《C 程序设计》习题集

（2021-2022 学年第 1 学期）

供 2021 级黄炜老师班级使用

2021 年 6 月 29 日

目　录

第 1 章　课程介绍 ……1
第 2 章　C 语言简介……2
第 3 章　数据类型 ……6
第 4 章　数据类型 ……14
第 5 章　操作符、表达式和语句 (1)……24
第 6 章　操作符、表达式和语句 (2)……29
第 7 章　控制语句：循环 ……37
第 8 章　控制语句：分支和跳转 ……49
第 9 章　字符串输入输出与输入验证 ……74
第 10 章　函数 (1) ……75
第 11 章　函数 (2) ……94
第 12 章　数组与指针 (1) ……99
第 13 章　数组与指针 (2) ……119
第 14 章　数组与指针 (3) ……126
第 15 章　字符串与字符串函数 ……129
第 16 章　存储类，链接与内存管理 ……149
第 17 章　文件输入输出 ……151
第 18 章　结构体与其他数据格式 (1)……165
第 19 章　结构体与其他数据格式 (2)……191
第 20 章　位操作 ……203
第 21 章　C 预处理器与 C 库 ……213
第 22 章　高级数据表示 ……225
第 23 章　复习课 (1) ……238
第 24 章　复习课 (2) ……249

第1章 课程介绍

1. 被誉为“C 语言之父”的是【 】

A. 谭浩强　　B. Bill Gates　　C. Ken Thompson　D. Dennis Ritchie

2. 以下不属于 C 语言正式标准的是【 】

A. C89　　B. C99　　C. C11　　D. C14

3. 请按顺序默写简单的 C 语言程序。

```
int main()
{
    return 0;       // normally exit
}
```

第2章 C 语言简介

4. C 语言规定，必须用【 】作为主函数名。

A. function

B. include

C. main

D. stdio

5. 一个 C 程序可以包含任意多个不同名的函数，但有且仅有一个【 】，一个 C 程序总是从它开始执行。

A. 过程

B. 主函数

C. 函数

D. include

6. 【 】是 C 程序的基本构成单位。

A. 函数

B. 函数和过程

C. 超文本过程

D. 子程序

7. 下列说法正确的是【 】。

A. 一个函数的函数体必须要有变量定义和执行部分，二者缺一不可

B. 一个函数的函数体必须要有执行部分，可以没有变量定义

C. 一个函数的函数体可以没有变量定义和执行部分，函数可以是空函数

D. 以上都不对

8. 下列说法正确的是 main() 函数【 】。

A. 必须放在 C 程序的开头

B. 必须放在 C 程序的最后

C. 可以放在 C 程序的中间部分，但在执行 C 程序时是从程序开头执行的

D. 可以放在 C 程序的中间部分，但在执行 C 程序时是从 main() 函数开始的

9. 下列说法正确的是【 】。

A. 在执行 C 程序时不是从 main() 函数开始的

B. 程序书写格式严格限制，一行内必须写一个语句

C. 程序书写格式自由，一个语句可以分写在多行上

D. 程序书写格式严格限制，一行内必须写一个语句，并要有行号

10. 在 C 语言中，每个语句和数据定义以【 】结束。

A. 句号

B. 逗号

C. 分号

D. 括号

11. 下列字符串是标识符的是【 】。

A. _HJ

B. 9_student

C. long

D. LINE 1

12. 以下说法正确的是【 】。

A. 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. 语言程序总是从 main() 函数开始执行

D. 语言程序中的 main() 函数必须放在程序的开始部分

13. 下列四组选项中，均不是 C 语言关键字的选项是【 】。

A. define; If; type

B. getc; char; printf

C. include; scanf; case

D. while; go; pow

14. 一个 C 语言程序是由【 】。

A. 一个主程序和若干个子程序组成

B. 函数组成，必须且只有一个主函数

C. 若干过程组成

D. 若干子程序组成

15. 以下不正确的 C 语言标识符是【 】。

A. BC

B. %c

C. _bc

D. b_c

16. 一个 C 程序的执行是从【 】。

A. main()函数开始，直到 main()函数结束

B. 第一个函数开始，直到最后一个函数结束

C. 第一个语句开始，直到最后一个语句结束

D. main()函数开始，直到最后一个函数结束

17. C 语言程序的基本单位是【 】。

A. 过程

B. 函数

C. 子程序

D. 标识符

18. 编辑程序的功能是【 】。

A. 建立并修改程序

B. 将源程序编译成目标程序

C. 调试程序

D. 命令计算机执行指定的操作

19. 一个函数由两部分组成，它们是【 】和【 】。

20. 函数体的范围是【 】。

21. C 语言是通过【 】来进行输入输出的。

22. C 语言中，在一个标识符后紧跟着一对圆括弧，表明它是一个【 】。

23. C 语言符号集包括【 】。

24. 一个 C 程序有且仅有一个【 】函数。

25. C 程序的基本单位是【 】。

26. 一个 C 程序有【 】个 main()函数和【 】个其他函数。

27. 结构化设计中的三种基本结构是【 】。

28. C 语言输入操作由库函数【 】完成，输出函数由库函数【 】完成。

29. 请按顺序（从主到次）默写简单的 C 语言程序。

```
30. #include <stdio.h>
31. int main()
32. {
33.     printf("Hello, world.\n");
34.     return 0;       // normally exit
35. }
```

第3章 数据类型

36. 设 char ch; 以下正确的赋值语句是【 】。

A. ch='123';

B. ch='\xff';

C. ch='\08';

D. ch="\";

37. C 语言标识符只由字母、数字和下划线三种字符组成，且第一个字符【 】。

A. 必须为字母

B. 必须为下划线

C. 必须为字母或下划线

D. 可以是字母、数字和下划线中任一种字符

38. 下面四个选项中，均是合法整型常量的选项是【 】。

A. 160; -0xffff; 011

B. -0xcdf; 01a; 0xe

C. -01; 986,012; 0668

D. -0x48a; 2e5; 0x

39. 下面四个选项中，均是合法浮点数的选项是【 】。

A. 160.;0.12;e3

B. 123;2e4.2;.e5

C. -.18;123e4;0.0

D. -e3;0.234;1e3

40. 下面四个选项中，均是合法浮点数的选项是【 】。

A. +1e+1;5e-9.4;03e2

B. -.60;12e-4;-8e5

C. 123e;12e-.4;+2e-1

D. -e3;.8e-4;5.e-0

41. 下面四个选项中，均是不合法转义字符的选项是【 】。

A. '\"'; '\\'; '\xf'

B. '\1011'; '\'; '\A'

C. '\011'; '\f'; '\}'

D. '\abc'; '\101'; 'x1f'

42. 下面四个选项中，均是不正确的八进制数或十六进制数的选项是【 】。

A. 016;0x8f;018

B. 0abc;017;0xa

C. 010;-0x11;0x16

D. 0a12;7ff;-123

43. 下面四个选项中，均是正确的八进制数或十六进制数的选项是【 】。

A. -10;0x8f;-011

B. 0abc;-017;0xc

C. 0010;-0x11;0xf1

D. 0a12;-0x13;-0xa

44. 下面四个选项中，均是正确的数值常量或字符常量的选项是【 】。

A. 0.0;0x8f;8.9e;'&'

B. "a";3.9E-2.5;1e1;'\"'

C. '3';011;0xFF00;0a

D. +001;0xabcd;2e2;50.

45. 下面不正确的字符串常量是【 】。

A. 'abc'

B. "12'12"

C. "0"

D. " "

46. 在程序中可以用来作为变量名的合法的标识符是【 】。

A. static

B. 23_b1

C. stu_1t

D. \#33

47. 以下正确的叙述是【 】。

A. 在 C 程序中，每行中只能写一条语句

B. 若 a 是实型变量，C 程序中允许赋值 a=10 ，因此实型变量中允许存放整型数

C. 在 C 程序中，无论是整数还是实数，都能被准确无误的表示

D. 在 C 程序中，% 是只能用于整数运算的运算符

48. 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'-'3'; 后，c2 的值为【 】。

A. D

B. 68

C. 不确定的值

D. 'C'

49. 若有说明语句 char c='\72'; 则变量 c【 】。

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，c 的值不确定

50. 下面四个选项中，均是非法常量的选项是【 】。

A. 'as' -0fff '\0xa'

B. '\\' '\01 12,456

C. -0x18 01177 0xf

D. 0xabc '\0' "a"

51. 在 C 语言中，int、char 和 short 三种类型数据在内存中所占用的字节数【 】。

A. 由用户自己定义

B. 均为 2 个字节

C. 是任意的

D. 有所用的机器的机器字长决定

52. 设 C 语言中，若某种机器上一个 int 型数据类型在内存中占 2 个字节，则 unsigned int 型数据的取值范围为【 】。

A. 0~255

B. 0~32767

C. 0~65535

D. 0~2147483647

53. 语句 char ch='\72'，则：

A. ch 包含三个字符

B. ch 是一个字符串

C. ch 包含一个字符

54. D.程序错误

55. 以下能正确的定义整型变量 a，b，c 并为其赋初值 5 的语句是【 】。

A. int a=b=c=5;

B. int a,b,c=5;

C. int a=5,b=5,c=5;

D. int a=b=c=5;

56. 已知 ch 是是字符形变量，下面不正确的赋值语句是【 】。

A. ch='a+b';

B. ch='o';

C. ch='7'+'9';

D. ch=5+9;

57. 已知 ch 是是字符型变量，下面正确的赋值语句是【 】。

A. ch='123';

B. ch='\xff';

C. ch='\08';

D. ch="\";

58. 已知 unsigned char ch=-125;，下面产生溢出语句是【 】。

A. ch=256;

B. ch+='\xff';

C. ch-=3;

D. ch+=256;

59. 在 C 语言中（以 32 位 PC 机为例），一个 char 型数据在内存中所占用的字节数为【 】；int 型为【 】；float 型为【 】；double 型为【 】。

60. 在 C 语言中的实型变量分为两种类型，他们是【 】和【 】。

61. 若有定义：char c='\010'; 则变量 c 中包含的字符个数为【 】。

62. C 语言中的标识符只能由三种字符组成，它们是【 】，【 】和【 】。

63. 下列属于 C 关键字的是： main，integer，global，printf，char，key，*

64. 考虑下面的程序，第 7 行之后的程序状态（各变量的值）如何？第 8 行？第 9 行？

```
#include <stdio.h>
int main(void)
{
    int x, y;
    x = 10;
    y = 5; /* line 7 */
    y = x + y; /* line 8 */
    x = x * y; /* line 9 */
    printf("%d %d\n", x, y);
    return 0;
}
```

76. 在 C 语言中，一个 int 型数据在内存中占用 2 个字节，则 int 型数据的取值范围为【 】

77. float 和 double 类型的精度是多少？请说明精度丢失的根本原因。

78. 请将下列不同进制数相互转换：（保留必要的小数位数）

a. 将二进制 101011.0010 转为十进制数；

b. 将十进制数 65737.34 转为十六进制数；

c. 将八进制数 245 转为十进制数。

79. 设变量 val 在内存长度为 8 个比特分别为 10010100，请说明 val 声明为 char 和 unsigned char 类型时，其值分别为多少？

80. Indiana Sloth 编写了以下程序，请你修改后写出正确的版本，尽量接近原始版本。

```
81. include <stdio.h>
82.  int main{void} /* this prints the number of weeks in a year /*
83.  (
84.  int s
85.  s := 56;
86.  print(There are s weeks in a year.);
87. return 0;
```

88. Dottie Cawm 编写了一个满是错误的程序。请帮助她找出错误，请你修改后写出正确的版本，尽量接近原始版本。

```
89. include <stdio.h>
90. main
91. (
92. float g; h;
93. float tax, rate;
94. scanf("%f", h);
95. g = e21;
96. tax = rate*g;
97. )
```

