的元素值

7. 有如下的说明：

union test

{ int a; char c; }test1;

则在 VC 环境下 sizeof(union test) 的结果是 \_\_\_\_\_\_ 。

A. 4 B.2 C.5 D.3

8. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

union con

{ struct {int x,y,z;}m;

int i; } num;

main()

{ num.m.x=4;num.m.y=5;num.m.z=6;num.i=0;

printf("%d\n",num.m.x);

}

A.4 B. 0 C.5 D.6

9. 有如下程序段，执行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ structa{int x;int y;}num[2]={{20,5},{6,7}};

printf("%d\n",num[0].x/num[0].y\*num[1].y); }

A.0 B. 28  C.20  D.5

10. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

#include <string.h>

typedef struct {char name[9]; char sex; float score[2];} STU;

void f(STU a)

{ STU b={"Zhao",'m',85.0,90.0}; int i;

strcpy(a.name, b.name);

a.sex=b.sex;

for(i=0;i<2;i++) a.score[i]=b.score[i];

}

main()

{ STU c={"Qian",'f',95.0,92.0};

f(c);

printf("%s,%c,%2.0f,%2.0f\n",c.name,c.sex,c.score[0],c.score[1]); }

A. Qian,f,95,92 B.Qian,m,85,90

C.Zhao,f,95,92 D.Zhao,m,85,90

四、习题答案

（一）、是非题

1 .A 2.B 3.B 4.B 5.A 6.A 7.A 8.B 9.B 10.A

（二）、选择题

1 .C 2.C 3.C 4.D 5.D 6.C 7.A 8.B 9.B 10.A

## 第 8 章 指针

二、习题

（一）、是非题

1. 运算符 “ \* ”有时是单目运算符，有时又可能是双目运算符。

A. 对 B. 错

2. 取地址运算符“ & ” 与指针运算符 “ \* ”的优先级相同，结合方向为自右向左。

A. 对 B. 错

3. 指针变量的类型不是该变量本身的类型，而是它所指向的变量的类型。

A. 对 B. 错

4. 指针变量不可以作为函数的参数。

A. 对 B. 错

5 . &b 指的是变量 b 的地址所存放的值。

A. 对 B. 错

6 . 语句 int \*p; 定义了一个指针变量 p ，其值是整型的。

A. 对 B. 错

7 . 语句“ int \*p ; ”中 \*p 的含义为取值。 // 定义指针变量

A. 对 B. 错

8 . 一个变量的地址就称为该变量的“指针 ” 。

A. 对 B. 错

9 . 语句 int i,\*p=&i; 是正确的 C 定义。

A. 对 B. 错

1 0. 变量的指针，其含义是指该变量的一个标志。

A. 对 B. 错

11 . void 类型的指针不指向某种类型的数据，所以没有意义。

A. 对 B. 错

12 . 表达式 \*p++ 的运算顺序为先自增后取值。

A. 对 B. 错

13 . C 语言中，执行表达式 (\*p)++ 后 p 的指向改变了。

A. 对 B. 错

14 . 两个指针相加没有意义。

A. 对 B. 错

15 . 若有定义： int \*p1,\*p2,m=3,n; ，则 p1=&m;p2=p1; 是正确的赋值语句。

A. 对 B. 错

16 . 已知 int a[4],\*p; ，则语句 p=&a; 是正确的。 // p=a;

A. 对 B. 错

17 . char \*name[5] 定义了一个一维指针数组，它有 5 个元素，每个元素都是指向字符数据的指针型数据。

A. 对 B. 错

18 . \*(&a[i] ＋ j) 与 a[i][j] 的含义相同。

A. 对 B. 错

19 . 若有定义： int (\*p)[4]; ，则标示符 p 是一个指针，它指向一个含有四个整型元素的一维数组。

A. 对 B. 错

20 . 设有以下定义： char \*c[2]={ " 12 " , " 34 " } ; ，表示 c 数组的两个元素中分别存放了字符串 " 12 " 和 " 34 " 的首地址。

A. 对 B. 错

（二）、选择题

1. 变量的指针，其含义是指该变量的 \_\_\_\_\_\_ 。

A. 值 B. 地址 C. 名 D. 一个标志

2. 下面能正确进行字符串赋值操作的是 \_\_\_\_\_\_ 。

A.char s[5]={"ABCDE"}; B.char s[5]={'A' , 'B' , 'C' , 'D' , 'E' } ;

C. char \*s; s="ABCDE"; D.char \*s; scnaf("%s",&s);

3. 以下语句或语句组中，能正确进行字符串赋值的是 \_\_\_\_\_\_ 。。

A.char \*sp;\*sp="right"; B.char s[10];s="right";

C.char s[10];\*s="right"; D. char \*sp="right";

4. 设有定义： char \*c; ，以下选项中能够使字符型指针 c 正确指向一个字符串的是 \_\_\_\_\_\_ 。

A. char str[]= " string " ;c=str; B.scanf( " %s " ,c);

C.c=getchar() ; D.\*c= " string " ;

5. 下列语句中，正确的是 \_\_\_\_\_\_ 。

A. char \*s;s= " Olympic " ; B.char s[7];s= " Olympic " ;

C.char \*s;s={ " Olympic " } ; D.char s[7];s={ " Olympic " } ;

6. 已有定义 int k=2; int \*p1,\*p2; 且 p1 和 p2 均已指向变量 k ，下面不能正确执行的赋值语句是 \_\_\_\_\_\_ 。

A.k=\*p1+\*p2; B. p2=k; C.p1=p2; D.k=\*p1\*(\*p2);

7. 若有定义： double a[10],\*s=a; ，以下能够代表数组元素 a[3] 的是 \_\_\_\_\_\_ 。

A.(\*s)[3] B. \*(s+3) C.\*s[3] D.\*s+3

8. 若有定义 int a[5],\*p=a; ，则对 a 数组元素的正确引用是 \_\_\_\_\_\_ 。

A.\*&a[5] B.a+2 C.\*(p+5) D. \*(a+2)

9. 若有定义语句： double a,\*p=&a; 以下叙述中错误的是 \_\_\_\_\_\_ 。

A. 定义语句中 \* 是一个间接访问运算符

B. 定义语句中 \* 号只是一个说明符

C. 定义语句中 p 只能存放 double 类型变量的地址

D. 定义语句中， \*p=&a 把变量 a 的地址作为初始赋给指针变量 p

10. 若有程序段： char str[]="china";char \*p;p=str; 则下列叙述正确的是 \_\_\_\_\_\_ 。

A. \*p 与 str[0] 相等

B.str 数组长度和 p 所指向的字符串长度相等

C.str 和 p 完全相同

D. 数组 str 中的内容和指针变量 p 中的内容相等

11. 若有语句 int \*point,a=4; 和 point=&a; 下面均代表地址的一组选项是 \_\_\_\_\_\_ 。

A.a,point,\*&a B.&\*a,&a,\*point

C.&point,\*point,&a D. &a,&\*point,point

12. 以下选项中，对指针变量 p 的正确操作是 \_\_\_\_\_\_ 。

A.int a[5],\*p; p=&a; B. int a[5],\*p; p=a;

C.int a[5]; int \*p=a=1000; D.int a[5]; int \*p1,\*p2=a; \*p1=\*p2;

13. 设有定义语句 :int m[]={2,4,6,8 } ,\*k=m; ，以下选项中表达式的值为 6 的是 \_\_\_\_\_\_ 。

A. \*(k+2) B.k+2 C.\*k+2 D.\*k+=2

14. 若有 定义 ： int \*p,m=5,n; ，以下正确的程序段是 \_\_\_\_\_\_ 。

A.p=&n;scanf( " %d " ,&p); B.p=&n;scanf( " %d " ,\*p);

C.scanf( " %d " ,&n); \*p=n; D. p=&n;\*p=m;

15. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include "stdio.h"

main()

{int m=1,n=2,\*p=&m,\*q=&n,\*r;

r=p;p=q;q=r;

printf("%d,%d,%d,%d\n",m,n,\*p,\*q); }

A.1,2,1,2 B. 1,2,2,1 C.2,1,2,1 D.2,1,1,2

16. 若有定义语句： int a[4][10],\*p,\*q[4] ；且 0 ≤ i ＜ 4 ，则错误的赋值是

A. p=a B.q[i]=a[i] C.p=a[i] D.p=&a[2][1]

17. 以下程序中调用 scanf 函数给变量 a 输入数值的方法是错误的，其错误原因是 \_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ int \*p,q,a,b; p=&a; printf("input a: "); scnaf("%d",\*p); … }

A.\*p 表示的是指针变量 p 的地址

B. \*p 表示的是变量 a 的值，而不是变量 a 的地址

C.\*p 表示的是指针变量 p 的值

D.\*p 只能用来说明 p 是一个指针变量

18. 程序段 char \*s="abcde"; s+=2; printf("%d",s); 的运行结果是 \_\_\_\_\_\_ 。

A.cde B. 字符 'c'

C. 字符 'c' 的地址 D. 无确定的输出结果

19. 下面程序段的运行结果是 \_\_\_\_\_\_ 。

char str[]="ABC", \*p=str; printf("%d\n",\*(p+3)); // \*(p+3) 表示 p 的地址从开始处移动 3

A.67 B. 0 C. 字符 'C' 的地址 D. 字符 'C'

20. 下面程序段的运行结果是 \_\_\_\_\_\_ 。

char a[]="language", \*p; p=a;

while(\*p!='u') {printf("%c",\*p-32); p++; }

A.LANGUAGE B.language C. LANG D.langUAGE

21. 若有定义 char s[10]; ，则在下面表达式中不表示 s[1] 的地址的是 \_\_\_\_\_\_ 。

A.s+1 B. s++ C.&s[0]+1 D.&s[1]

22. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

#include <string.h>

main()