98. 默写位移公式程序。

```
/*********************************************************
 * Input the initial velocity, acceleration and time,
 * calculates the shift, by the equation
 *   s = v0*t + 1.0 / 2 * a*t*t;
 * Author:  Wei Huang <whuang@xmu.edu.cn>
 * Version: 1.0 [Jun. 11, 2014]
 *********************************************************/
#include <stdio.h> /* Standard I/O header */

int main(void)
{
   double v0 = 0; //initial velocity
   double a = 0;  //acceleration
   double t = 0;  //time
   double s = 0;  //shift
   printf("Enter the Initial Velocity (m/s): ");
   scanf("%lf", &v0);
   printf("Enter the Acceleration (m/(s*s)): ");
   scanf("%lf", &a);
   printf("Enter the Time (s): ");
   scanf("%lf", &t);
   if (t < 0) //time cannot be negative
   {
      printf("The time cannot be negative.\n");
      return -1; //error code -1: time<0
   }
   s = v0*t + 1.0 / 2 * a*t*t; //main equation
   printf("The total shift is %.3lf m.\n", s);
   return 0; //return code 0: no error
}
```

第4章 数据类型

99. 设 char ch; 以下正确的赋值语句是【 】。

A. ch='123';

B. ch='\xff';

C. ch='\08';

D. ch="\";

100. C 语言标识符只由字母、数字和下划线三种字符组成，且第一个字符【 】。

A. 必须为字母

B. 必须为下划线

C. 必须为字母或下划线

D. 可以是字母、数字和下划线中任一种字符

101. 下面四个选项中，均是合法整型常量的选项是【 】。

A. 160; -0xffff; 011

B. -0xcdf; 01a; 0xe

C. -01; 986,012; 0668

D. -0x48a; 2e5; 0x

102. 下面四个选项中，均是合法浮点数的选项是【 】。

A. 160.;0.12;e3

B. 123;2e4.2;.e5

C. -.18;123e4;0.0

D. -e3;0.234;1e3

103. 下面四个选项中，均是合法浮点数的选项是【 】。

A. +1e+1;5e-9.4;03e2

B. -.60;12e-4;-8e5

C. 123e;12e-.4;+2e-1

D. -e3;.8e-4;5.e-0

104. 下面四个选项中，均是不合法转义字符的选项是【 】。

A. '\"'; '\\'; '\xf'

B. '\1011'; '\'; '\A'

C. '\011'; '\f'; '\}'

D. '\abc'; '\101'; 'x1f'

105. 下面四个选项中，均是不正确的八进制数或十六进制数的选项是【 】。

A. 016;0x8f;018

B. 0abc;017;0xa

C. 010;-0x11;0x16

D. 0a12;7ff;-123

106. 下面四个选项中，均是正确的八进制数或十六进制数的选项是【 】。

A. -10;0x8f;-011

B. 0abc;-017;0xc

C. 0010;-0x11;0xf1

D. 0a12;-0x13;-0xa

107. 下面四个选项中，均是正确的数值常量或字符常量的选项是【 】。

A. 0.0;0x8f;8.9e;'&'

B. "a";3.9E-2.5;1e1;'\"'

C. '3';011;0xFF00;0a

D. +001;0xabcd;2e2;50.

108. 下面不正确的字符串常量是【 】。

A. 'abc'

B. "12'12"

C. "0"

D. " "

109. 在程序中可以用来作为变量名的合法的标识符是【 】。

A. static

B. 23_b1

C. stu_1t

D. \#33

110. 以下正确的叙述是【 】。

A. 在 C 程序中，每行中只能写一条语句

B. 若 a 是实型变量，C 程序中允许赋值 a=10 ，因此实型变量中允许存放整型数

C. 在 C 程序中，无论是整数还是实数，都能被准确无误的表示

D. 在 C 程序中，% 是只能用于整数运算的运算符

111. 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'-'3'; 后，c2 的值为【 】。

A. D

B. 68

C. 不确定的值

D. 'C'

112. 若有说明语句 char c='\72'; 则变量 c【 】。

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，c 的值不确定

113. 下面四个选项中，均是非法常量的选项是【 】。

A. 'as' -0fff '\0xa'

B. '\\' '\01 12,456

C. -0x18 01177 0xf

D. 0xabc '\0' "a"

114. 在 C 语言中，int、char 和 short 三种类型数据在内存中所占用的字节数【 】。

A. 由用户自己定义

B. 均为 2 个字节

C. 是任意的

D. 有所用的机器的机器字长决定

115. 设 C 语言中，若某种机器上一个 int 型数据类型在内存中占 2 个字节，则 unsigned int 型数据的取值范围为【 】。

A. 0~255

B. 0~32767

C. 0~65535

D. 0~2147483647

116. 语句 char ch='\72'，则：

A. ch 包含三个字符

B. ch 是一个字符串

C. ch 包含一个字符

117. D.程序错误

118. 以下能正确的定义整型变量 a，b，c 并为其赋初值 5 的语句是【 】。

A. int a=b=c=5;

B. int a,b,c=5;

C. int a=5,b=5,c=5;

D. int a=b=c=5;

119. 已知 ch 是是字符形变量，下面不正确的赋值语句是【 】。

A. ch='a+b';

B. ch='o';

C. ch='7'+'9';

D. ch=5+9;

120. 已知 ch 是是字符型变量，下面正确的赋值语句是【 】。

A. ch='123';

B. ch='\xff';

C. ch='\08';

D. ch="\";

121. 已知 unsigned char ch=-125;，下面产生溢出语句是【 】。

A. ch=256;

B. ch+='\xff';

C. ch-=3;

D. ch+=256;

122. 在 C 语言中（以 32 位 PC 机为例），一个 char 型数据在内存中所占用的字节数为【 】；int 型为【 】；float 型为【 】；double 型为【 】。

123. 在 C 语言中的实型变量分为两种类型，他们是【 】和【 】。

124. 若有定义：char c='\010'; 则变量 c 中包含的字符个数为【 】。

125. C 语言中的标识符只能由三种字符组成，它们是【 】，【 】和【 】。

126. 下列属于 C 关键字的是：【 】main，integer，global，printf，char，key，*

127. 考虑下面的程序，第 7 行之后的程序状态（各变量的值）如何？第 8 行？第 9 行？

```
#include <stdio.h>

int main(void)

{

    int x, y;

    x = 10;

    y = 5; /* line 7 */

    y = x + y; /* line 8 */

    x = x * y; /* line 9 */

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

    return 0;

}
```

128. 在 C 语言中，一个 int 型数据在内存中占用 2 个字节，则 int 型数据的取值范围为【 】

129. float 和 double 类型的精度是多少？请说明精度丢失的根本原因。

130. 请将下列不同进制数相互转换：（保留必要的小数位数）

a. 将二进制 101011.0010 转为十进制数；

b. 将十进制数 65737.34 转为十六进制数；

c. 将八进制数 245 转为十进制数。

131. 设变量 val 在内存长度为 8 个比特分别为 10010100，请说明 val 声明为 char 和 unsigned char 类型时，其值分别为多少？

132. Indiana Sloth 编写了以下程序，请你修改后写出正确的版本，尽量接近原始版本。

```
include <stdio.h>

int main{void} /* this prints the number of weeks in a year /*

(

int s

s := 56;

print(There are s weeks in a year.);

return 0;
```

133. Dottie Cawm 编写了一个满是错误的程序。请帮助她找出错误，请你修改后写出正确的版本，尽量接近原始版本。

include <stdio.h>

main

(

float g; h;

float tax, rate;

scanf("%f", h);

g = e21;

tax = rate*g;

)

134. 默写位移公式程序。

/***

* Input the initial velocity, acceleration and time,

* calculates the shift, by the equation

* s = v0*t + 1.0 / 2 * a*t*t;

```
* Author:  Wei Huang <whuang@xmu.edu.cn>

* Version: 1.0 [Jun. 11, 2014]

**********************************************************/

#include <stdio.h> /* Standard I/O header */



int main(void)

{

   double v0 = 0; //initial velocity

   double a = 0;  //acceleration

   double t = 0;  //time

   double s = 0;  //shift

   printf("Enter the Initial Velocity (m/s): ");

   scanf("%lf", &v0);

   printf("Enter the Acceleration (m/(s*s)): ");

   scanf("%lf", &a);
```

```
    printf("Enter the Time (s): ");

    scanf("%lf", &t);

    if (t < 0) //time cannot be negative

    {

        printf("The time cannot be negative.\n");

        return -1; //error code -1: time<0

    }

    s = v0*t + 1.0 / 2 * a*t*t; //main equation

    printf("The total shift is %.3lf m.\n", s);

    return 0; //return code 0: no error

}
```

第5章 操作符、表达式和语句 (1)

135. 若有以下定义 int a,b; float x;，则正确的赋值语句是【 】。

A. a=1,b=2;

B. b++;

C. a=b=5

D. b=int(x);

136. 设 x,y 均为 float 变量，则以下不合法的赋值语句是【 】

A. ++x;

B. y=(x%2)/10;

C. x*=y+8;

D. x=y=10;

137. 设 x,y,z 均为 int 变量，则执行语句 x=(y=(z=(10)+5)-5); 后， x,y,z 的值是【 】。

A. x=10 y=15 z=10

B. x=10 y=10 z=10

C. x=10 y=10 z=15

D. x=10 y=5 z=10

138. 设有说明：char w; int x; float y; double z; 则表达式 w*x+z-y 值的数据类型为【 】。

A. float

B. char

C. int

D. double

139. 逗号表达式 (a=3*5,a*4),a+15 的值为【 】，

A. 15

B. 60

C. 30

D. 不确定

140. 上题 a 的值为【 】。

A. 60

B. 30

C. 15

D. 90

141. 设 n=10，i=4 ，则赋值运算 n%=i+1 执行后，n 的值是【 】。

A. 0

B. 3

C. 2

D. 1

142. 若有代数式 3aebc ，则不正确的 C 语言表达式是【 】。

A. a/b/c*e*3

B. 3*a*e/b/c

C. 3*a*e/b*c

D. a*e/c/b*3

143. 在 C 语言中，要求运算数必须是整型的运算符是【 】。

A. /

B. ++

C. !=

D. %

144. 若以下变量均是整型，且 num=sum=7；则计算表达式 sum=num++,sum++,++num 后 sum 的值为【 】。

A. 7

B. 8

C. 9

D. 10

145. 若有定义： int a=7; float x=205 ,y=4.7; 则表达式 a%3*(int)(x+y)%2/4 的值是【 】。

A. 2.500000

B. 2.750000

C. 3.500000

D. 0.000000

146. 设 int n=3; 则 ++n 的结果是【 】。

A. 2

B. 3

C. 4

D. 5

147. 上题 n 的结果是【 】

A. 2

B. 3

C. 4

D. 5

148. 设 int n=3;则 n++的结果是【 】,

149. 上题 n 的结果是【 】。

A. 2

B. 3

C. 4

D. 5

150. 设变量 n 为 float 类型，m 为 int 类型，则以下能实现将 n 中的数值保留小数点后两位，第三位进行四舍五入运算的表达式是【 】。

A. n=(n*100+0.5)/100.0

B. m=n*100+0.5, n=m/100.0

C. n=n*100+0.5/100.0

D. n=(n/100+0.5)*100.0

151. 表达式 10/3 的结果是【 】，表达式 10%3 的结果是【 】

152. 设 int x; 当 x 的值分别为 1、2、3、4 时，表达式(x&1==1)?1:0 的值分别是【 】，【 】，【 】，【 】。

153. 已知字母 a 的 ASCII 码为十进制数 97，且设 ch 为字符型变量，则表达式 ch='a'+'8'-'3'、【 】。

154. 若有以下定义 int m=7,y=2;，则计算表达式 y+=y-=m*=y 后 y 值是【 】。

155. 若 s 是 int 型变量，且 s=6，则下面表达式 s%2+(s+1)%2 的值为【 】。

156. 若 a、b 和 c 均是 int 型变量，则计算表达式 a=(b=4)+(c=2)后，a 值为【 】，b 值为【 】，c 值为【 】。

157. 若 a 是 int 型变量，则计算表达式 a=25/3%3 后 a 的值为【 】

158. 若 x 和 n 均是 int 型变量，且 x 和 n 的初值均为 5，则计算表达式 x+=n++ 后 x 的值为【 】，n 的值为【 】。

159. 若有定义：int b=7; float a=2.5,c=4.7; e 则下面表达式 a+(int)(b/3*(int)(a+c)/2)%4 的值为【 】。

160. 若有定义：int a=2,b=3;float x=3.5,y=2.5;则下面表达式的值(a+b)/2+x/y 为【 】。

161. 若 x 和 n 均是 int 型变量，且 x 的初值为 12，n 的初值为 5，则计算表达式 x%=(n%=2)后 x 的值为【 】。

162. 假设所有变量均为整型，则表达式(a=2,b=5,a++,b++,a+b)的值为【 】。

163. 表达式 8/4*(int)2.5/(int)(1.25*(3.7+2.3)) 值的数据类型为【 】。

164. 表达式 pow(2.8,sqrt(double(x))) 值的数据类型为【 】。

165. 设 int x=9,y=8; 表达式 x==y+1 的结果是【 】。

166. 设 int a=1,b=2,c,d,e;执行 c=(-a++)+(++b); d=(b--)+(++a)-a; e=(a/(++b))-(b/(--a));后，变量 a、b、c、d、e 的结果分别是【 】【 】【 】【 】【 】。

第6章 操作符、表达式和语句 (2)

167. 以下使 i 的运算结果为 4 的表达式是【 】。

A. int i=0, j=0; (i=3, (j++)+i)

B. int i=1, j=0; j=i=((i=3)*2)

C. int i=0, j=1; (j==1)?(i=1):(i=3)

D. int i=1, j=1; i+=j+=2

168. sizeof(float)是【 】。

A. 一个双精度型表达式

B. 一个整型表达式

C. 一种函数调用

D. 一个不合法的表达式

169. 设有 int i; char c; float f;，以下结果为整数的表达式是【 】。

A. i+f

B. i*c

C. c+f

D. i+c+f

170. 已知各变量类型如下：int i=8,k,a,b; unsigned long w=5; double x=1.42,y=5.2;，则以下符合 C 语言语法的表达式是【 】。

A. a+=a-=(b=4)*(a=3)

B. a=a*3=2

C. x%(-3)

D. y=float (i)

171. 以下符合 C 语言语法的赋值表达式是【 】。

A. d=9+e+f=d+9

B. d=9+e=f=d+9

C. d=9+e,e++,d+9

D. d=9+e++=d+7

172. 设变量 a 是整型，f 是实型，i 是双精度型，则表达式 10+'a'+i*f 值的数据类型为【 】。

A. int

B. float

C. double

D. 不确定

173. 以下程序的执行结果是【 】。

```
#include <stdio.h>

main()

{

   int sum,pad;

   sum=pad=5;

   pad=sum++;

   pad++;

   ++pad;

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

}
```

A. 7

B. 6

C. 5

D. 4

174. 以下不正确的叙述是【 】。

A. 在 C 程序中，逗号运算符的优先级最低

B. 在 C 程序中，APH 和 aph 是两个不同的变量

C. 若 a 和 b 类型相同，计算表达式 a=b 后，b 的值放入 a 中，且 b 的值不变

D. 键盘输入时，对于整型变量只能输入整数，对于实型变量只能输入实数

175. 若有定义：int e=1,f=4,g=2;float m=10.5,n=4.0,k;则计算赋值表达式 k=(e+f)/g+sqrt((double)n)*1.2/g+m 后 k 的值是【 】。

176. 假设 m 是一个三位数，从左到右用 a、b、c 表示各位的数字，则从左到右各个数字是 bac 的三位数的表达式是【 】。

177. 程序片段 if (test = 0) val=21; else val=10; 与 if (test == 0) val=21; else val=10;运行结果有何不同，并简要解释。（书写在草稿纸上，经利用 GDB 调试或设计程序验证后将结果修正并誊写在答题纸上）

178. 请写出下列变量的值：（书写在草稿纸上，经 GCC 调试后将结果修正并誊写在答题纸上）

a. 设有整型变量 x、y，求它们在运行 y = 3 + 2*(x = 7/2);后的值。

b. 设有浮点型变量 x、y，求它们在运行 y = 3 + 2*(x = 7/2);后的值。

c. 设有整型变量 x，求它在运行语句 x = 3 / 5 * 22.0;后的值。

d. 求出 30 / 4 * 5、30 * 5 / 4、30 / 4.0 * 5、30 / 4 * 5.0 的值。

179. 请预测下列程序运行后的输出，并上机测试，并修正你的答案。若与你的设想不同，请解释该现象。

```
#include <stdio.h>

#define FORMAT "%s! C is cool!\n"

int main(void)

{

    int num = 10;

    printf(FORMAT,FORMAT);

    printf("%d\n", ++num);

    printf("%d\n", num++);

    printf("%d\n", num--);

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

    return 0;

}
```

180. 在声明并赋值 `unsigned int val = 4294967295;` 之后，执行 `val+=4;` 是否能得到 4294967299？如何修改类型符使之正确计算 4294967295+4。（书写在草稿纸上，经 GCC 调试后将结果修正并誊写在答题纸上）

181. 假设变量 x,y,z 为整型变量,m 为 long 变量,则在 32 位微型机上执行下面的赋直语句后,y 值为【 】,z 值为【 】,m 值为【 】。

y=(x=32767,x-1);

z=m=0xFFFF;

182. **选做** 运行 `x=10; u = (x++ * --x);` 之后 u 和 x 的值应为多少？为何程序员应避免书写这样的程序？

183. 找出错误，并将正确版本的程序写出来，尽量接近原意。（注：先用眼睛观察，逐条列出到草稿纸上，然后在GCC检查校对自己的答案，誊写到作业纸上。）

```
define B booboo

define X 10

main(int)

{

   int age;

   char name;

    printf("Please enter your first name.");

   scanf("%s", name);
```

```
        printf("All right, %c, what's your age?\n", name);

        scanf("%f", age);

        xp = age + X;

        printf("That's a %s! You must be at least %d.\n", B, xp);

        rerun 0;

    }
```

184. 编程计算下面函数，注意检查输入变量的合法性，并做出适当的处理。

$$f(x)=\{-x+2.5,0\leqslant x<22-1.5(x-3)2,2\leqslant x<4x-1.5,4\leqslant x<6$$

185. **.选做** 用牛顿迭代法（具体方法请在互联网上搜索）求下面方程在 1.5 附近的根。 $2x^3-4x^2+3x-6=0$

186. 请默写位移公式程序，注意每个标点符号。默写后，用 GCC 编译自行批改至正确再提交。

```
/*********************************************************

* Input the initial velocity, acceleration and time,

* calculates the shift, by the equation

*   s = v0*t + 1.0 / 2 * a*t*t;
```

```
* Author:  Wei Huang <whuang@xmu.edu.cn&gt;

* Version: 1.0 [Jun. 11, 2014]

**********************************************************/

#include <stdio.h> /* Standard I/O header */



int main(void)

{

   double v0 = 0; //initial velocity

   double a = 0;  //acceleration

   double t = 0;  //time

   double s = 0;  //shift

   printf("Enter the Initial Velocity (m/s): ");

   scanf("%lf", &v0);

   printf("Enter the Acceleration (m/(s*s)): ");

   scanf("%lf", &a);
```

```
    printf("Enter the Time (s): ");

    scanf("%lf", &t);

    if (t < 0)                    //time cannot be negative

        return -1;                //error code -1: time<0

    s = v0*t + 1.0 / 2 * a*t*t;     //main equation

    printf("The total shift is %.3lf m.\n", s);

    return 0;                     //return code 0: no error

}
```

第7章 控制语句：循环

说明 以下用字符↲表示键盘上的 Enter 键（对应字符为'\n'）。

187. 以下描述正确的是【 】

A. do {A} while (B); 循环中的复合语句 A 比 while (B) {A}循环中的多执行一次。

B. for (A; B; C) {D}循环中的各个部分均可以省略（其中{D}省略为分号;）。

C. for (A; B; C) {D}循环中的各个部分均可以为复合语句。

D. 在 do-while 循环体中，一定要有能使 while 后面表达式的值变为 0 的操作。

188. C 语言用【 】表示逻辑"真"值。

A. true

B. t 或 y

C. 非零整数值

D. 整数 0

189. 语句 while(!e); 中的条件!e 等价于【 】。

A. e==0

B. e!=1

C. e!=0

D. ~e

190. 以下 for 循环 for(x=0,y=0;(y!=123)&&(x<4);x++); 是【 】。

A. 无限循环

B. 循环次数不定

C. 执行 4 次

D. 执行 3 次

191. 下面有关 for 循环的正确描述是【 】。

A. for 循环只能用于循环次数已经确定的情况

B. for 循环是先执行循环体语句，后判定表达式

C. 在 for 循环中，不能用 break 语句跳出循环体

D. for 循环体语句中，可以包含多条语句，但要用花括号括起来

192. 下面程序段的运行结果是【 】

```
x=y=0;
while(x<15) y++, x+=++y;
printf("%d,%d",y,x);
```

A. 20,7

B. 6,12

C. 20,8

D. 8,20

193. 下面程序段的运行结果是【 】

```
int n=0;
while(n++<=2); printf("%d",n);
```

A. 2

B. 3

C. 4

D. 编译错误

194. 设以下有程序段，不考虑输出，描述正确的是【 】。

```
t=0;
while(printf("*"))
{t++;
    if(t<3) break;
}
```

A. 其中循环控制表达式与 0 等价

B. 其中循环控制表达式与'0'等价

C. 其中循环控制表达式是不合法的

D. 以上说法都不对

195. C 语言中 while 和 do-while 循环的主要区别是【 】。

A. do-while 的循环体至少无条件执行一次

B. do-while 的循环控制条件比 while 的循环控制条件严格

C. do-while 允许从外部转到循环体内

D. do-while 的循环体不能是复合语句

196. 以下能正确计算 1×2×3×…×10 的程序是【 】。

A. `do{i=1;s=1;s=s*i;i++;}while(i<=10);`

B. `do{i=1;s=0; s=s*i;i++;}while(i<=10);`

C. `i=1;s=1;do{s=s*i; i++;}while(i<=10);`

D. `i=1;s=0; do{s=s*i; i++; }while(i<=10);`

197. 以下描述正确的是【 】。

A. 由于 do-while 循环中循环体语句只能是一条可执行语句，所以循环体内不能使用复合语句。

B. do-while 循环由 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，至少执行一次循环中的语句体。

D. do-while 循环中，根据情况可以省略 while。

198. 若有如下语句 `int x=3; do {printf("%d\n",x-=2);} while(!(--x));`，则上面程序段【 】。

A. 输出的是 1

B. 输出的是 1 和-2

C. 输出的是 3 和 0

D. 是死循环

199. 等比数列的第一项 a=1，公比 q=2，下面程序的功能是求满足前 n 项和小于 100 的最大 n，请选择题空。

```
#include <stdio.h>
main()
{int a,q,n,sum;
a=1;q=2;n=sum=0;
do{【 】; ++n;a*=q;
}while (sum<100);
【 】;
printf("%d\n",n);}
```