{ char str[][20]={"One\*World","One\*Dream!"},\*p=str[1];

printf("%d,",strlen(p));

printf("%s\n",p); }

A.9, One\*World B.9, One\*Dream! C. 10, One\*Dream! D.10, One\*World

23. 下列函数的功能是 \_\_\_\_\_\_ 。

fun(char \*a,char \*b)

{ while((\*b=\*a)!=' \0 ') {a++,b++;} }

A. 使 a 所指字符串赋给 b 所指空间

B. 使指针 b 指向 a 所指字符串

C. 将 a 所指字符串和 b 所指字符串进行比较

D. 检查 a 和 b 所指字符串中是否有 ' \0 '

24. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char \*s={"abc"};

do{ printf("%d",\*s%10);++s; }while(\*s); }

A.abc B. 789 C.7890 D.979899

25. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void fun(int \*p)

{ printf("%d\n",p[5]); }

main()

{ int a[10]={1,2,3,4,5,6,7,8,9,10};

fun(&a[3]); }

A.5 B.6 C.8 D. 9

26. 若有定义语句： char \*s= " OK " ,\*s2= " ok " ; ，以下选项中能够输出“ OK ”的语句是 \_\_\_\_\_\_ 。

A.if(strcmp(s1,s2)=0) puts(s1); B.if(strcmp(s1,s2)!=0) puts(s2);

C.if(strcmp(s1,s2)=1) puts(s1); D. if(strcmp(s1,s2)!=0) puts(s1);

27. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void fun(char \*c,int d)

{ \*c=\*c+1;d=d+1;

printf("%c,%c,",\*c,d); }

main()

{ char b='a',a='A';

fun(&b,a);

printf("%c,%c\n",b,a); }

A. b,B,b,A B.b,B,B,A C.a,B,B,a D.a,B,a,B

28. 有以下定义和语句：

struct workers

{ int num;char name[20];char c;

struct { int day;int month;int year;}s; };

struct workers w,\*pw;

pw=&w;

能给 w 中 year 成员赋 1980 的语句是 \_\_\_\_\_\_ 。

A.\*pw.year=1980; B.w.year=1980 C.pw—>year=1980; D. w.s.year=1980;

29. 下面程序把数组元素中的最大值放入 a[0] 中，则在 if 语句中的条件表达式应该是 \_\_\_\_\_\_ 。

main()

{ int a[10]={6,7,2,9,1,10,5,8,4,3},\*p=a,i;

for(i=o;i<10;i++,p++) if(\_\_\_\_\_\_\_\_) \*a=\*p; printf("%d",\*a); }

A.p>a B. \*p>a[0] C.\*p>\*a[0] D.\*p[0]>\*a[0]

30. 以下程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ int a[]={10,20,30,40},\*p=a,i;

for(i=0;i<=3;i++) {a[i]=\*p;p++;}

printf("%d\n",a[2]); }

A. 30 B.40 C.10 D.10

31. 以下程序的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char \*s="12134";

int k=0,a=0;

while(s[k+1]!='\0')

{ k++;

if(k%2==0) {a=a+(s[k]-'0'+1);continue;}

a=a+(s[k]-'0'); }

printf("k=%d a=%d\n",k,a); }

A.k=6 a=11 B.k=3 a=14 C. k=4 a=12 D.k=5 a=15

32. 以下程序的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char a[5][10]={"one","two","three","four","five"};

int i,j; char t;

for(i=0;i<4;i++)

for(j=i+1;j<5;j++)

if(a[i ][0]>a[j][0]){t=a[i][0];a[i][0]=a[j][0];a[j][0]=t;}

puts(a[1]); }

A. fwo B.fix C.two D.owo

33. 下面程序对两个整型变量的值进行交换。以下正确的说法是 \_\_\_\_\_\_ 。

main()

{ int a=10,b=20;printf(" (1)a=%d,b=%d\n",a,b);

swap(&a,&b); printf(" (2)a=%d,b=%d\n",a,b); }

swap(int p,int q)

{ int t; t=p; p=q; q=t; }

A. 该程序完全正确

B. 该程序有错，只要将语句 swap(&a,&b); 中的参数改为 a,b 即可

C. 该程序有错，只要将 swap() 函数中形参 p 和 q 以及 t 均定义为指针即可

D. 以上说法都不正确

34. 下面程序的功能是按字典顺序比较两个字符串 a,b 的大小，如果 a 大于 b 则返回正值，等于则返回 0 ，小于则返回负值。下划线处应该填 \_\_\_\_\_\_ 。

#include "stdio.h"

s(char \*s ,char \*t)

{ for( ; \*s==\*t; t++,s++) if(\*s==' \0 ') return 0;

return(\*s-\*t); }

main()

{ char a[20],b[10],\*p,\*q; int i;

p=a; q=b; scanf("%s%s",a,b);

i=s( \_\_\_\_\_\_\_\_\_\_ ); printf("%d",i); }

A. p,q B.q,p C.a,p D.b,q

35. 下面程序的运行结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void delch(char \*s)

{ int i,j; char \*a; a=s;

for(i=0,j=0;a[i]!= ' \0 ';i++)

if(a[i]>= ' 0 ' &&a[i]<= '9'){s[j]=a[i];j++;}

s[j]= ' \0 '; }

main()

{ char \*item="a34bc"; delch(item); printf("\n%s",item); }

A.abc B. 34 C.a23 D.a34bc

36. 以下程序的执行后输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

main()

{ char str[]="xyz",\*ps=str;

while(\*ps) ps++;

for(ps--;ps-str>=0;ps--) puts(ps); }

A.yz B.z C. z D.x

xyz yz yz xy

xyz xyz

37. 以下程序的执行后输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int a[3][3],\*p,i;

p=&a[0][0];

for(i=0;i<9;i++) p[i]=i;

for(i=0;i<3;i++) printf("%d",a[i][j]); }

A.012 B.123 C.234 D. 345

38. 以下程序执行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

void main()

{ int a[]={1,2,3,4},y,\*p=&a[1];

y=(\*--p)++; printf("%d",y); }

A. 1 B.2 C.3 D.4

39. 以下程序执行后的输出结果是 \_\_\_\_\_\_ 。

#include <stdio.h>

#include <string.h>

void main()

{ char s1[10],\*s2="ab\0cdef";

strcpy(s1,s2);

printf("%s",s1); }

A.ab\0cdef B.abcdef C. ab D. 以上答案都不对

40. 若有函数： void fun(double a[],int \*n) { …… } ，以下叙述中正确的是 \_\_\_\_\_\_ 。

A. 调用 fun 函数时只有数组执行按值传送，其他实参和形参之间执行按地址传送

B. 形参 a 和 n 都是指针变量

C. 形参 a 是一个数组名， n 是指针变量

D. 调用 fun 函数时将把 double 型实参数组元素一一对应地传送给形参 a 数组

四、习题答案

（一）、是非题

1 .A 2.A 3.A 4.B 5.B 6.B 7.B 8.A 9.B 10.B

11 .B 12.B 13.B 14.A 15.A 16.B 17.A 18.A 19.A 20.A

（二）、选择题

1 .B 2.C 3.D 4.A 5.A 6.B 7.B 8.D 9.A 10.A

11 .D 12.B 13.A 14.D 15.B 16.A 17.B 18.C 19.B 20.C

21 .B 22.C 23.A 24.B 25.D 26.D 27.A 28.D 29.B 30.A

31 .C 32.A 33.D 34.A 35.B 36.C 37.D 38.A 39.C 40.C

（三）、编程题（ 略 ）

## 第 9 章 文件

二、习题

（一）、判断题

1.C 语言通过文件指针对它所指向的文件进行操作。

A. 对 B. 错

2. 函数 fseek(fp,n,k) 中的第二个参数代表的位移量是相对于的文件的开始来说的。

A. 对 B. 错

3. 函数 rewind() 的作用是位置指针返回移动前的位置。

A. 对 B. 错

4. 为了提高读写效率，在进行读写操作后不应关闭文件以便下次再进行读写。

A. 对 B. 错

5. 使用 fwrite() 向文件中写入数据之前，该文件必须是以 wb 方式打开。

A. 对 B. 错

6. 命令行参数是指在操作系统状态下执行命令程序时允许带的参数。

A. 对 B. 错

7. 在 C 语言中将文件视为无结构的字节流。

A. 对 B. 错

8. 当以参数 w 打开文件时，若指定路径下已有同名文件，则覆盖原有文件。

A. 对 B. 错

9. 可以使用 ftell() 获得文件的当前读写位置。

A. 对 B. 错

10. 文件正常关闭时返回 1 。

A. 对 B. 错

11. 由于在 C 语言中将文件视为无结构的字节流，所以不能对文件进行二进制读写。

A. 对 B. 错

12. 表达式 c=fgetc(fp)!=EOF 的功能是从 fp 指向的文件中读取字符并判断文件是否结束。

A. 对 B. 错

13.C 语言中的文件是一种流式文件，读写时均以字符为单位。

A. 对 B. 错

14. 函数 fopen() 不可以打开二进制文件。

A. 对 B. 错

15. 函数 fgets(s,n,fp) 读出的字符串长为 n 。

A. 对 B. 错

16. 系统打开的标准设备文件不需要用 fclose() 函数关闭。

A. 对 B. 错

17. 语句 puts(fp,"hello"); 可以向 fp 指向的文件中写入字符串。

A. 对 B. 错

18.fseek() 函数中的位移量必须是一个 long 型数据。

A. 对 B. 错

19. 用“ rb ”方式打开的文件必须是已经存在的文件。

A. 对 B. 错

20. 使用 fgetc() 函数读取字符后，文件内部位置指针移动到文件末尾。

A. 对 B. 错

（二）、选择题

1 . 以下函数不能用于向文件写入数据的是 \_\_\_\_\_\_ 。

A. ftell B.fwrite C.fputc D.fprintf

2 . 设 fp 已定义，执行语句 fp=fopen("file","w"); 后，以下针对文本文件 file 操作叙述的选项中正确的是 \_\_\_\_\_\_ 。