A. sum++

B. sum+=a

C. sum*=a

D. a+=sum

200. 接上题，请填空。

A. n=n-2

B. n=n

C. n++

D. n-=1

201. 下面程序运行结果是【 】。

```
#include <stdio.h>
main()
{ int a=1,b=10;
    do
    { b-=a; a++; } while ( b--<0) ;
    printf ( "a=%d,b=%d\n",a,b) ;
}
```

A. a=3,b=11

B. a=2,b=8

C. a=1,b=-1

D. a=4,b=9

202. 下面有关 for 循环的正确描述是【 】。

A. for 循环只能用于循环次数已经确定的情况。

B. for 循环是先执行循环体语句，后判断表达式。

C. 在 for 循环中，不能用 break 语句跳出循环体。

D. for 循环的循环体语句中，可以包含多条语句，但必须用花括号括起来。

203. 若 i 为整型变量，则以下循环 `for(i=2;i==1;) printf("%d",i--);` 执行次数是【 】。

A. 无限次

B. 0 次

C. 1 次

D. 2 次

204. 下面程序段不是死循环的是【 】。

A. `int i=100; while(1) { i=i%100+1; if(i>100) break;}`

B. `for ( ; ; );`

C. `int k=0;do{++k; } while(k>=0);`

D. `int s=36; while(s); --s;`

205. 下面程序段的功能是计算 1000!的末尾含有多少个零。请选择填空。（提示：只要算出 1000!中含有因数 5 的个数即可）

```
for(k=0,i=5;i<=1000;i+=5)
{m=i;
while(【 】){k++; m=m/5;}
}
```

A. `m%5=0`

B. `m=m%5==0`

C. `m%5==0`

D. `m%5!=0`

206. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{int i,b,k=0;
    for(i=1;i <= 5;i++)
    {b=i%2;
        while(b-->=0) k++;
    }
    printf("%d,%d",k,b);
}
```

A. 3,-1

B. 8,-1

C. 3,0

D. 8,-2

207. 下面程序段【 】。

```
for(t=1;t<=100;t++)
{scanf ("%d",&x);
    if(x<0) continue;
    printf("%3d",t);
}
```

A. 当 x<0 时整个循环结束

B. x>=0 时什么也不输出

C. printf 函数永远也不执行

D. 最多允许输出 100 个非负整数

208. 下面程序的运行结果是【 】。

```
main()
{ int i,j,a=0;
for(i=0;i < 2;i++)
{ for (j=0; j < 4; j++)
{if (j%2) break;
a++;
}
a++;
}
```

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

A. 4

B. 5

C. 6

D. 7

209. 以下程序的输出结果是【 】。

```
#include<stdio.h>
main()
{
    int i;
    for(i=1;i <= 5;i++)
    {
        if(i%2)
            printf("*");
        else
            continue;
        printf("#");
    }
    printf("$\n");
}
```

A. *#*#*#$

B. #*#*#*$

C. *#*#$

D. #*#*$

210. 以下程序的输出结果是【 】。

```
#include<stdio.h>
main()
{
    int num=0;
    while(num<=2)
    {
    num++;
    printf("%d\n",num);
    }
```

```
}
```

A. 1↲2↲3↲4↲

B. 1↲2↲3↲

C. 1↲2↲

D. 1↲

211. 若 for 循环用以下形式表示：for（表达式 1；表达式 2；表达式 3） 循环体语句; 则执行语句 for(i=0;i<3;i++) printf("*"); 时，表达式 1 执行【 】次，表达式 3 执行【 】次。

212. 设有程序片段 for (value = 36; value > 0; value /= 2) printf("%3d", value);，当 value 分别为 double 和 int、char 类型讨论其输出结果。（书写在草稿纸上，经利用 GDB 调试或设计程序验证后将结果修正并誊写在答题纸上）

213. 阅读程序片段 for (i=0; i < slen; i++) if (str[i]==to_search) { pos=i; ____; }，请写出空白处填写 break、continue 和空白时的程序作用。（请设计程序，验证后将结果修正并誊写在答题纸上）

214. 程序片段 int x = 100; while (x++ < 103) printf("%4d\n",x); printf("%4d\n",x);，写出其输出结果。

215. 下列程序运行后的输出结果是【 】。

```
#include<stdio.h>
main()
{
    int i,j,k;
    for(i=1;i<=6;i++)
    {
    for(j=1;j<=20-3*j;j++)
    printf("%3d",k);
    for(k=i-1;k>0;k--)
    printf("%3d",k);
    printf("\n");
    }
}
```

216. 下面程序的功能是用公式 π26≈112+122+132+…+1n2

217. 求 π 的近似值，直到最后一项的值小于 10−6 为止。请填空。

```
#include<stdio.h>
#include<math.h>
main( )
{long i=1;
【 】pi=0;
while(i*i<=10e+6) {pi=【 】; i++;}
pi-sqrt(6.0*pi);
printf("pi=%10.6f\n",pi);
}
```

218. 有 1020 个西瓜，第一天卖一半多两个，以后每天卖剩下的一半多两个，问几天后可以卖完，请填空

```
#include<stdio.h>
main( )
{int day,x1,x2;
day=0; x1=1020;
while(【    】) {x2=(【    】); x1=x2; day++;}
printf("day=%d\n",day);
}
```

219. 下面程序段中循环体的执行次数是【 】。

```
a=10;
b=0;
do{b+=2;a-=2+b; }while (a>=0);
```

220. 下面程序的功能是求 1111 的个、十、百位上的数字之和。请填空。

```
#include<stdio.h>
main()
{ int i,x,s=1,m=0;
for(i=1;i<=11;i++) s=s*11%1000;
do {m+=【 】; s=【 】; } while(s);
printf("m=%d\n",m);
}
```

221. 当运行以下程序时，从键盘输入 1 2 3 4 5 -1↵，则下面程序的运行结果是【 】。

```
#include<stdio.h>
```

```
main()
{ int k=0,n;
do { scanf("%d",&n); k+=n; } while(n!=-1);
printf("k=%d n=%d\n",k,n);
}
```

222. 下面程序段是找出整数的所有因子，请填空。

```
scanf("%d",&x);
i=1;
for( ; 【    】; )
{if(x%i==0) printf("%3d",i);
i++;
}
```

223. 以下程序的功能是根据公式 e=1+11!+12!+13!+…求 e 的近似值，精度要求为 10^{-6} 请填空。

```
#include<stdio.h>
main()
{ int i;double e,new;
    【 】;new=1.0;
   for(i=1; 【  】;i++)
   {new/=(double)i;e+=new;}
   printf("e=%f\n",e);
}
```

224. 若从键盘输入 65 14↵，则下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{int m,n;
printf("Enter m,n:");
scanf("%d%d",&m,&n);
while(m!=n)
{while(m>n) m-=n;
while(n>m) n-=m;
}
printf("m=%d\n",m);
}
```

225. 下面程序段的运行结果是【 】。

```
i=1;s=3;
do{s+=i++;
```

```
if(s%7==0) continue;
else ++i;
}while(s<15);
printf("%d",i);
```

226. **选做** 请预测下列程序运行后的输出，并上机测试，并修正你的答案。若与你的设想不同，请解释该现象。

```
#include <stdio.h>
int main(void)
{
    int i = 0;
    while (++i < 4)
        printf("Hi! ");
    do
        printf("Bye! ");
    while (i++ < 8);
    return 0;
}
```

227. **选做** 请计算 int i = 3; int ans = (++i)+(++i)+(++i);之后的 ans 和 i 的值。（请设计程序，分别用 GCC 与 Visual C++编译验证后将结果修正并誊写在答题纸上）

228. **选做** 部分 JudgeOnline 系统要求参赛者提交含有类似 while(scanf("%d %d",&a, &b) != EOF)片段的程序，请查阅 scanf 函数的返回值含义，作出解释。

229. 找出错误，并将正确版本的程序写出来，尽量接近原意。（注：先用眼睛观察，逐条列出到草稿纸上，然后在 GCC 检查校对自己的答案，誊写到作业纸上。）

```
int main(void)
{
    int i = 1,
    float n;
    printf("Watch out! Here come a bunch of fractions!\n");
    while (i < 30)
        n = 1/i;
        printf(" %f", n); printf("That's all, folks!\n");
    return;
```

```
}
```

230. 默写平均成绩公式。

```
// score_average.c -- calculate the average score
#include <stdio.h>
#define SCORE_LENGTH 10
int main()
{
    float score[SCORE_LENGTH];
    float score_average = 0;
    for (unsigned int i = 0; i < SCORE_LENGTH; i++)
    {
        scanf("%f", &score[i]);
        score_average += score[i];
    }
    score_average /= SCORE_LENGTH;
    printf("The average score is %6.3f.\n", score_average);
    return 0;
}
```

第8章 控制语句：分支和跳转

231. 为了避免嵌套的if-else语句的二义性，C语言规定else总是与【 】组成配对关系。

A. 缩排位置相同的if

B. 在其之前未配对的if

C. 在其之前未配对的最近的if

D. 同一行上的if

232. 选择出合法的if语句（设int x，a，b，c;）【 】。

A. `if(a=b) x++;`

B. `if(a=<b) x++;`

C. `if(a<>b) x++;`

D. `if(a=>b) x++;`

233. 选择出合法的if语句（设int x，y;）【 】。

A.
```
if(x!=y) if(x>y) printf("x>y\n"); else printf("x<y\n");
   else printf("x==y\n");
```

B.
```
if(x!=y) if(x<y) printf("x>y\n"); else printf("x<y\n");
   else printf("x==y\n");
```

C.
```
if(x>y) if(x!=y) printf("x>y\n"); else printf("x<y\n");
   else printf("x==y\n");
```

D.
```
if(x<y) if(x!=y) printf("x>y\n"); else printf("x<y\n");
   e1se printf("x==y\n");
```

234. 以下正确的描述是【 】。

A. continue语句的作用是结束整个循环的执行

B. 只能在循环体内和switch语句体内使用break语句

C. 在循环体内使用 break 语句或 continue 语句的作用相同

D. 从多层循环嵌套中退出时，只能使用 goto 语句

235. 执行下列程序，输入为 1 的输出结果是【 】，

```
#include<stdio.h> main() { int k; scanf("%d",&k); switch(k)
{case 1: printf("%d\n",k++); case 2: printf("%d\n",k++);
case 3: printf("%d\n",k++); case 4: printf("%d\n",k++);
break; default: printf("FULL!\n"); } } }
```

A. 1↲3↲4↲5

B. 1↲2↲3↲4

C. 2↲3↲4↲5

D. 1

236. 输入为 3 的输出结果是【 】。

A. 3↲4

B. 4↲5

C. 3

D. 4

237. 逻辑运算符两侧运算对象的数据类型【 】。

A. 能是 0 或 1

B. 只能是 0 或非 0 正数

C. 只能是整型或字符型数据

D. 可以是任何类型的数据

238. 以下关于运算符优先顺序的描述中正确的是【 】。

A. 关系运算符<算术运算符<赋值运算符<逻辑与运算符

B. 逻辑与运算符<关系运算符<算术运算符<赋值运算符

C. 赋值运算符<逻辑与运算符<关系运算符<算术运算符

D. 算术运算符<关系运算符<赋值运算符<逻辑与运算符

239. 下列运算符中优先级最高的是【 】。

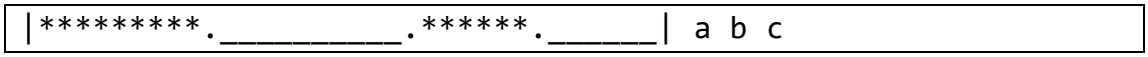
A. <

B. +

C. %

D. !=

240. 表示图中坐标轴上星号部分的正确表达式是【 】。



A. (x<=a) && (x>=b) && (x<=c)

B. (x<=a) || (b<=x<=c)

C. (x<=a) || (x>=b) && (x<=c)

D. (x<=a) && (b<=x<=c)

241. 判断 char 型变量 ch 是否为大写字母的正确表达式是【 】。

A. 'A'<=ch<='Z'

B. (cb>='A') & (ch<='Z')

C. (ch>='A') && (ch<='Z')

D. ('A'<=ch) AND ('Z'>=ch)

242. 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)