A. 写操作结束后可以从头开始读 B. 只能写不能读

C. 可以在原有内容后追加 D. 可以随意读和写

3. 下列关于 C 语言文件的叙述中正确的是 \_\_\_\_\_\_ 。

A. 文件由一系列数据依次排列组成，只能构成二进制文件

B. 文件由结构序列组成，可以构成二进制文件或文本文件

C. 文件由数据序列组成，可以构成二进制文件或文本文件

D. 文件由字符序列组成，其类型只能是文本文件

4 . 读取二进制文件的函数调用形式为： fread(buffer,size,count,fp); ，其中 buffer 代表的是 \_\_\_\_\_\_ 。

A. 一个文件指针，指向待读取的文件

B. 一个整型变量，代表待读取的数据的字节数

C. 一个内存块的首地址，代表读入数据存放的地址

D. 一个内存块的字节数

5. 标准库函数 fgets(buf,n,fp) 的功能是 \_\_\_\_ 。

A. 从 fp 所指向的文件中读取长度为 n 的字符串存入缓冲区 buf

B. 从 fp 所指向的文件中读取长度不超过 n-1 的字符串存入缓冲区 buf

C. 从 fp 所指向的文件中读取 n 个字符串存入缓冲区 buf

D. 从 fp 所指向的文件中读取长度为 n-1 的字符串存入缓冲区 buf

6 . 以下程序完成的功能是 \_\_\_ 。

#include " stdio.h "

main()

{ FILE \*in,\*out;

char ch,infile[10],outfile[10];

printf( " Enter the infile name: " );

scanf( " %s " ,infile);

printf( " Enter the outfile name: " );

scanf( " %s " ,outfile);

if((in=fopen(infile, " r " ))==NULL)

printf( " cannot open infile\n " );

if((in=fopen(outfile, " w " ))= =NULL)

printf( " cannot open outfile\n " );

while(!feof(in))fputc(fgetc(in),out);

fclose(in);

fclose(out); }

A. 程序完成将磁盘文件的信息在屏幕上显示的功能

B. 程序完成将两个磁盘文件合二为一的功能

C. 程序完成将一个磁盘文件复制到另一个磁盘文件中

D. 程序完成将两个磁盘文件合并并在屏幕上输出

7. 以下程序运行后的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ FILE \*fp; int i=20,j=30,k,n;

fp=fopen("dl.dat","w");

fprintf (fp,"%d\n",i);fprintf(fp,"%d\n",j);

fclose (fp);

fp= fopen ("dl.dat","r");

fscanf (fp,"%d%d",&k,&n);

printf ("%d %d\n",k,n);

fclose(fp); }

A. 20 30 B.20 50 C.30 50 D.30 20

8. 阅读下面的程序，程序实现的功能是 (a123.txt 在当前盘符下已经存在 ) \_\_\_\_\_\_\_ 。

#include "stdio.h"

#include "string.h"

void main()

{ FILE \*fp;

int a[10],\*p=a;

fp=fopen("a123.txt","w");

while(strlen(gets(p))>0)

{ fputs(a,fp);

fputs("\n",fp); }

fclose(fp); }

A. 从键盘输入若干行字符，按行号倒序写入文本文件 a123.txt 中

B. 从键盘输入若干行字符，取前 2 行写入文本文件 al23.txt 中

C. 从键盘输入若于行字符，第一行写入文本文件 a123.txt 中

D. 从键盘输入若干行字符，依次写入文本文件 a123.txt 中

9. 下面的程序执行后，文件 test 中的内容是 \_\_\_\_\_ 。

#include " stdio.h "

#include " string.h "

void fun(char \*fname,char \*st)

{ FILE \*myf;int i;

myf=fopen(fname, " w " );

for(i=0;i<strlen(st);i++) fputc(st[i],myf);

fclose(myf);}

main()

{ fun( " test " , " new world " );

fun( " test " , " hello " ); }

A. hello B.new worldhello C.new world D.hello,rld

10 . 阅读下面的程序，此程序的功能为 \_\_\_\_\_\_

#include "stdio.h"

#include "string.h"

main(int argc,char \*argv[])

{ FILE \*p1,\*p2;

int c;

p1=fopen(argv[1], "r");

p2=fopen(argv[2], "a");

c=fseek(p2,0L,2);

while((c=fgetc(p1))!=EOF)fputc(c,p2);

fclose(p1);

fclose(p2); }

A. 实现将 p1 打开的文件中的内容复制到 p2 打开的文件

B. 实现将 p2 打开的文件中的内容复制到 p1 打开的文件

C. 实现将 p1 打开的文件中的内容追加到 p2 打开的文件内容之后

D. 实现将 p2 打开的文件中的内容追加到 p1 打开的文件内容之屠

11 . fseek 函数的正确调用形式是 \_\_\_\_\_ 。

A. fseek( 文件指针 , 起始点 , 位移量 ) B. fseek( 文件指针 , 位移量 , 起始点 )

C. fseek( 位移量 , 起始点 , 文件指针 ) D. fseek( 起始点 , 位移量 , 文件指针 )

12 . 若 fp 是指向某文件的指针，且已读到文件末尾，则函数 feof(fp) 的返回值是 \_\_\_\_\_ 。

A.EOF B. -1 C.1 D.NULL // 结束标志 -1

13 . 函数 fseek(pf,OL,SEEK\_END) 中的 SEEK\_END 代表的起始点是 \_\_\_\_\_\_\_ 。

A. 文件开始 B. 文件末尾 C. 文件当前位置 D. 以上都不对

14 . 16 位系统中，将一个整数 12 000 分别以 ASCII 码文件和二进制文件形式存放，各自需 要占据的存储空间字节数分别是 \_\_\_\_\_\_ 。

A . 5 和 2 B . 2 和 5 C . 2 和 2 D . 5 和 5

15 . 若要用 fopen() 函数打开一个新的二进制文件，该文件要既能读也能写，则文件读 方式字符串应是 \_\_\_\_\_\_\_ 。

A . " a+ " B . " r+ " C . " rb+ " D. " ab+ "

16 . 函数 fscanf 的功能是 \_\_\_\_\_\_ 。

A . 将信息从文件输入 B . 将信息输出到文件

C . 将信息从控制台输入 D . 将信息输出到控制台终端上

17 . 下面叙述中，不正确的是 \_\_\_\_\_\_\_ 。

A . C 语言中的文本文件以 ASCII 形式存储数据

B . C 语言对二进制文件的访问速度比文本文件快

C . C 语言中，随机读写方式不适用于文本文件

D . C 语言中，顺序读写方式不适用于二进制文件

18. 如果程序中有语句 FILE \*fp ； fp=fopen( " abc.txt " , " w " ); ，则程序准备做 \_\_\_\_\_\_ 。 A . 对文件读写操作 B . 对文件读操作 C . 对文件写操作 D . 对文件不操作

19 . 使用函数 fopen() 以文本方式打开或建立可读可写文件，要求：若指定的文件不存在，则 新建一个，并使文件指针指向其开头；若指定的文件存在，打开它，将文件指针指向其结尾。正确的“文件使用方式”是 \_\_\_\_\_\_ 。 A . " r+ " B . " w+ " C. " a+ " D . " a "

20 . 若定义： int a[5] ；， fp 是指向某一已经正确打开了的文件的指针， 下面的函数调用形式中不正确的是 \_\_\_\_\_\_\_\_ 。

A . fread(a[0],sizeof(int),5,fp); B . fread(&a[0],5\*sizeof(int),1,fp); C . fread(a,sizeof(int),5,fp); D . fread(a,5\*sizeof(int),1,fp);

21 . 若要打开 D 盘上子目录 tt 下的二进制文件 test.bin ，在调用函数 fopen 时， 第一个参数的正确格式是 ( ) 。 A . " d:tt\\test.bin " B . " d:\tt\\test.bin "

C. " d:\\tt\\test.bin " D . " d:\tt\test.bin "

22 . 正常执行文件关闭操作时， fclose() 函数的返回值是 \_\_\_\_\_\_ 。

A . -1 B . TRUE C . 0 D . 1

23 . feof() 函数 \_\_\_\_\_\_ 。

A . 可用于二进制文件也可用于文本文件 B . 只用于二进制文件

C . 不能用于二进制文件 D . 只能用于文本文件

24 . 语句“ fseek(fp,-100L,1); ”的功能是 \_\_\_\_\_\_\_ 。

A . 将 fp 所指向的文件的读写指针移到距文件首 100 个字节处

B . 将 fp 所指向的文件的读写指针移到距文件尾 100 个字节处

C . 将 fp 所指向的文件的读写指针从当前位置向文件首方向移动 100 个字节

D . 将 fp 所指向的文件的读写指针从当前位置向文件尾方向移动 100 个字节

25 . 在 C 程序中，可把整型数以二进制形式存放到文件中的函数是 \_\_\_\_\_\_ 。 A . fprintf() B . fread() C . fwrite() D . fputc()

26 . 下面关于 C 语言文件操作的结论中， \_\_\_\_\_\_ 是正确的。 A . 对文件操作必须先关闭文件

B . 对文件操作必须先打开文件

C . 对文件操作顺序无要求

D . 对文件操作前必须先测试文件是否存在，然后再打开文件

27 . 如果需要打开一个已经存在的非空文件“ f ”进行修改，正确的打开语句是 \_\_\_\_\_ 。 A . fp=fopen( " f " , " r " ); B . fp=fopen( " f " , " ab+ " ); C . fp=fopen( " f " , " w+ " ); D. fp=fopen( " f " , " r+ " );

28 . fsca nf() 函数的正确调用形式是 \_\_\_\_\_\_\_ 。 A . fscanf( 文件指针 , 格式字符串 , 输出表列 );

B . fscanf( 格式字符串 , 输出表列 , 文件指针 ); C . fscanf( 格式字符串 , 文件指针 , 输出表列 );

D . fscanf( 文件指针 , 格式字符串 , 输入表列 );

29 . 下面不是系统指定的标准设备文件有 \_\_\_\_\_\_\_ 。 A . fopen () B . stdin () C . stdout () D . stderr! ()

30 . 下面 \_\_\_\_\_\_\_ 操作后，文件的读写指针不指向文件首。

A . rewind(fp) B . fseek(fp,0L,0) C. fseek(fp,0L,2) D . fopen( " f1.c " , " r " )

31 . 下面程序把终端输入的字符输出到名为 abc.txt 的文件中，直到从终端读入字符“ # ”时 结束输入和输出操作程序有错，出错原因是 \_\_\_\_\_ 。

#include " stdio.h " void main(){ FILE \*fout; char ch;

fout=fopen('abc.txt', 'w');

ch=fgetc(stdin);

while(ch!= '#') { fputc(ch,fout); ch=fgete(stdin);}

fclose(fout); } A . 函数 fopen() 调用形式有误 B . 输入文件没有关闭 C . 函数 fgetc () 调用形式有误 D . 文件指针 stdin () 没有定义

32 . 函数 fgetc() 的作用是从文件读入一个字符，该文件的打开方式必须是 \_\_\_\_\_\_ 。

A. 读或读写 B. 追加 C. 只写 D. 答案 B 和 C 都正确

33 . 利用 fseek() 函数可实现的操作是 \_\_\_\_\_\_ 。

A. 文件的随机读写 B. 文件的顺序读写

C. 改变文件的位置指针 D. 以上答案均正确

34 . 若执行 fopen() 函数时发生错误，则函数的返回值是 \_\_\_\_\_\_ 。

A. 地址值 B. EOF C. 0 D.1

35 . 系统的标准输入文件是指 \_\_\_\_\_\_ 。

A. 显示器 B. 硬盘 C. 键盘 D. 软盘

36 . 以下程序的运行结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ FILE \*fp; int a[10]={1,2,3,0,0},i;

fp=fopen("d2.dat,"wb");

fwrite(a,sizeof(int),5,fp);

fwrite(a,sizeof(int),5,fp);

fclose(fp);

fp=fopen("d2.dat","rb");

fread(a,sizeof(int),10,fp);

fclose(fp);

for(i=0;i<10;i++) printf("%d",a); }

A. 1,2,3,0,0,0,0,0,0,0, B. 1,2,3,1,2,3,0,0,0,0,

C. 123,0,0,0,0,123,0,0,0,0, D. 1,2,3,0,0,1,2,3,0,0,

37 . 以下程序的运行结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ FILE \*fp;

int a[10]={1,2,3},i,n;

fp=fopen("dl.dat","w");

for(i=0;i<3;i++) fprintf(fp,"%d",a[i]);

fprintf(fp,"\n");

fclose(fp);

fp=fopen("dl.dat","r");

fscanf(fp,"%d",&n);

fclose(fp);

printf("%d\n",n); }

A.12300 B. 123 C. 1 D.321

38 . 以下程序执行后， abc.dat 文件的内容是 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ FILE \*pf;

char \*s1="China",\*s2="Beijing";

pf=fopen("abc.dat","wb+");

fwrite(s2,7,l,pf);

rewind(pf);

fwrite(s1,5,1,pf);

fclose(pf); }

A.China B. Chinang C.ChinaBeijing D.BeijingChina

39 . 若文本文件 filea.txt 中原有内容为： hello ，则运行以下程序后，文件 filea.txt 中的内容为 \_\_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ FILE \*f;

f=fopen("filea.txt","w");

fprintf(f,"abc");

fclose(f); }

A.helloabc B.abclo C. abc D.abchello

40 . 以下程序 运行后的输出结果是 \_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ FILE \*fp,

int k , n , j , a[6]={1 , 2 , 3 , 4 , 5 , 6};

fp=fopen("d2.dat","w");

for(i=0;i<6;i++) fprintf(fp,"%d\n",a[i]);

fclose(fp);

fp=fopen("d2 . dat" , "r");

for(i=0;i<3;i++) fscanf(fp,"%d,%d" , &k , &n);

fclose(fp);

printf("%d , %d\n" , k , n); }

A.1,2 B.3,4 C. 5,6 D.123.456

四、习题答案

（一）、是非题

1. A 2. B 3. B 4. B 5. A 6. A 7. A 8. A 9. A 10. B

11. B 12. A 13. A 14. B 15. B 16. A 17. B 18. A 19. A 20. B

（二）、选择题

1. A 2. B 3. C 4. C 5. B 6. B 7. A 8. D 9. A 10. C

11. B 12. B 13. B 14. A 15. D 16. A 17. B 18. C 19. C 20. A

21. C 22. C 23. A 24. C 25. C 26. B 27. D 28. D 29. A 30. C

31. A 32. A 33. C 34. C 35. C 36. D 37. B 38. B 39. C 40. C

## 第 10 章 位运算

二、习题

（一）是非题

1. 用Ｃ语言可直接进行位运算，因此Ｃ语言是一种低级语言。

A. 对 B. 错

2 . 参加位运算的数据可以是任何类型的数据。

A. 对 B. 错

3 . 在一个数左移时被溢出的高位中不包含 1 的情况下，左移 1 位相当于该数乘以 2 。

A. 对 B. 错

4 . 有表达式 y=~5 ，则 y 的值等于 -5

A. 对 B. 错

5 . 无论 x 的取值如何，关系表达式 x^x==0 的结果均为“真”。

A. 对 B. 错

6 . 位段的存储位置与长度可以由程序员根据需要自由确定。

A. 对 B. 错

7 . 对数值进行右移操作，负数左侧空位一律补 1 。

A. 对 B. 错

8 . 位段只能够以结构体成员的方式进行定义。

A. 对 B. 错

9 . 指针变量无法指向某一数值的特定的位。

A. 对 B. 错

10 . 两个整型变量，为实现两者的值交换，必须依赖第三个变量。

A. 对 B. 错

11 . 结构体 a 的位段成员 int b:4 ，可以通过 scanf("%d",&a.b) 的形式完成赋值。

A. 对 B. 错

12 . 有宏定义 #define MASK 2 ，则 flag=flag|MASK; 可以对 flag 低位第 2 个 bit 置 1 。

A. 对 B. 错

13 . 无名位段不能引用，但同样占用内存地址空间。

A. 对 B. 错

14 . 位运算得到的计算结果可以直接以二进制的形式输出。

A. 对 B. 错

15 . 位运算能够完成加法和乘法运算，但无法完成除法运算。

A. 对 B. 错

（二）选择题

1. 以下程序段的运行结果是 \_\_\_\_\_\_\_\_\_\_ 。

char x=56; x=x&056; printf("%d,%o \ n",x,x);

A . 56,70 B . 0,0 C . 40,50 D . 62,76

2. 用双字节存储整数，表达式 ~0x13 的值是 \_\_\_\_\_\_\_\_\_\_\_\_\_ 。

A .0 XFFEC B .0 XFF71 C .0 XFF68 D .0 XFF17

3 . 设有以下语句段。则 z 的二进制值是 \_\_\_\_\_\_\_\_\_\_ 。

char x=3,y=6,z; z=x^y<<2;

A . 00010100 B . 00011011 C .0 0011100 D . 00011000

4. 语句 printf("%d \n",12 &012); 的输出结果是 \_\_\_\_\_\_\_ 。

A.12 B.8 C.6 D.012

5. 设 int b=2; ，则 表达式 (b>>2)/(b>>1)) 的值是 \_\_\_\_\_\_\_\_\_ 。

A.8 B.4 C.2 D.0

6. 执行下面的程序段后 ， b 的值为 \_\_\_\_\_\_\_\_ 。

int x=35,b;char z='A';b=((x&15)&&(z<'a'));

A.0 B.1 C.2 D.3

7. 设二进制数 a 是 00101101 ，若想通过异或运算 a^b 使 a 的高 4 位取反，低 4 位不变，则二进制数 b 应是 \_\_\_\_\_\_\_\_\_\_ 。

A.00000000 B.00001111 C.11110000 D.11111111

8. 下列程序的输出是 \_\_\_\_\_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ char x=040; printf("%d\n",x=x<<1); }

A.100 B.160 C.120 D.64

9. 设有如下定义： int x=1,y=-1; ，则语句： printf("%d\n",(x--&y)); 的输出结果是 \_\_\_\_\_\_\_ 。

A.1 B.0 C.-1 D.2

10. 设位段的空间分配由右到左，则以下程序的运行结果是 \_\_\_\_\_\_\_ 。

struct bit

{ unsigned a:2;

unsigned b:3;

unsigned c:4;

int i; } data;

#include "stdio.h"

main()

{ data.a=8,data.b=2;

printf("%d%d\n",data.a,data.b); }

A. 语法错误 B.02 C.6 D.82

11. 以下程序运行后的输出结果是 \_\_\_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ unsigned int a,b;

a=7^3;

b=~4&3;

printf("%d,%d\n",a,b); }

A.4 3 　　　 B.7 3 　　　 C.7 0 　　　 D.4 0

12. 设有定义语句： char c1=92,c2=92; ，则以下表达式中值为零的是 \_\_\_\_\_\_\_\_\_ 。

A.c1&c2 B.~c2 C.c1^c2 D.c1|c2

13. 设 char 型变量 x 中的值为 10100111 ，则表达式 (2+x)^(~3) 的值是 \_\_\_\_\_\_\_\_ 。

A.10101001 B.10101000 　 C. 11111101 D. 01010101

14. x 为任意整数，能将变量 x 清零的表达式是 \_\_\_\_\_\_\_\_\_ 。

A.x+(~x) B .x&0 C.x^(~x) D.x|0

15. 利用位运算，能够将变量 ch 中的大写字母转换为小写字母的表达式是 \_\_\_\_\_\_\_ 。

A.ch=ch&32 B.ch=ch|32 C.ch=ch<<4 D.ch=ch>>4

16. 下列表达式， \_\_\_\_\_\_\_\_\_\_ 可以实现对十六进制数 0xA5 除以 8 ，然后再赋给变量 x 。

A.x=0xA5<<3 B.x=0xA5<<8 C.x=0xA5>>3 D.x=0xA5>>8

17 . 如下宏定义，对其作用描述正确的是 \_\_\_\_\_\_\_\_\_\_ 。

#define BitGet(Number,pos) ((Number)>>(pos-1)&1)