243. 若希望当 A 的值为奇数时，表达式的值为“真”，A 的值为偶数时，表达式的值为"假"。则以下不能满足要求的表达式是【 】。

A. A%2==1

B. !(A%2==0)

C. !(A%2)

D. A%2

244. 设有 int a=1，b=2，c=3，d=4，m=2，n=2 执行 (m=a>b)&&(n=c>d) 后 n 的值为【 】。

A. 1

B. 2

C. 3

D. 4

245. 判断 char 型变量 cl 是否为小写字母的正确表达式为【 】。

A. 'a'<=c1<=f'z'

B. (c1>=a) &&(c1<=z)

C. ('a'>=c1) ('z'<=c1)

D. (c1>='a')&&(c1<='z')

246. 执行以下语句后 c 的值为【 】。

```
int a,b,c; a=b=c=1; ++a || ++b && ++c;
```

A. 错误

B. 0

C. 2

D. 1

247. b 的值为【 】。

A. 1

B. 2

C. 错误

D. 0

248. 执行以下语句后 a 的值为【 】，

```
int a=5，b=6,w=1，x=2，y=3，z=4； (a=w>x)&&(b=y>z);
```

A. 5

B. 0

C. 2

D. 1

249. b 的值为【 】。

A. 6

B. 0

C. 1

D. 4

250. 以下不正确的 if 语句形式是【 】。

A. `if(x>y && x!=y);`

B. `if(x==y)x+=y;`

C. `if(x !=y)scanf("%d"，&x) e1se scanf("%d,&y);`

D. `if(x < y){x++；y++;}`

251. 下列运算符中优先级最低的是【 】，

A. `?:`

B. `&&`

C. `+`

D. `!=`

252. 接上题，优先级最高的是【 】。

253. 已知 `int x=10,y=20，z=30;`以下语句 `if(x>y) z=x; x=y; y=z;` 执行后 x，y，z 的值是【 】。

A. x=10，y=20，z=30

B. x=20，y=30，z=20

C. x=20，y=30，z=30

D. x=20，y=30，z=20

254. 以下程序运行结果是【 】。

```
main() {int m=5; if(m++>5)printf("%d\n",m); else
printf("%d\n",m--); }
```

A. 4

B. 5

C. 6

D. 7

255. 当 a=1,b=3,c=5,d=4 时，执行完下面一段程序后 x 的值是【 】。

256. `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. 1

B. 2

C. 3

D. 6

257. 由以函数关系见下表

x	x<0	x=0	x>0
y	x-1	x	x+1

258. 下面程序能正确表示上面关系是【 】。

A. `y=x+1; if (x>=0) if(x==0) y=x; else y=x-1;`

B. `y=x-1; if (x!=0) if(x>0) y=x+1; else y=x;`

C. `if(x<=0) if (x<0) y=x-1; else y=x; else y=x+1;`

D. `y=x; if (x<=0) if(x<0) y=x-1; else y=x+1;`

259. 以下程序运行结果是【 】。

```
main() {int a=100,x=10,y=20,ok1=5,ok2=0; if (x<y) if (y!=10)
if(!ok1) a=1; else if(ok2) a=10; a=-1; printf("%d\n",a); }
```

A. 1

B. 0

c. -1

D. 不确定

260. 为了避免在嵌套的条件语句 `if-else` 中产生二意性，c 语言规定 else 子句总是与【 】配对

A. 缩排位置相同的 if

B. 其之前最近的 if

C. 其之后最近的 if

D. 同一行的 if

261. 以下语句不正确的语句为【 】。

A. `if(x>y);`

B. `if(x=y)&&(x!=0)x+=y;`

C. `if(x!=y)scanf("%d",&x);else scanf("%d",&y);`

D. `if(x<y) {x++;y++;}`

262. 若有 `(exp)?a++:b--` ,则以下表达式中能完全等价于 `(exp)` 是【 】。

A. `(exp==0)`

B. `(exp!=0)`

C. `(exp==1)`

D. `(exp!=1)`

263. 若运行时给变量 x 输入 12，以下程序运行结果是【 】。

```
main() {int x,y; scanf("%d",&x); y=x>12?x+10:x-12;
printf("%d\n",y); }
```

A. 0

B. 22

C. 12

D. 10

264. 以下程序的运行结果是【 】。

```
main() {int k=4,a=3,b=2,c=1; printf("\n%d\n",k<a? k:c<b?
c:a); }
```

A. 4

B. 3

C. 2

D. 1

265. 执行以下程序段后，变量 a,b,c 得值分别是【 】。

```
int x=10,y=9; int a,b,c; a=(--x==y++)?--x:++y; b=x++; c=y;
```

A. a=9,b=9,c=9

B. a=8,b=8,c=10

C. a=9,b=10,c=9

D. a=1,b=11,c=10

266. 若 w,x,y,z,m 均为 int 型变量，则执行下面语句后的 m 值是【 】。

```
w=1; x=2; y=3; z=4; m=(w<x)? w:x; m=(m<y)? m:y; m=(m<z)?
m:z;
```

A. 1

B. 2

C. 3

D. 4

267. 若 w=1,x=2,y=3,z=4,则条件表达式 w<x? w:y<z? y:z 的值是【 】。

A. 4

B. 3

C. 2

D. 1

268. 执行以下程序段后的输出结果是【 】。

```
int w=3, z=7, x=10; printf("%d\n",x>10? x=100: x-10);
printf("%d\n",w++||z++); printf("%d\n",! w>z);
printf("%d\n",w&&z);
```

A. 0↲1↲1↲1

B. 1↲1↲1↲1

C. 0↲1↲0↲1

D. 0↲1↲0↲0

269. 有一堆零件（100 到场 200 之间），如果分成 4 个零件一组的若干组，则多两个零件；若分成 7 个零件一组，则多三个零件；若分成 9 个零件一组，则多 5 个零件。下面程序是求这堆零件的总数，请选择填空。

```
#include<stdio.h> main() {int i; for(i=100;i<200;i++) if((i-
2)%4==0) if(! ((i-3)%7))) if(【 】) printf("%d",i); }
```

A. i%9=5

B. i%9!=5

C. (i-5)%9!=0

D. i%9==5

270. 下面程序的功能是计算 1 至 50 中是 7 的倍数的数值之和，请选择填空。

```
#include<stdio.h> main() {int i,sum=0; for(i=1;i<=50;i++)
if(【 】) sum+=i; printf("%d",sum); }
```

A. (int)(i/7)==i/7

B. (int) i/7==i/7

C. i%7=0

D. i%7==0

271. 下面程序的运行结果是【 】。

```
#include <stdio.h> main() {int x,i; for(i=1;i<=100;i++)
{x=i; if(++x%2==0) if(++x%3==0) if(++x%7==0)
printf("%d",x); } }
```

A. 39 81

B. 42 84

C. 26 68

D. 28 70

272. 若运行以下程序时，从键盘输入 2473↲，则下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{int c;
while((c=getchar())!='\n')
switch(c-'2')
{case 0:
case 1: putchar(c+4); case 2: putchar(c+4);break;
case 3: putchar(c+3);
default: putchar(c+2);break;
}
printf("\n");
}
```

A. 668977

B. 668966

C. 66778777

D. 6688766

273. 若运行以下程序时，从键盘输入 ADescriptor↲，则下面程序的运行结果是【 】。

```
#include <stdio.h>
main()
{char c;
```

```
int v0=0,v1=0,v2=0;
do{
switch(c=getchar())
{case'a':case'A':
case'e':case'E':
case'i':case'I':
case'o':case'O':
case'u':case'U':v1+=1;
default:v0+=1;v2+=1;}
}while(c!='\n');
printf("v0=%d,v1=%d,v2=%d\n",v0,v1,v2);
}
```

A. v0=7,v1=4,v2=7

B. v0=8,v1=4,v2=8

C. v0=11,v1=4,v2=11

D. v0=12,v1=4,v2=12

274. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{int i;
    for(i=1;i <= 5;i++)
    switch(i%5)
    {case 0:printf("*");break;
    case 1:printf("#");break;
    case 2:printf("&");break;
    default:printf("\n");
    }
}
```

A.

```
#&&&*
&
&
```

B.

```
#&
&*
&
```

```
*
```

C.

```
#
&
```

D.

```
#&

*
```

275. 以下描述正确的是【 】。

A. goto 语句只能用于退出多层循环

B. switch 语句中不能出现 continue 语句

C. 只能用 continue 语句来终止本次循环

D. 在循环中 break 语句不能独立出现

276. 如果 a=1,b=2,c=3,d=4，则条件表达式 a < b ? a : c < d ? c : d 的值为【 】。

A. 1

B. 2

C. 3

D. 4

277. 设有 int p,q;，以下不正确的语句是【 】。

A. p*=3;

B. p/=q;

C. p+=3;

D. p&&=q;

278. 在 c 语言中，表示逻辑“真”值用【 】。

279. 请写出通过 break 语句可跳出的语句有哪些。

280. c 语言提供的 3 种逻辑运算符是【 】。

281. 设 x,y,z 均为 int 型变量，请写出描述"x,y 和 z 中有两个为负数"的表达式【 】。

282. 已知 a=7.5,b=2,c=3.6,表达式 a>b&&c>a ||a<b&&!c>b 的值是【 】。

283. 若 a=6,b=4,c=2,则表达式!(a-b) +c-1&&b+c/2 的值是【 】。

284. 条件 2<x<3 或 x<−10 的 c 语言表达式是【 】。

285. 有 int x,y,z; 且 x=3,y=-4,z=5,则表达式 x++-y+(++z)的值为【 】。

286. 有 int a=3,b=4,c=5; ，则表达式 a||b+c&&b==c 的值为【 】。

287. 设 int a=1,b=2,c=3,d;执行 d=!(a+b+c)后，d 的结果是【 】

288. 设有变量定义：a=5,c=4;则(--a==++c)?-a:c++的值是【 】，此时 c 的存储单元的值为【 】。

289. 以下程序的结果是【 】。

```
main()
{ int x,y,z;
x=1,y=1,z=0;
x=x||y&&z;
printf("%d,%d",x,x&&!y||z);
}
```

290. 当运行以下程序时，从键盘出入 China#↲,则下面程序的运行结果是【 】。

```
#include<stdio.h>
main( )
{int v1=0,v2=0;char ch;
while((ch=getchar())!='#')
switch(ch)
{case'a':
case'h':
default:v1++;
case'o':v2++;
}
printf("%d,%d\n",v1,v2);
}
```

291. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{ int i;
for(i=1;i <= 5;i++)
switch(i%2)
{ case 0:i++; printf（"#"）; break;
case 1:i+=2;prinf("*");
default:printf("\n");
}
}
```

292. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{ int i,j=4;
for(i=j;i<=2*j;i++)
switch(i%j)
{ case 0:
case 1:printf("*");break;
case 2:printf("#");
}
}
```

293. 以下程序实现：输入圆的半径 r 和运算标志 m,按照运算标志进行指定运算。

标志	运算
a	面积
c	周长
b	二者均计算

```
#define pi 3.14159
main()
{char m;
float r,c,a;
```

```
printf ("input mark a c or b && r\n");
scanf ("%c %f",&m,&r);
if (【 】)
{ a= pi*r*r;printf ("area is %f",a) ;}
if (【 】)
{ c=2* pi*r;printf ("circle is %f",c) ;}
if (【 】)
{ a= pi*r*r;c=2* pi*r;printf ("area && circle
are %f %f",a,c);}
}
```