A. 使 Number 的值除以 pos-1

B. 对 Number 的值除以 pos-1 后，与 1 进行按位与

C. 对数值 Number 的第 pos 位置 1 处理

D. 取数值 Number 的第 pos 位的二进制值

18 . 以下叙述中不正确的是 \_\_\_\_\_\_\_\_\_ 。

A. 表达式 a&=b 等价于 a=a&b B. 表达式 a|=b 等价于 a=a|b

C. 表达式 a~=b 等价于 a=a~b D. 表达式 a^=b 等价于 a=a^b

19. 以下程序的运行结果是 \_\_\_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ char a=0x95,b,c;

b=(a&0xf)<<4;

c=(a&0xf0)>>4;

a=b|c;

printf("%x\n",a);}

A.0 B.95 C.89 D.59

20 . 以下程序的运行结果为 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 。

#include "stdio.h"

main()

{ unsigned char a=2,b=4,c=5,d;

d=a|b; d&=c;

printf("%d\n",d); }

A.3 B.4 C.5 D.6

四、习题答案

（一）、是非题

1. B 2. B 3. A 4. B 5. A 6. B 7. B 8. A 9. A 10. B

11. B 12. A 13. A 14. B 15. B

（二）、选择题

1. C 2. A 3. B 4. B 5. D 6. B 7. C 8. D 9. A 10. B

11. A 12. C 13. D 14. B 15. B 16. C 17. D 18. C 19. D 20. B

## 第 11 章 面向对象的程序设计

二、习题

（一）、是非题

1. 使用关键字 class 定义的类中默认的访问权限是私有 (private) 的。

A. 对 B. 错

2. 作用域运算符 (::) 只能用来限定成员函数所属的类。

A. 对 B. 错

3. 析构函数是一种函数体为空的成员函数。

A. 对 B. 错

4. 构造函数和析构函数都不能重载。

A. 对 B. 错

5. 构造函数不能声明返回值类型。

A. 对 B. 错

6. 友元关系不具有传递性。

A. 对 B. 错

7. 类的有友元可以是另一个类的成员。

A. 对 B. 错

8. 要使成员在类族（基类及其派生类）中被访问，而在类族结构之外不能访问，这些成员要定义为私有的（ private ）。

A. 对 B. 错

9 . 对象成员的引用与结构变量成员的引用方法相同，都是使用运算符．或 -> 。

A. 对 B. 错

10 . 所谓私有成员是指只有类中所提供的成员函数才能直接访问它们，任何类外的函数对它们的访问都是非法的。

A. 对 B. 错

11 . 某类中的友元类的所有成员函数可以存取或修改该类中的私有成员。

A. 对 B. 错

12 . 可以在类的构造函数中对静态数据成员进行初始化。

A. 对 B. 错

13 . 如果一个成员函数只存取一个类的静态数据成员，则可将该成员函数说明为静态成员函数。

A. 对 B. 错

14. 类中的任何一个函数都可声明为内联函数，即内联函数的声明没有任何限制。

A. 对 B. 错

15 . 使用内联函数是以牺牲（增大）代码空间开销为代价的。

A. 对 B. 错

16. 派生类继承了除基类的构造函数和析构函数外的其他成员。

A. 对 B. 错

17. 要使成员在类族（基类及其派生类）中被访问，而在类族结构之外不能访问，这些成员要定义为保护成员（ protected ）。

A. 对 B. 错

18 . 内联函数的关键字为 void 。

A. 对 B. 错

19 . 类是具有相同属性和行为的一组对象的集合。

A. 对 B. 错

20 . 对象是系统中用来描述客观事物的一个实体，它是用来构成系统的一个基本单位。

A. 对 B. 错

21 . 封装就是把对象的属性和行为结合成一个独立的系统单位，并尽可能隐蔽对象的内部细节。

A. 对 B. 错

22 . 构造函数和析构函数都可以重载。

A. 对 B. 错

23 . 友元关系可以继承。

A. 对 B. 错

24.new 的功能是动态分配内存。

A. 对 B. 错

25 . 析构函数是在对象的生存期即将结束的时刻被自动调用的。

A. 对 B. 错

26 . 说明或定义对象时，类名前面不需要加 class 关键字。

A. 对 B. 错

27.C++ 中的 this 指针指出了成员函数当前所操作的数据所属的对象。

A. 对 B. 错

28.delete 用来删除由 new 建立的对象，释放指针所指向的内存空间。

A. 对 B. 错

29 . 构造函数不可以带参数。

A. 对 B. 错

30 . 对象数组的元素是不同类的对象。

A. 对 B. 错

31 . 带有纯虚函数的类是抽象类。

A. 对 B. 错

32 . 虚函数声明只能出现在类声明中的函数原型声明中，而不能在成员函数实现的时候。

A. 对 B. 错

33. 当类的继承方式为公有继承时，基类的私有成员可以被派生类访问。

A. 对 B. 错

34. 派生类不能访问其基类的私有成员。

A. 对 B. 错

35. 派生类不能继承基类的析构函数。

A. 对 B. 错

（二）、选择题

1 . C++ 中实现数据输出的输出流格式是 \_\_\_\_\_\_ 。

A.cprintf( " 格式串 " , 表达式 ) B.printf( " 格式串 " , 表达式 )

C.out>> " 输出字符 " D.cout<< 表达式

2. 类的某个成员函数如果要被外部函数引用，则该成员函数定义时的访问限定符应为 \_\_\_\_\_\_ 。

A.private B.public

C.protected D. 任意一种均可

3. 以下说法错误的是 \_\_\_\_\_\_ 。

A . 类的成员函数可以放在类中定义，也可以在类外定义

B . 内联函数就是在类的成员函数内定义的函数

C . new 操作返回的是分配的存储空间的指针，可代替 C 语言中的 malloc() 函数

D . 访问限定符为 protected 的成员，可供该类对象及其派生类对象成员函数引用

4. 有如下类的定义，则下列定义对象的方法是错误的 \_\_\_\_\_\_ 。

class cat

{ char variety[20], color[15];

int age;

float weight;

public: void input();

void disp();

void cat(){ strcpy(variety, " " );strcpy(color, " " );

age=0; weight=0.0; }

void cat(char \*ch,int Age) {strcpy(variety,ch); age=Age; }

}

A.cat Mini; B.cat Mini( " Fox " ,2);

C.cat Mini( " Fox " , " 2 " ) D.cat \*Mini =new dog;

5. 有如下类和对象的定义，则下列对象的引用方法是错误的 \_\_\_\_\_\_ 。

class Bpoint

{ int x,y;

public: Bpoint() { x=0; y=0; }

void setpoint(int x0,int y0);

void show(); };

void Bpoint::setpoint(int x0,int y0)

{ x=x0; y=y0; }

void Bpoint::show()

{ cout<< " ( " <<x<< " , " <<y<< " ) " ; }

Bpoint Boundary[100];

A. Boundary[0].setpoint(120,50) ; B. Boundary.show( ) ;

C. Boundary[10].show( ) ; D. Boundary[20]->setpoint(10,20) ;

6. 下列有关内联函数的叙述中，正确的是 \_\_\_\_\_\_ 。

A. 内联函数在调用时发生控制转移

B. 内联函数必须通过关键字 inline 来定义

C. 内联函数是通过编译器来实现的

D. 内联函数函数体的最后一条语句必须是 return 语句

7. 下列情况中，不会调用拷贝构造函数的是 \_\_\_\_\_\_ 。

A. 用一个对象去初始化同一类的另一个新对象时

B. 将类的一个对象赋值给该类的另一个对象时

C. 函数的形参是类的对象，调用函数进行形参和实参结合时

D. 函数的返回值是类的对象，函数执行返回调用时

8. 下列有关继承和派生的叙述中，正确的是 \_\_\_\_\_\_ 。

A. 如果一个派生类私有继承其基类，则该派生类对象不能访问基类的保护成员

B. 派生类的成员函数可以访问基类的所有成员

C. 基类对象可以赋值给派生类对象

D. 如果派生类没有实现基类的一个纯虚函数，则该派生类是一个抽象类

9. 下列运算符不能重载为友元函数的是 \_\_\_\_\_\_ 。

A.= () [] ->     B.+ - ++ --

C.> < >= <=         D.+= -= \*= /=

10. 下列语句中，错误的是 \_\_\_\_\_\_ 。

A.const int buffer=256;     B.const double \*point;

C.int const buffer=256;     D.double \* const point;

11. 以下关键字不能用来声明类的访问权限的是 \_\_\_\_\_\_ 。

A.public    B.protected

C.static            D.private

12. 在公有继承的情况下，允许派生类直接访问的基类成员包括 \_\_\_\_\_\_ 。

A. 公有成员       B. 公有成员和保护成员

C. 公有成员、保护成员和私有成员   D. 保护成员

13. 关于运算符重载，下列表述中正确的是 \_\_\_\_\_\_ 。

A.C++ 已有的任何运算符都可以重载

B. 运算符函数的返回类型不能声明为基本数据类型

C. 在类型转换符函数的定义中不需要声明返回类型

D. 可以通过运算符重载来创建 C++ 中原来没有的运算符

14. 以下程序，运行时的输出结果是 \_\_\_\_\_\_ 。

#include <iostream.h>

#include <string.h>

class Toy

{ char name[10];

static int count; // 静态成员变量，该类的所有对象共享

public:

Toy(char \*n) { strcpy(name,n); count++;}

~Toy() { count--; }

　　 char \*GetName() { return name; }

　　 static int getCount(){ return count; }

// 静态成员函数，该类的所有对象共享

} ;

int Toy::count=0; // 静态成员变量赋值

int main()

{ 　 Toy t1( " Snoopy " ),t2( " Mickey " ),t3( " Barbie " );

　 cout<<t1.getCount()<<endl;

　 return 0;

}

A . 1 　　 B . 2 　　 C . 3 　　 D . 运行时出错

15. 有如下程序，执行后屏幕上将输出 \_\_\_\_\_\_ 。

#include <iostream.h>

class MyClass

{ public:

  MyClass() { cout<<'A'; }

 MyClass(char c) { cout<<c;}

 ~MyClass(){ cout<<'B'; }

} ;

void main()

{ MyClass p1,\*p2; // 定义对象和对象指针

p2=new MyClass('X'); // 动态创建对象，让指针 p2 指向新创建的对象

delete p2;} // 删除对象

A.ABX     B.ABXB      C.AXB D.AXBB

16. 下列程序运行时的结果是 \_\_\_\_\_\_ 。

#include <iostream>

using namespace std;

class A

{ int r1;

public:

A(int i):r1(i) { } // 构造函数，初始化 r1 为 i 的值

void print() {cout<<'E'<<r1<<'-';}

void print() const {cout<<'C'<<r1\*r1<<'-';}

};

void main()

{ A a1(2);

const A a2(4);

a1.print();

a2.print();

}

A . 运行时出错　　 B . E2-C16- 　　 C . C4-C16- 　　 D . E2-E4-

17. 如下程序 的 程序的输出结果是 \_\_\_\_\_\_ 。

#include <iostream>

using namespace std;

class Obj

{ static int i;

public:

Obj(){ i++; }

~Obj(){ i--; }

static int getVal( ) { return i;}

};

int Obj::i=0;

void f()

{ Obj ob2;

cout<<ob2.getVal(); }

void main()

{ Obj ob1;

f();

Obj \*ob3=new Obj;

cout<<ob3->getVal();

delete ob3;

cout<<ob1.getVal();

}

A.232    B.231    C.221    D.222

18. 执行如下程序 ， 屏幕上将显示输出 \_\_\_\_\_\_ 。

#include <iostream.h>

class Base

{ protected:

Base(){ cout<<'A'; }

Base(char c){ cout<<c;}

};

class Derived: public Base

{ public:

Derived(char c){ cout<<c;}

};

void main(){ Derived d1('B'); }

A.B      B.BA     C.AB         D.BB

19. 有如下类定义 ， 编译时发现有一处语法错误 ， 对这个错误最准确的描述是 \_\_\_\_\_\_ 。

class MyBase

{ int k;

public:

   MyBase(int n=0):k(n){ }

  int value()const{ return k;}

};

class MyDerived: MyBase

{ int j;

public:

   MyDerived(int i):j(i) {}

  int getK()const{ return k; }

  int getJ()const{ return j; }

};

A . 函数 getK 试图访问基类的私有成员变量 k

B. 类 MyDerived 的定义中 ， 基类名 MyBase 前缺少关键字 public 、 protected 或 private

C. 类 MyDerived 缺少一个无参的构造函数

D . 类 MyDerived 的构造函数没有对基类数据成员 k 进行初始化

20. 在一个派生类对象结束其生命周期时，下列说法正确的是 \_\_\_\_\_\_ 。

A . 先调用派生类的析构函数后调用基类的析构函数

B . 先调用基类的析构函数后调用派生类的析构函数

C . 如果基类没有定义析构函数，则只调用派生类的析构函数

D . 如果派生类没有定义析构函数，则只调用基类的析构函数

21. 有如下的运算符重载函数定义： double operator +(int i, int k){ return double(i+k); } ， 但定义有错误，对这个错误最准确的描述是 \_\_\_\_\_\_ 。

A . + 只能作为成员函数重载，而这里的 + 是作为非成员函数重载的

B . 两个 int 型参数的和也应该是 int 型，而这里将 + 的返回类型声明为 double

C . 没有将运算符重载函数声明为某个类的友元

D . C++ 已经提供了求两个 int 型数据之和的运算符 + ，不能再定义同样的运算符

22. 有如下程序，运行后屏幕上将显示输出 \_\_\_\_\_\_ 。

#include <iostream.h>

class A

{ public:

   virtual void func1(){ cout<<"A1"; }

   void func2(){ cout<<"A2"; }

};

class B:public A

{ public:

   void func1(){ cout<<"B1"; }

   void func2(){ cout<<"B2"; }

};

int main()

{ A \*p=new B;

  p->func1();

  p->func2();

  return 0;

}

A.B1B2         B.A1A2   C.B1A2   D.A1B2

23.C++ 流中重载的运算符 >> 是一个 \_\_\_\_\_\_ 。

A. 用于输出操作的非成员函数 B. 用于输入操作的非成员函数

C. 用于输出操作的成员函数 D. 用于输入操作的成员函数

24. 有以下类定义，下列语句中会发生编译错误的是 \_\_\_\_\_\_ 。

class Point

{ int \_x,\_y;

public:

Point(int x=0,int y=0) { \_x=x; \_y=y; }

void Move(int xOff, int yOff)

{ \_x+=xOff; \_y+=yOff; }

void Print() const

{ cout<<'('<<\_x <<','<<\_y<<')'<<endl; }

};

A . Point pt; pt.Print(); B . const Point pt; pt.Print();

C . Point pt; pt.Move(l,2); D . const Point pt; pt.Move(l,2);

25. 有以下类定义，下列类对象定义语句中错误的是 \_\_\_\_\_\_ 。

class MyClass

{ private:

int id;

char gender;

char \*phone;

public:

MyClass():id(0),gender('#'),phone(NULL) { }

MyClass(int no,char ge='#',char \*ph=NULL)

{ id=no;

gender=ge;

phone=ph; }

};

A.MyClass myObj; B.MyClass myObj(11,"133\*\*\*\*\*\*\*\*");

C.MyClass myObj(12,'m'.); D.MyClass myObj(12);

26. 有以下程序，执行后的输出结果是 \_\_\_\_\_\_ 。

#include <iostream>

using namespace std;

class Complex

{ public:

Complex(double r=0, double i=0):re(r), im(i) { }

double real() const { return re; }

double imag() const { return im;}

Complex operator +(Complex c) const

{ return Complex(re+c.re, im+c.im); }

private:

double re,im;

};

void main()

{ Complex a = Complex(1, 1) + Complex(5);

cout<<a.real()<<'+'<< a.imag()<<'i'<<endl;

}

A.6+6i B.6+1i C.1+6i D.1+1i

27. 有以下类定义，则执行语句 MyClass a, b[2], \*p[2]; 后，程序的输出结果是 \_\_\_\_\_\_ 。

class MyClass

{ public:

MyClass(){cout<<1;}

};

A.11 B.111 C.1111 D.11111

28. 关于友元，下列说法错误的是 \_\_\_\_\_\_ 。

A. 如果类 A 是类 B 的友元，那么类 B 也是类 A 的友元

B. 如果函数 fun() 被说明为类 A 的友元，那么在 fun() 中可以访问类 A 的私有成员

C. 友元关系不能被继承

D. 如果类 A 是类 B 的友元，那么类 A 的所有成员函数都是类 B 的友元

29. 关于动态存储分配，下列说法正确的是 \_\_\_\_\_\_ 。

A.new 和 delete 是 C++ 语言中专门用于动态内存分配和释放的函数

B. 动态分配的内存空间也可以被初始化

C. 当系统内存不够时，会自动回收不再使用的内存单元，因此程序中不必用 delete 释放内存空间

D. 当动态分配内存失败时，系统会立刻崩溃，因此一定要慎用 new

30. 有以下程序，执行的结果是 \_\_\_\_\_\_ 。

#include <iostream>

#include <iomanip>

using namespace std;

void main()

{ int s[]={123, 234};

cout << right<< setfill('\*')<<setw(6);

for(int i=0; i<2; i++) { cout<<s[i]<<endl; }

}

　　 A . 123 　　 B . \*\*\*123 　　 C . \*\*\*123 　　 D . \*\*\*123

　　 234 234 \*\*\*234 　　 234\*\*\*

31. 在公有派生的情况下，派生类中定义的成员函数只能访问原基类的 \_\_\_\_\_\_ 。

A. 公有成员和私有成员 B. 私有成员和保护成员

C. 公有成员和保护成员 D. 私有成员、保护成员和公有成员

32. 在 C++ 中用来实现运行时多态性的是 \_\_\_\_\_\_ 。

A. 重载函数 B. 析构函数

C. 构造函数 D. 虚函数

33. 一个类可以同时继承多个类，称为多继承。下列关于多继承和虚基类的表述中，错误的是 \_\_\_\_\_\_ 。

A. 每个派生类的构造函数都要为虚基类构造函数提供实参

B. 多继承时有可能出现对基类成员访问的二义性问题

C. 使用虚基类可以解决二义性问题并实现运行时的多态性

D. 建立派生类对象时，虚基类的构造函数会首先被调用

34. 在一个类体的下列声明中，正确的纯虚函数声明是 \_\_\_\_\_\_ 。

A . virtual void vf()=0; B . void vf(int)=0;

C . virtual int vf(int); D . virtual void vf(int) { }

35. 在下面的运算符重载函数的原型中，错误的是 \_\_\_\_\_\_ 。

A . Volume operator-(double,double); B . double Volume::operator-(double);

C . Volume Volume::operator-(Volume); D . Volume operator-(Volume,Volume);