294. 若运行时输入: 5999↲，则以下程序的运行结果(保留小数点后一位)是【 】。

```
main()
{int x;
float y;
scanf ("%d",&x);
if(x>=0&&x<=2999) y=18+0.12*x;
if(x>=3000&&x<=5999) y=36+0.6*x;
if(x>=6000&&x<=10000) y=54+0.3*x;
printf("%6.1f",y);
}
```

295. 以下程序实现：输入三个整数，按从大到小的顺序进行输出。请在【 】内填入正确内容。

```
main()
{int x,y,z,c;
scanf("%d %d %d",&x,&y,&z);
if(【 】)
{c=y;y=z;z=c;}
if(【 】)
{c=y;x=z;z=c;}
if(【 】)
{c=x;x=y;y=c}
printf("%d,%d,%d",x,y,z);
}
```

296. 输入一个字符，如果它是一个大写字母，则把它变成小写字母；如果它是一个小写字母，则把它变成大写字母；其他字符不变。请在[ ]内填入正确的内容。

```
main()
{char ch;
```

```
scanf("%c",&ch);
if(【 】) ch=ch+32;
else if(ch>='a'&&ch<='z') 【 】;
printf("%c",ch);
}
```

297. 以下程序的运行结果是【 】。

```
main()
{int a,b,c;
int s,w,t;
s=w=t=0;
a=-1;b=3;c=3;
if(c>0) s=a+b;
if(a<=0)
{if (b>0)
if(c<=0) w=a-b;}
else if (c>0) w=a-b;
else t=c;
printf("%d %d %d",s,w,t);
}
```

298. 请阅读下面程序:

```
main()
{int s,t,a,b;
scanf("%d,%d",&a,&b) ;
s=1,t=1;
if(a>0)s=s+1;
if(a>b) t=s+t;
else if(a==b) t=5;
else t=2*s;
printf("s=%d,t=%d",s,t);
}
```

299. 为了使输出结果 t=4，输入量 a 和 b 应满足的条件是【 】。

300. 以下程序的功能是计算一元二次方程 ax2+bx+c=0 的根. 请在［ ］内输入正确内容.

```
#include"math.h"
main()
{
float a,b,c,t,disc,twoa,term1,term2;
```

```
printf("entera,b,c:");
scanf("%f %f %f",&a,&b,&c);
if(【    】)
   if(【   】) printf("no answer due to input error\n");
   else    printf("the single root is%f\n",-c/b) ;
else
   {disc=b*b-4*a*c;
    twoa=2*a;
    term1=-b/twoa;
    t=abs(disc);
    term2=sqrt(t)/twoa;
    if(【      】)
       printf("complex root\n real part=%f imag
part=%f\n",term1,term2);
    else
       printf("real roots\n root1=%f
root2=%f\n",term1+term2,term1-term2);
    }
}
```

301. 以下程序根据输入的三角形的三边判断是否能组成三角形，若可以则输出它的面积和三角形的类型. 请在［ ］内填入正确内容.

```
#include"math.h"
{floata,b,c;
floats,area;
scanf(%f %f %f",&a,&b,&c);
if (【 】)
{s=(a+b+c)/2;
area=sqrt(s*(s-a)*(s-b)*(s-c));
printf("%f",area);
if(【 】)
printf("等边三角形");
else if(【 】)
printf("等腰三角形");
else if((a*a+b*b==c*c||(a*a+c*c==b*b||(b*b+c*c==a*a))
printf("直角三角形");
else printf("一般三角形");
}
else printf("不能组成三角形");
}
```

302. 某邮局对邮寄包裹有如下规定：若包裹的长宽高任一尺寸超过 1 米或重量超过 30 千克，不予邮寄；对可以邮寄的包裹每件收手续费 0.2 元，再加上根据下表按重量 wei 计算的邮资：

重量（千克）	wei<10	10	20<=wei<=30
收费标准（元）	0.80	0.75	0.70

请在程序的【 】内填入正确内容。

```
main()
{float len,wei,hei,wid,mon,r;
scanf("%f%f%f%f",&len,&wid,&hei,&wei);
if(len>1||wid>1||hei>1||wei>30) 【 】
else if (wei<10) r=0.8;
else if (wei<=20) r=0.75
else if (wei<=30) 【 】;
if(r==-1)printf("error\n")'
else
{【 】;printf("%f",mon);}
}
```

303. 以下程序的功能是判断输入的年份是否是闰年。请在【 】内填入正确内容。

```
main()
{int y,f;
scanf("%d",&y);
if (y%400==0) f=1;
else if (【 】) f=1;
else 【 】;
if (f) printf("%d is not ",y);
printf("a leap year\n");
}
```

304. 以下程序针对输入的截止日期（年：yend,月:mend，日：dend） 和出生日期（yman，mman，dman），计算出实际年龄。请在【 】内填入正确内容。

```
int yend,mend,dend,yman,mman,dman,age;
age=yend-yman;
if(mend【 】mman)age--
else if (mend【 】mman&&dend【 】dman)
```

```
age--;
```

305. 有四个数 a,b,c,d,要求从大到小的顺序输出。请在【 】内填入正确内容。

```
main()
{int a,b,c,d,t;
scanf("%d %d %d %d", &a,&b,&c,&d) ;
if(a<b) {t=a;a=b;b=t;}
if(【 】){t=c;c=d;d=t;}
if(a<c) {t=a;a=c;c=t;}
if(【 】){t=b;b=c;c=t;}
if(b<d){t=b;b=d;d=t;}
if(c<d){t=c;c=d;d=t;}
printf("%d %d %d %d\n",a,b,c,d);
}
```

306. 以下程序的运行结果是【 】

```
#include<stdio.h>
main()
{
int a=-10,b=-3;
printf("%d,",-a%b) ;
printf("%d,",(a-b,a+b) );
printf("%d\n",a-b? a-b:a+b) ;
}
```

307. 以下程序的运行结果是【 】.

```
main()
{
int x,y,z;
x=3;
y=z=4;
printf("%d,",(x>=y>=x)? 1:0);
printf("%d\n",z>=y&&y>=x);
}
```

308. 若运行时输入：100↲，则以下程序的输出结果是【 】.

```
main()
{
int a;
scanf("%d",&a);
printf("%s",(a%2!=0)? "no": "yes");
}
```

309. 如果运行输入字符为 q↵ 时，下面程序的运行结果是【 】。

```
main()
{ char ch;
scanf("%c",&ch);
ch=(ch>='A'&&ch<='Z')?(ch+32):ch;
ch=(ch>='a'&&ch<='z')?(ch-32):ch;
printf("%c",ch);
}
```

310. 以下程序是计算 x,y,z 三个数中最小的。请在【 】内填入正确内容。

```
main()
{ int x=4,y=5,z=8;
int u,v;
u=x<y? 【    】;
v=u<z? 【    】;
printf("%d",v);
}
```

311. 下面程序的功能是用"辗转相除法"求两个正整数的最大公约数。请填空

```
#include<stdio.h>
main( )
{int r,m,n;
scanf("%d%d",&m,&n);
if (m<n) 【     】;
r=m%n;
while(r) {m=n; n=r; r=【     】;}
printf("%d\n",n);
}
```

312. 下面程序的四统计正整数的各位数字中零的个数，并求各位数字中的最大者。请填空。

```
#include<stdio.h>
main()
{ int n,count,max,t;
count=max=0;
scanf("%d",&n);
do
{t=【 】;
if(t==0) ++count;
else if(max < t) 【 】;
n/=10;
```

```
} while(n);
printf("count=%d,max=%d",count,max);
}
```

313. 当运行以下程序时，从键盘输入-1 0↲，则下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{ int a,b,m,n;
m=n=1;
scanf("%d%d",&a,&b) ;
do{ if(a>0) {m=2*n; b++;}
else {n=m+n; a+=2; b++;}
}while(a==b) ;
printf("m=%d n=%d",m,n);
}
```

314. 下面程序的功能是完成用一元人民币换成一分、两分、五分的所有兑换方案。请填空。

```
#include<stdio.h>
main()
{int i,j,k,l=1;
for(i=0;i<=20;i++)
for(j=0;j<=50;j++)
{k=【 】;
if(【 】)
{printf(" %2d%2d%2d ",i,j,k);
l=l+1;
if(l%5==0) printf("\n");
}
}
}
```

315. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{int i,k=0;
for(i=1;;i++)
{k++;
while(k<i*i)
{k++;
if(k%3==0) goto loop;
```

```
}
}
loop:printf("%d,%d",i,k);
}
```

316. 以下程序的执行结果是【 】。

```
#include<stdio.h>
main()
{
int a=2,b=7,c=5;
switch(a>0)
{
case 1:switch(b<0)
{
case 1:printf("@");break;
case 2:printf("!");break;
}
case 0:switch(c==5)
{
case 0:printf("*");break;
case 1:printf("#");break;
case 2:printf("$");break;
}
default:printf("&");
}
printf("n");
}
```

317. 将以下程序段改用非嵌套的 if 语句实现。

```
int s,t,m;
t=(int)(s/10);
switch(t)
{case 10:m=5;break;
case9:m=4;break;
case8:m=3;break;
case7:m=2;break;
case6:m=1;break;
default m=0;
}
```

318. The following program has unnecessarily （不必要的） complex （复杂） relational （关系） expressions （表达式） as well as some outright （直白的） errors. Simplify （简化） and correct it.（注：先用眼睛观察，逐条列出到草稿纸上，然后在GCC检查校对自己的答案，勿忘加上输出所有告警 -Wall 选项，誊写到作业纸上。）

```
#include <stdio.h>
int main(void)                                          /*
1 */
{                                                       /*
2 */
int weight, height;         /* weight in lbs, height in
inches */
                                                        /* 4 */
scanf("%d , weight, height);                            /* 5 */
if (weight < 100 && height > 64)                        /* 6 */
    if (height >= 72)                                   /* 7 */
        printf("You are very tall for your weight.\n");
    else if (height < 72 && > 64)                       /* 9 */
        printf("You are tall for your weight.\n");  /* 10 */
  else if (weight > 300 && ! (weight <= 300)            /* 11 */
               && height < 48)                          /* 12 */
  if (!(height >= 48) )                                 /* 13 */
     printf(" You are quite short for your weight.\n");
else                                                    /* 15 */
printf("Your weight is ideal.\n");                      /* 16 */
                                                        /* 17 */
return 0;
}
```

319. **选做** 请计算 `if ((5 < val < 100) || (val++) || (printf("%d", val))) printf("%c", '*');`之后的输出。（请设计程序，分别用 GCC 与 Visual C++编译验证后将结果修正并誊写在答题纸上）

320. **选做** 阅读程序片段 `x = 2.1, y = 1.3, z = 1.5; x = ( y = 3, (z = ++y + 2) +5 || (z = ++y * 2) +5 );`，请写出运行后 x、y 和 z 的取值，并简要说明计算过程。（请设计程序，验证后将结果修正并誊写在答题纸上）

321. 请写出下列表达式的值：

a. d = 5 + (6 > 2)

b. 'X' > 'T' ? 10 : 5

c. x > y ? y > x : x > y

322. 有以下程序

```
#include <stdio.h>
int main()
{
int x;
scanf("%d", &x);
if (x>15) printf("%d",x-5);
if (x>10) printf("%d",x);
if (x>5) printf("%d\n",x+5);
return 0;
}
```

a. 请写出程序运行时从键盘输入 12 和回车后的输出结果。

b. 请写出等价的 switch-case 语句。

323. 有以下函数

```
#include <stdio.h>
void putchars(char ch, int n)
{
int i;
for(i=1;i<=n;i++)
    printf(i%6!=0?"%c":"%c\n",ch);
}
```

请说明运行 putchars('*', 24)后函数共输出星号（*）的行数。

324. 阅读程序片段，写出输出结果。

```
#include<stdio.h>
int main(void)
{
char b, c;
int i;
b='a';
c='A';
for(i=0;i<6;i++)
{
```

```
    if (i%2)
        putchar(i+b);
    else
        putchar(i+c);
}
printf("\n");
}
```

325. 阅读程序片段，选择正确的一项描述

```
#include <stdio.h>
int main(void)
{
// Some statements omitted here.
while(getchar()!='\n');
    ...// Some statements omitted here.
}
```

a. 该 while 语句将陷入死循环

b. getchar()不可以出现在 while 语句的条件表达式中

c. 执行至 while 循环时，只有按回车键才能继续执行循环后的语句

d. 执行至 while 循环时，按任意键将继续执行循环后的语句

326. 阅读程序片段，写出输出结果。

```
#include <stdio.h>
int main(void)
{
int k=1, s=0;
do
{
    if ((k & 2)!=0)
        continue;
    s+=k;
    k++;
} while (k>10);
printf("s=%d\n", s);
}
```

（注：k&2 的意思是 k 与 2 按位取与，例如 5 & 3 为 00000101 & 0000011，即 00000001）

第9章 字符串输入输出与输入验证

327. 假设你在写做 JudgeOnline 的大整数乘法题目，输入的测试数据毫无规律而且长度大，而且程序并非一次就能调试正确。为避免输入，测试时应如何做?

328. 请比较缓冲输入和非缓冲输入的异同。

第10章 函数 (1)

329. **选做，选做时该题留空格** 若使用一维数组名作函数实参，则以下正确的说法是【 】。

A. 必须在主调函数中说明此数组的大小

B. 实参数组类型与形参数组类型可以不匹配

C. 在被调函数中，不需要考虑形参数组的大小

D. 实参数组名与形参数组名必须一致

330. 折半查找法的思路是：先确定待查元素的范围，将其分成两半，然后测试位于中间点元素的值。如果该待查元素的值大于中间点元素，就缩小待查范围，只测试中点之后的元素；反之，测试中点之前的元素，测试方法铜钱。函数 binary 的作用是应用折半查找法从存有的 10 个整数的 a 数组中对关键字 m 进行查找，若找到，返回其下标值；反之返回-1。

```
Binary(int a[10],int m)
{int low=0,high=9,mid;
while(low<=high)
{mid=(low+high)/2;
if(m<a[mid]) 【】
else if(m>a[mid]) 【】
else return(mid);
}
return(-1);
}
```

331. 第 1 个空格应填写的是【 】。

A. high=mid-1

B. low=mid+1

C. high=mid+1

D. low=mid-1

332. 上题第 2 个空格应填写的是【 】。

333. 如果在一个函数中的复合语句中定义了一个变量，则以下正确的说法是【 】。

A. 该变量只在该复合语句中有效

B. 在该函数中有效

C. 在本程序范围内均有效

D. 非法变量

334. 以下不正确的说法为【 】。

A. 在不同的函数中可以使用相同名字的变量

B. 形式参量是局部变量

C. 在函数内定义的变量只在本函数范围内有效

D. 在函数内的复合语句中定义的变量在本函数范围内有效

335. 以下程序的正确运行结果是【 】。

```
#define MAX 10
int a[MAX],i;
main()
{printf("\n");sub1();sub3(a);sub2();sub3(a);
}
sub2()
{int a[MAX],i,max;
max=5;
for(i=0;i<max;i++) a[i]=i;
}
sub1()
{for(i=0;i<MAX;i++) a[j]=i*2;
}
sub3(int a[])
{int i;
for(i=0;i<MAX;i++) printf("%d ",a[i]);
printf("\n");
}
```

A. 0 2 4 6 8 10 12 14 16 18↲0 1 2 3 4

B. 0 1 2 3 4↲0 2 4 6 8 10 12 14 16 18

C. 0 1 2 3 4 5 6 7 8 9↵0 1 2 3 4

D. 0 2 4 6 8 10 12 14 16 18↵0 2 4 6 8 10 12 14 16 18

336. 以下程序的正确运行结果是【 】。

```
#include <stdio.h>
void num()
{extern int x,y;int a=15,b=10;
x=a-b;
y=a+b;
}
int x,y;
main()
{ int a=7,b=5;
x=a+b;
y=a-b;
num();
printf("%d,%d\n",x,y);
}
```

A. 12,2

B. 不确定

C. 5,25

D. 1,12

337. **选做，选做时该题留空格** 凡是函数中未指定存储类别的局部变量，其隐含的存储类别为【 】。

A. 自动(auto)

B. 静态(static)

C. 外部(extern)

D. 寄存器(register)

338. **选做，选做时该题留空格** 在一个C程序文件中，若要定义一个只允许本源文件中所有函数使用的全局变量，则该变量需要使用的存储类别是【 】。

A. extern

B. register

C. auto

D. static

339. 以下程序的正确运行结果是【 】。

```
main()
{int a=2,i;
for (i=0;i<3;i++) printf("%d ",f(a));
}
f(int a)
{ int b=0;static int c=3;
b++; c++;
return(a+b+c);
}
```

A. 7 7 7

B. 7 10 13

C. 7 9 11

D. 7 8 9

340. 以下程序的正确运行结果是【 】。

```
#include<stdio.h>
main()
{ int k=4,m=1,p;
p=func(k,m);printf("%d,",p);
p=func(k,m);printf("%d\n",p);
}
func(int a,intb)
{ static int m=0,i=2;
i+=m+1;
m=i＋a+b;
return(m);
}
```

A. 8,17

B. 8,16

C. 8,20

D. 8,8

341. C 语言规定，可执行程序的开始执行点是【 】。

342. 在 C 语言中，一个函数一般由两个部分组成，它们是【 】和【 】。

343. 若输入的值是-125，以下程序的运行结果是【 】。

```
#include "math.h"
main()
{ int n;
    scanf("%d",&n);
    printf("%d=",n);
    if(n<0) printf("-");
    n=fabs(n);
    fun(n);
}
fun(int n)
{ int k,r;
    for(k=2;k<=sqrt(n);k++)
    { r=n%k;
        while(r==0)
        {printf("%d",k);
        n=n/k;
        if(n>1) printf("*");
        r=n%k;
    }
    }
    if(n!=1) printf("%d\n",n);
}
```

344. 下面 add 函数的功能是求两个参数的和，并将和值返回调用函数。函数中错误的部分是【 】；改正后为【 】。

```
void add(float a,float b)
{ float c;
    c=a+b;
    return c;
}
```

345. 以下程序的运行结果是【 】。

```
main()
{ int i=2,x=5,j=7;
```

```
    fun(j,6);
    printf("i =%d;j=%d;x=%d\n",i,j,x);
}
fun(int i,int j)
{ int x=7;
    printf("i=%d;j=%d;x=%d\n",i,j,x);
}
```

346. 以下程序的运行结果是【 】。

```
main()
{ increment();
    increment();
    increment();
}
increment()
{ int x=0;
    x+=1;
    printf("%d",x);
}
```

347. 以下程序的运行结果是【 】。

```
#include <stdio.h>
main()
{ int a=1,b=2,c;
    c=max(a,b);
    printf("max is %d\n",c);
}
max(int x, int y)
{ int z;
    z=(x>y) x:y;
    return(z);
}
```

348. 以下程序的功能是根据输入的"y"("Y")与 "n"("N"), 在屏幕上分别显示出 This is YES.与 This is NO.。请填空。

```
#include <stdio.h>
void YesNo(char ch)
{ switch(ch)
    { case'y':
    case'Y':printf("\nThis is YES.\n");break;
    case'n':
```

```
    case'N': printf("\nThis is NO.\n")
    }
}
main()
{char ch;
    printf("\nEnter a char 'y','Y',or'n','N');
    ch=【 】;
    printf("ch:%c",ch);
    YesNo(ch);
}
```

349. 以下 Check 函数的功能是对 value 中的值进行四舍五入计算，若计算后的值与 ponse 值相等，则显示"WELL DONE！！"，否则显示计算后的值。已有函数调用语句 Check（ponse,value）；请填空。

```
void check(int ponse,float value)
{int val;
    val=【 】;
    printf("计算后的值：%d",val);
    if(ponse == val) printf("\n WELL DONE!!!\n");
    else printf("\nSorry the correct is %d\n",val);
}
```

350. 以下程序功能是【 】。

```
#include<stdio.h>
f(int n)
{int i,j,k;
    i=n/100;j=n/10-i*10;k=n%10;
    if(i*i*i+j*j*j+k*k*k==n)return n;
    else return 0;
}
main()
{ int n,k;
    printf("output");
    for(n=100;n<1000;n++)
    {k=f(n)
        if(k!=0)
            printf("%d",k);
    }
    printf("\n");
}
```

351. 以下程序的功能是用二分法求方程 2x3−4x2+3x−6=0 的根，并要求绝对误差不超过 0.001。请填空

```
#include<stdio.h>
float f(float x)
{return(2*x*x*x-4*x*x+3*x-6);}
main()
{ float m=-100,n=90,r;
r=(m+n)/2;
while(f(r)*f(n)!=0)
{if(【 】)m=r;
else n=r;
if(【 】) break;
r=(m+n)/2;
}
printf("The is fangcheng jie is %6.3f\n",r);
}
```

352. 若输入一个整数 10，以下程序的运行结果是【 】。

```
main()
{ int a,e[10],c,i=0;
    printf("输入一整数\n");
    scanf("%d",&a);
    while(a!=0)
    {c=sub(a);
        a=a/2;
        e[i]=c;
        i++
    }
    for(;i>0;i--) printf("%d",e[i-1]);
}
sub(int a)
{ int c;
    c=a%2;
    return c;
}
```

353. 以下程序的功能是计算下面函数的值。请填空。

354. f(x,y,z)=sin(x)sin(x−y)sin(x−z)+sin(y)sin(y−z)sin(y−x)+sin(z)sin(z−x)sin(z−y)

```
#include<stdio.h>
```

```
#include<math.h>
float f();
main()
{float x,y,z,sum;
printf("\ninput x,y,z:\n");
scanf("%f%f%f",&x,&y,&z);
sum=【 】;
printf("sum=%f\n",sum);
}
float f(float a,float b,float c)
{float value;
value=【 】;
return(value);
}
```

355. 已有函数 pow，现要求取消变量 i 后 pow 函数的功能不变。请填空。

修改前的 pow 函数

```
pow(int x,int y)
{int i,j=1;
for(i=1;i<=y:++i) j=j*x;
return(j);
}
```

修改后的 pow 函数

```
pow(int x,int y)
{int j;
for(【 】;【 】;【 】)j=j*x;
return(j);
}
```

356. 以下程序的运行结果是输出如下图形。请填空。

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

```
#include<stdio.h>
void a(int i)
{ int j,k;
```

```
for(j=0;j<=7-i;j++) printf(" ");
for(k=0;k<【 】;k++) printf("*");
printf("\n");
}
main()
{ int i;
for(i=0;i<3;i++) 【 】;
for(i=3;i>=0;i--) 【 】;
}
```

357. 以下程序的功能是求三个数的最小公倍数,请填空.

```
#include<stdio.h>
max(int x,int y,int z)
{ if(x>y&&x>z) return(x);
else if(【 】) return(y);
else return(z);
}
main()
{ int x1,x2,x3,i=1,j,x0;
printf("Input 3 number:");
scanf("%d%d%d",&x1,&x2,&x3);
x0=max(x1,x2,x3);
while(1)
{ j=x0*i;
if(【 】) break;
i=i+1;
}
printf("The is %d %d zuixiaogongbei is %d\n",x1,x2,x3,j);
}
```