四、习题答案

（一）、是非题

1 .A 2.B 3.B 4.B 5.A 6.A 7.A 8.B 9.A 10.B

11 .A 12.B 13.A 14.A 15.A 16.A 17.A 18.B 19.A 20.A

21 .A 22.B 23.A 24.A 25.A 26.A 27.A 28.A 29.B 30.B

31 .A 32.A 33.B 34.A 35.A

（二）、选择题

1 .D 2.B 3.B 4.C 5.B 6.C 7.B 8.D 9.A 10.C

11 .C 12.B 13.B 14.C 15.D 16.B 17.C 18.C 19.A 20.A

21 .D 22.C 23.D 24.D 25.B 26.B 27.B 28.A 29.A 30.B

31 .C 32.D 33.A 34.A 35.A

## 第 12 章 程序设计基础知识

二、习题

（一）、是非题

1. 算法的空间复杂度是指执行算法所需要的计算工作量，可以执行算法的过程中所需要的基本运算的执行次数来度量。

A. 对 B. 错

2. 数据的存储结构是指数据的逻辑结构在计算机存储器中的存放形式，称数据的物理结构。

A. 对 B. 错

3. 队列是指允许在一端（队尾）进入插入，而在另一端（队头）进行删除的线性表。

A. 对 B. 错

4. 树是一种简单的线性结构，所有元素之间具有明显的层次特性。

A. 对 B. 错

5. 在一棵二叉树上第 5 层的结点数最多是 32 。

A. 对 B. 错

6. 在最坏情况下，冒泡排序的时间复杂度为 n(n-1)/2 。

A. 对 B. 错

7. 堆排序法在最坏的情况下需要比较的次数为 O(nlog 2 n) 。

A. 对 B. 错

8. 软件 测试的目的是发现错误的位置并改正错误；而调试的目的是暴露错误，评价程序的可靠性。

A. 对 B. 错

9. 如果一个工人可管理多个设施，而一个设施只被一个工人管理，则实体 " 工人 " 与实体 " 设施 " 之间存在一对一联系。

A. 对 B. 错

10. 数据库的物理设计主要目标是对数据内部物理结构作调整并选择合理的存取路径，以提高数据库访问速度有效利用存储空间。

A. 对 B. 错

（二）、选择题 ( 1 至 17 题为 数据结构与算法部分 、 18 至 29 题为 软件工程基础部分、 30 至 46 题为 数据库设计基础部分 )

1. 算法的时间复杂度是指 \_\_\_\_\_\_ 。

A. 执行算法程序所需要的时间

B. 算法程序的长度

C. 算法执行过程中所需要的基本运算次数

D. 算法程序中的指令条数

2. 下面叙述正确的是 \_\_\_\_\_\_ 。

A. 算法的执行效率与数据的存储结构无关

B. 算法的空间复杂度是指算法程序中指令（或语句）的条数

C. 算法的有穷性是指算法必须能在执行有限个步骤之后终止

D. 以上三种描述都不对

3. 算法分析的目的是 \_\_\_\_\_\_ 。

A. 找出数据结构的合理性 B. 找出算法中输入和输出之间的关系

C. 分析算法的易懂性和可靠性 D. 分析算法的效率以求改进

4. 数据结构中，与所使用的计算机无关的是数据的 \_\_\_\_\_\_ 。

A. 存储结构 B. 物理结构 C. 逻辑结构 D. 物理和存储结构

5. 以下数据结构中不属于线性数据结构的是 \_\_\_\_\_\_ 。

A. 队列 B. 线性表 C. 栈 D. 二叉树

6. 栈底至栈顶依次存放元素 A 、 B 、 C 、 D ，在第五个元素 E 入栈前，栈中元素可以出栈，则出栈序列可能是 \_\_\_\_\_\_ 。

A.ABCED B.DBCEA C.CDABE D.DCBEA

7. 下列关于队列的叙述中正确的是 \_\_\_\_\_\_ 。

A. 在队列中只能插入数据 B. 在队列中只能删除数据

C. 队列是先进先出的线性表 D. 队列是先进后出的线性表

8. 栈和队列的共同点是 \_\_\_\_\_\_ 。

A. 都是先进后出 B. 都是先进先出

C. 只允许在端点处插入和删除元素 D. 没有共同点

9 . 线性表的顺序存储结构和线性表的链式存储结构分别是 \_\_\_\_\_\_ 。

A. 顺序存取的存储结构、顺序存取的存储结构

B. 顺序存取的存储结构、随机存取的存储结构

C. 随机存取的存储结构、随机存取的存储结构

D. 任意存取的存储结构、任意存取的存储结构

10. 在单链表中，增加头结点的目的是 \_\_\_\_\_\_ 。

A. 方便运算的实现 B. 使单链表至少有一个结点

C. 标识表结点中首结点的位置 D. 说明单链表是线性表的链式存储实现

11. 用链表表示线性表的优点是 \_\_\_\_\_\_ 。

A. 便于插入和删除操作 B. 数据元素的物理顺序与逻辑顺序相同

C. 花费的存储空间较顺序存储少 D. 便于随机存取

12. 对长度为 N 的线性表进行顺序查找，在最坏情况下所需要的比较次数为 \_\_\_\_\_\_ 。

A.N+1 B.N C.(N+1)/2 D.N/2

13. 在深度为 5 的满二叉树中，叶子结点的个数为 \_\_\_\_\_\_ 。

A.32 B.31 C.16 D.15

14. 设一棵完全二叉树共有 699 个结点，则在该二叉树中的叶子结点数为 \_\_\_\_\_\_ 。

A.349 B.350 C.255 D.351

15. 已知二叉树后序遍历序列是 dabec ，中序遍历序列是 debac ，它的前序遍历序列是 \_\_\_\_\_\_ 。

A.cedba B.acbed C.decab D.deabc

16. 在下列几种排序方法中，要求内存量最大的是 \_\_\_\_\_\_ 。

A. 插入排序 B. 选择排序 C. 快速排序 D. 归并排序

17. 已知数据表 A 中每个元素距其最终位置不远，为节省时间，应采用的算法是 \_\_\_\_\_\_ 。

A. 堆排序 B. 直接插入排序 C. 快速排序 D. 直接选择排序

18. 下面不属于软件工程的 3 个要素的是 \_\_\_\_\_\_ 。

A. 工具 B. 过程 C. 方法 D. 环境

19. 在软件生命周期中，能准确地确定软件系统必须做什么和必须具备哪些功能的阶段是 \_\_\_\_\_\_ 。

A. 概要设计 B. 详细设计 C. 可行性分析 D. 需求分析

20. 软件需求分析阶段的工作，可以分为四个方面：需求获取、需求分析、编写需求规格说明书以及 \_\_\_\_\_\_ 。

A. 阶段性报告 B. 需求评审 C. 总结 D. 都不正确

21. 软件设计包括软件的结构、数据、接口和过程设计，其中软件的过程设计是指 \_\_\_\_\_\_ 。

A. 模块间的关系 B. 系统结构部件转换成软件的过程描述

C. 软件层次结构 D. 软件开发过程

22. 下列工具中属于需求分析常用工具的是 \_\_\_\_\_\_ 。

A.PAD B.PFD C.N-S D.DFD

23. 为了避免流程图在描述程序逻辑时的灵活性，提出了用方框图来代替传统的程序流程图，通常也把这种图称为 \_\_\_\_\_\_ 。

A.PAD 图 B.N-S 图 C. 结构图 D. 数据流图

24. 下面不属于软件设计原则的是 \_\_\_\_\_\_ 。

A. 抽象 B. 模块化 C. 自底向上 D. 信息隐蔽

25. 信息隐蔽的概念与下述哪一种概念直接相关 \_\_\_\_\_\_ 。

A. 软件结构定义 B. 模块独立性 C. 模块类型划分 D. 模拟耦合度

26. 下列叙述中，不属于软件需求规格说明书的作用的是 \_\_\_\_\_\_ 。

A. 便于用户、开发人员进行理解和交流

B. 反映出用户问题的结构，可以作为软件开发工作的基础和依据

C. 作为确认测试和验收的依据

D. 便于开发人员进行需求分析

27. 检查软件产品是否符合需求定义的过程称为 \_\_\_\_\_\_ 。

A. 确认测试 B. 集成测试 C. 验证测试 D. 验收测试

28. 在软件工程中，白盒测试法可用于测试程序的内部结构。此方法将程序看做是 \_\_\_\_\_\_ 。

A. 循环的集合 B. 地址的集合 C. 路径的集合 D. 目标的集合

29. 下列不属于软件调试技术的是 \_\_\_\_\_\_ 。

A. 强行排错法 B. 集成测试法 C. 回溯法 D. 原因排除法

30. 下列有关数据库的描述，正确的是 \_\_\_\_\_\_ 。

A. 数据库是一个 DBF 文件 B. 数据库是一个关系

C. 数据库是一个结构化的数据集合 D. 数据库是一组文件

31. 下列叙述中正确的是 \_\_\_\_\_\_ 。

A. 数据库是一个独立的系统，不需要操作系统的支持

B. 数据库设计是指设计数据库管理系统

C. 数据库技术的根本目标是要解决数据共享的问题

D. 数据库系统中，数据的物理结构必须与逻辑结构一致

32. 下述关于数据库系统的叙述中正确的是 \_\_\_\_\_\_ 。

A. 数据库系统减少了数据冗余

B. 数据库系统避免了一切冗余

C. 数据库系统中数据的一致性是指数据类型的一致

D. 数据库系统比文件系统能管理更多的数据

33. 在数据管理技术发展过程中，文件系统与数据库系统的主要区别是数据库系统具有 \_\_\_\_\_\_ 。

A. 数据无冗余 B. 数据可共享

C. 专门的数据管理软件 D. 特定的数据模型

34. 下列模式中，能够给出数据库物理存储结构与物理存取方法的是 \_\_\_\_\_\_ 。

A. 内模式 B. 外模式 C. 概念模式 D. 逻辑模式

35. 索引属于 \_\_\_\_\_\_ 。

A. 模式 B. 内模式 C. 外模式 D. 概念模式

36. 下列说法中，不属于数据模型所描述的内容的是 \_\_\_\_\_\_ 。

A. 数据结构 B. 数据操作 C. 数据查询 D. 数据约束

37. 数据库设计包括两个方面的设计内容，它们是 \_\_\_\_\_\_ 。

A. 概念设计和逻辑设计 B. 模式设计和内模式设计

C. 内模式设计和物理设计 D. 结构特性设计和行为特性设计

38. 单个用户使用的数据视图的描述称为 \_\_\_\_\_\_ 。

A. 外模式 B. 概念模式 C. 内模式 D. 存储模式

39 用树形结构来表示实体之间联系的模型称为 \_\_\_\_\_\_ 。

A. 关系模型 B. 层次模型 C. 网状模型 D. 数据模型

40. 将 E-R 图转换到关系模式时，实体与联系都可以表示成 \_\_\_\_\_\_ 。

A. 属性 B. 关系 C. 键 D. 域

41. 在关系数据库中，用来表示实体之间联系的是 \_\_\_\_\_\_ 。

A. 树结构 B. 网结构 C. 线性表 D. 二维表

42. 关系表中的每一横行称为一个 \_\_\_\_\_\_ 。

A. 元组 B. 字段 C. 属性 D. 码

43. 关系数据库管理系统能实现的专门关系运算包括 \_\_\_\_\_\_ 。

A. 排序、索引、统计 B. 选择、投影、连接

C. 关联、更新、排序 D. 显示、打印、制表

44. 按条件 f 对关系 R 进行选择，其关系代数表达式为 \_\_\_\_\_\_ 。

A.R|X|R B. R|X|Rf C. б f ® D. ∏ f ®

45. 数据库概念设计的过程中，视图设计一般有三种设计次序，以下各项中不对的是 \_\_\_\_\_\_ 。

A. 自顶向下 B. 由底向上 C. 由内向外 D. 由整体到局部

46.SQL 语言又称为 \_\_\_\_\_\_ 。

A. 结构化定义语言 B. 结构化控制语言

C. 结构化查询语言 D. 结构化操纵语言

（三）、填空题 ( 1 至 10 题为 数据结构与算法部分 、 11 至 19 题为 软件工程基础部分、 20 至 29 题为 数据库设计基础部分 )

1. 算法的基本特征是可行性、确定性、 \_\_\_\_\_\_ 和拥有足够的情报。

2. 实现算法所需的存储单元多少和算法的工作量大小分别称为算法的 \_\_\_\_\_\_ 。

3. 数据结构包括数据的逻辑结构、数据的 \_\_\_\_\_\_ 以及对数据的操作运算。

4. 数据的逻辑结构在计算机存储空间中的存放形式称为数据的 \_\_\_\_\_\_ 。

5. 栈的基本运算有三种：入栈、退栈和 \_\_\_\_\_\_ 。

6. 顺序存储方法是把逻辑上相邻的结点存储在物理位置 \_\_\_\_\_\_ 的存储单元中。

7. 在一棵二叉树上第 5 层的结点数最多是 \_\_\_\_\_\_ 。

8. 在先左后右的原则下，根据访问根结点的次序，二叉树的遍历可以分为三种：前序遍历、 \_\_\_\_\_\_ 遍历和后序遍历。

9. 在最坏情况下，冒泡排序的时间复杂度为 \_\_\_\_\_\_ 。

10. 在最坏情况下，堆排序需要比较的次数为 \_\_\_\_\_\_ 。

11. 软件工程研究的内容主要包括： \_\_\_\_\_\_ 技术和软件工程管理。

12. 软件的需求分析阶段的工作，可以概括为四个方面： \_\_\_\_\_\_ 、需求分析、编写需求规格说明书和需求评审。

13. 与结构化需求分析方法相对应的是 \_\_\_\_\_\_ 方法。

14. 数据字典是各类数据描述的集合，它通常包括 5 个部分，即数据项、数据结构、数据流、 \_\_\_\_\_\_ 和处理过程。

15. 数据流的类型有 \_\_\_\_\_\_ 和事务型。

16. 若按功能划分，软件测试的方法通常分为白盒测试方法和 \_\_\_\_\_\_ 测试方法。

17. 测试的目的是暴露错误，评价程序的可靠性；而 \_\_\_\_\_\_ 的目的是发现错误的位置并改正错误。

18. 软件的调试方法主要有：强行排错法、 \_\_\_\_\_\_ 和原因排除法。

19. 软件维护活动包括以下几类：改正性维护、适应性维护、 \_\_\_\_\_\_ 维护和预防性维护。

20. 数据库系统中实现各种数据管理功能的核心软件称为 \_\_\_\_\_\_ 。

21. 数据库系统的三级模式分别为 \_\_\_\_\_\_ 模式、内部级模式与外部级模式。

22. 数据库管理系统常见的数据模型有层次模型、网状模型和 \_\_\_\_\_\_ 三种。

23. 如果一个工人可管理多个设施，而一个设施只被一个工人管理，则实体 " 工人 " 与实体 " 设备 " 之间存在 \_\_\_\_\_\_ 联系。

24. 关系数据库管理系统能实现的专门关系运算包括选择、连接和 \_\_\_\_\_\_ 。

25. 关系模型的完整性规则是对关系的某种约束条件，包括实体完整性、 \_\_\_\_\_\_ 和自定义完整性。

26. 关系模型的数据操纵即是建立在关系上的数据操纵，一般有 \_\_\_\_\_\_ 、增加、删除和修改四种操作。

27. 数据库保护分为：安全性控制 、 \_\_\_\_\_\_ 、并发性控制和数据的恢复。

28.\_\_\_\_\_\_ 是数据库应用的核心。

29 数据库设计分为以下 6 个设计阶段：需求分析阶段、 \_\_\_\_\_\_ 、逻辑设计阶段、物理设计阶段、实施阶段、运行和维护阶段。

四、习题答案

（一）、是非题

1 . B 2.A 3.A 4.B 5.B 6.A 7. A 8.B 9.B 10.A

（二）、选择题

1.C 2.C 3.D 4.C 5.D 6.D 7.C 8.D 9.B 10.A

11.A 12.B 13.C 14.B 15.A 16.D 17.B 18.D 19.D 20.B

21.B 22.D 23.B 24.C 25.B 26.D 27.A 28.C 29.B 30.C

31.C 32.A 33.D 34.A 35.B 36.C 37.A 38.A 39.B 40.B

41.D 42.A 43.B 44.C 45.D 46.C

（三）、填空题

1. 有穷性 2. 空间复杂度和时间复杂度

3. 存储结构 4. 存储结构（或物理结构）

5. 读栈顶元素 6. 相邻

7.16 8. 中序

9.n(n-1)/2 10.O(nlog2n)

11. 软件开发 12. 需求获取

13. 结构化设计 14. 数据存储

15. 变换型 16. 黑盒

17. 调试 18. 回溯法

19. 完善性 20. 数据库管理系统（或 DBMS ）

21. 概念 22. 关系模型

23. 一对多（或 1 ： N 或 1:n ） 24. 投影

25. 参照完整性 26. 查询

27. 完整性控制 28. 数据库设计

29. 概念设计阶段

（三）实验

1 、改错题

第一处： p=(NODE \*)malloc(sizeof(NODE));

第二处： p=p->next;

2 、填空题

（ 1 ）【 1 】： x 【 2 】： p 【 3 】： q->next=s;

（ 2 ）【 1 】： q 【 2 】： next 【 3 】： q->next

3 、编程题

参考代码：

void fun(NODE \*h)

{ NODE \*p, \*q, \*r;

p = h->next; /\*p 指向第一个结点 \*/

if (p==NULL) return; /\* 若是空表，则返回 \*/

q = p->next; /\*q 指向第二个结点 \*/

p->next = NULL; /\* 设置第一个结点的链接指针为 NULL ，使其成为新尾结点 \*/

while (q) /\* 从第二个结点开始，依次取出每个结点，按顺序插入到新尾结点之前 \*/

{ r = q->next; /\* r 指向 q 的后继结点 \*/

q->next = p; /\*q 结点插入到 p 结点之前 \*/

p = q;

q = r; } /\*q 指向下一个结点，继续取下一个结点 \*/

h->next = p; /\* 生成的新链表链接到头结点的尾部 \*/

}

金榜VIP已享免费阅读及下载

打开百度APP阅读全文

立即领取

VIP教育大礼包

热门小说免费读

本文配套内容

含${item.docNum}篇文档

${item.title}

￥**${item.price}**

立即购买

查看文集

### 精品课程

* ${item.title}
* 免费 ￥${item.price}￥${item.oriPrice} ￥${item.oriPrice} ${item.orgName}
* ${item.videoCount}课节

相关推荐文档

* ${searchSpecial.title}
* ${v.docTitle}
* *推荐* *热门* *好评*
* 用App查看
* 打开百度APP

返回百度搜索

下载原文档，方便随时阅读

下载文档

## 2亿文档资料库

涵盖各行课件、资料、模板、题库、报告等

## 多种记录存储好工具

提供图转文字、拍照翻译、语音速记等

## APP端内容永久保存

随时阅读，多端同步

立即下载

看视频广告，获取20元代金券礼包

看视频，立领券 视频大小约3.7M

您是老用户，送您2张代金券

* 5元
* 适用除连续包月外的其他VIP
* 24小时内有效
* 10元
* 限百度文库VIP-12个月适用
* 24小时内有效

领取优惠券

您已成功领取老用户福利

已转存到百度网盘

存储在文件夹【来自：百度文库】

去看看

文库新人专享礼包

限时免费

价值¥500+

去文库APP免费领