358. 函数 gongyu 的作用是求整数 num1 和 num2 的最大公约数,并返回该值,请填空

```
gongyu(int num1,int num2)
{ int temp,a,b;
if(num1【 】num2)
{ temp=num1; num1=num2; num2=temp;}
a=num1; b=num2;
while(【 】)
{ temp=a%b; a=b; b=temp; }
return【 】;
}
```

359. 下面函数 π 的功能是:根据以下公式,返回满足精度(0.0005)要求的 π 的值

$$\frac{\pi}{2}=1+\frac{1}{3}+\frac{1}{3}\cdot\frac{2}{5}+\frac{1}{3}\cdot\frac{2}{5}\cdot\frac{3}{7}+\frac{1}{3}\cdot\frac{2}{5}\cdot\frac{3}{7}\cdot\frac{4}{9}+\cdots$$

```
#include<conio.h>
#include<math.h>
#include<stdio.h>
double pi(double eps)
{ double s,t; int n;
for(【 】;t>eps;n++)
{ s+=t;
t=n*t/(2*n+1);
}
return(【 】);
}
main()
{ double x;
printf("\nPlease enter a precision:");
scanf("%lf",&x);
printf("\neps=%lf,pi=%lf",x,pi(x));
}
```

360. 下面是一个计算阶乘的程序.程序中的错误语句是【 】,应改为【 】。

```
#include"stdio.h"
double factorial(int);
main()
{ int n;
printf("Enter an interger:");
scanf("%d",&n);
printf("\n\n%d! =%lg\n\n",n,factorial(n));
}
double factorial(int n)
{ double result=1.0;
while(n>1 || n<170) result*=--n;
return result;
}
```

361. 函数 f 中的形参 a 为一个 10×10 的二维数组,n 的值为 5,以下程序段的运行结果为【 】。

```
f(int a[10][10],int n)
{ int i,j,k;
    j=n/2+1; a[1][j]=1; i=1;
    for(k=2;k<=n*n;k++)
```

```
    { i=i-1; j=j+1;
    if((i<1)&&(j>n)) {i=i+2;j=j-1;}
    else {if(i<1) i=n;
    if(j>n) j=1;
    if(a[i][j]==0) a[i][j]=k;
    else {i=i+2; j=j-1; a[i][j]=k; }
    }
}
```

362. 下面函数 func 的功能是【 】。

```
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
long func(long num)
{long k=1;
num=labs(num);
do
{k*=num%10;
num/=10;
}while(num);
return(k);
}
main()
{long n;
printf("\nPlease enter a number:");
scanf("%ld",&n);
printf("\nThe product of its digits is %ld.",func(n));
}
```

363. 以下程序的运行结果是【 】。

```
#include <stdio.h>
main()
{printf("FACT(5):%d\n",fact(5));
printf("FACT(1):%d\n",fact(1));
fact(-5);
}
fact(int value)
{if(value<0) {printf("FACT(<0):Error!\n");return(-1);}
else if(value==1 ||value==0) return(1);
}
```

364. 以下程序的运行结果是【 】。

```
#include<stdio.h>
f(int a[])
{int i=0;
while (a[i]<=10)
{printf("%d",a[i]);
i++;}
}
main()
{int a[]={1,5,10,9,11,7};
f(a+1);
}
```

365. 以下程序的运行结果【 】。

```
main()
{ int a[3][3]={1,3,5,7,9,11,15,17};
int sum;
sum=func(a);
printf("\nsum=%d\n",sum);
}
func(int a[][3])
{ int i,j,sum=0;
for(i=0;i<3;I++)
for(j=0;j<3;j++)
{a[i][j]=i+j;
if(i==j)sum=sum+a[i][j];
}
return(sum);
}
```

366. 阅读下面程序,完成下列问题中的填空。问题

（1）此程序在调用函数 f 后的运行结果是【 】。

（2）若将函数 f 中的 for(j=i+1;j<4;j++)改为 for(j=0;j<3-i;j++).

则程序的运行结果是【 】。

```
f(int s[][4])
{ int i,j,k;
    for(i=0;i<3;i++)
    for(j=i+1;j<4;j++)
    {k=s[i][j];s[i][j];s[j][i]=k;}
}
main()
{ int s[4][4],i,j;
```

```
    for(i=0;i<4;i++)
    for(j=0;j<4;j++) s[i][j]=i-j;
    f(s);
    for(i=0;i<4;i++)
    { printf("\n");
        for(j=0;j<4;j++)
        printf("%4d",s[i][j];
    }
}
```

367. 以下 search 函数的功能是利用顺序查找法从数组 a 的 10 个元素中对关键字 m 进行查找。顺序查找法的思路是：从第一个元素开始，从前向后依次与关键字比较，直到找到此元素或查找到数组尾部时结束。若找到，返回此元素的下标；若仍未找到，则返回值-1。请填空。

```
#include<stdio.h>
int search(int a[10],int m)
{ int i;
for(i=0;i<=9;i++) if(【  】) return(i);
return(-1);
}
main()
{ int a[10],m,i,no;
    . . .
    no=search(【  】);
    if(【  】) printf("\Nok found! %d",no+1);
    else printf("\nSorry NOT found!");
}
```

368. 已定义一个含有 30 个元素的数组 s,函数 fav1 的功能是按顺序分别赋予各元素从 2 开始的偶数，函数 fav2 则按顺序每五个元素求一个平均值，并将该值存放在数组 w 中。请填空。

```
#define SIZE 30
fav1(float s[])
{ int k,i;
for (k=2,i=0;i<SIZE;i++)
{【  】;
k+=2;
}
}
fav2(float s[],float w[])
```

```
{float sum;int k,i;
    sum=0.0;
    for(k=0,i=0;i<SIZE;i++)
    {sum+=s[i];
    if((i+1)%5==0)
    {w[k]=sum/5;
    【  】;
    k++;
    }
    }
}
main()
{float s[SIZE],w[SIZE/5],sum;
int i,k;
fav1(s);
fav(s,w);
}
```

369. 以下程序的运行结果是【 】，其算法是【 】。

```
main()
{ int a[5]={5,10,-7,3,7},i,t,j;
sort【 】;
for(i=0;i<=4;i++) printf("%d",a[i]);
}
sort(int a[])
{ int i,j,t;
for(i=0;i<4;i++)
for(j=0;j<4-i;j++)
if(a[j]>a[j+1] {t=a[j];a[j]=a[j+1];a[j+1]=t;}
}
```

370. 以下程序的运行结果是【 】，其算法是【 】。

```
main()
{ int a[5]={9,6,8,3,-1},i,t,j,p;
    sort(a);
    for(i=0;i<=4;i++) printf("%4d",a[i]);
}
sort(int a[])
{ int i,j,t,p;
    for(j=0;j<4;j++)
    {p=j;
    for(i=j;i<=4;i++) if(a[i]<a[p]) p=i;
```

```
    t=a[p];a[p]=a[j];a[j]=t;
    }
}
```

371. 函数 del 的作用是删除有序数组 a 中的指定元素 x。已有调用语句 n=del(a,n,x);其中实参 n 为删除前数组元素的个数，赋值号左边的 n 为删除后数组元素的个数。请填空。

```
del(int a[],int n,int x)
{ int p,i;
p=0;
while(x>=a[p]&&p<n) 【 】;
for(i=p-1;i<n;i++) 【 】;
n=n-1;
return n;
}
```

372. 以下程序的运行结果是【 】。

```
#include<stdio.h>
func(int array[][4],int m)
{int i,j,k;
k=0;
for(i=0;i<3;i++)
for(j=0;j<4;j++) if(array[i][j]<m) k=k+array[i][j];
return(k);
}
main()
{static int a[3][4]={{1,13,5,7},{2,4,26,8},{10,1,3,12}};
int i,j,m;
for(i=0;i<3;i++)
{for(j=0;j<4;j++) printf("%4d",a[i][j]);
printf("\n");
}
m=10;
printf("\nthe value is %d\n",func(a,m));
}
```

373. 函数 swap(int x,int y)可完成对 x 和 y 值的交换。在运行调用函数中的如下语句后，a[0]和 a[1]的值分别为【 】，原因是【 】。

```
a[0]=1;a[1]=2;
swap(a[0],a[1]);
```

374. 函数 swap(arr,n)可完成对 arr 数组从第 1 个元素到第 n 个元素两两交换。在运行调用函数中的如下语句后，a[0]和 a[1]的值分别为【 】，原因是【 】。

```
a[0]=1;a[1]=2;
swap(a,2);
```

375. 以下程序可计算 10 名学生 1 门课成绩的平均分，请填空。

```
float average(float array[10])
{ int i;float aver,sum=array[0];
for(i=1;【 】;i++) sum+=【 】;
aver=sum/10;
return(aver);
}
main()
{ float score[10],aver;int i;
printf("\ninput 10 scores:");
for(i=0;i<10;i++) scanf("%f",&score[i]);
aver=【 】;
printf("\naverage score is %5.2f\n",aver);
}
```

376. 函数 yahui 能够按以下形式构成一个杨辉三角形，请填空。

```
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
1 5 10 10 5 1
……
```

377.

```
#define N 11
yahui(int a[][N])
{ int i,j;
for(i=1;i<N;i++) { a[i][1]=1; a[i][i]=1;}
for(【 】;i<N;i++)
for(j=2;【 】;j++) a[i][j]=【 】+a[i-1][j];
}
```

378. 以下程序的功能是应用弦截法求方程 X3−5X2+16X−80=0 的根,其中 f 函数课根据指定的 X 的值求出方程的值;函数 xpoint 可根据 x1 呵 x2 求出 f(x1)和 f(x2)的连线与 x 轴的交点;函数 root 用来求区间(x1,x2)的实根,请编写 root 函数.

```
#include<math.h>
float root(float x1,float x2)
{ }
float f(float x) /* 略 */
{ . . . }
float xpoint(float x1,float x2) /* 略 */
{ . . . }
main()
{float x1,x2,f1,f2,x;
    do
    {printf("input x1,x2:\n);
    scanf("%f%f",&x1,&x2);
    printf("x1=%5.2f,x2=%5.2f\n',x1,x2);
    f1=f(x1);f2=f(x2);
    }while(f1*f2>=0);
    x=root(x1,x2);
    printf(A root of equation is %8.4f",x);
}
```

379. 以下程序的功能是应用下面的近似公式 ex=1+x+x22!+x33!+… (前 20 项的和)计算 en.函数 f1 用来计算每项分子的值,函数 f2 用来计算每项分母的值.请编写 f1 和 f2 函数.

```
float f2(int n)
{ }
float f1(int x,int n)
{ }
main()
{ float exp=1.0;int n,x;
printf("Input a number:")'
scanf("%d",&x); printf("%d\n",x);
exp=exp+x;
for(n=2;n<=19;n++) exp=exp+f1(x,n)/f2(n);
printf("\nThe is exp(%d)=%8.4f\n",x,exp)'
}
```

380. 运行结果:Input a number:3 The is exp(3)=20.0855。

381. a 是一个 2*4 的整型数组,且各元素均已赋值.函数 max_value 可求出其中的最大元素值 max,并将此值返回主调函数.今有函数调用语句 max=max_value(a);请编写 void max_value(int arr[][4]);函数.

382. 设计一个函数，输出整数 n 的所有素数因子。

383. 输入整数 n，输出高度为 n 的等边三角形。当 n=5 时的等边三角形如下：

```
    *
   ***
  *****
 *******
*********
```

384. 设计函数，从键盘输入一行字符，返回最长单词的长度，同时输出该单词的位置。

第11章 函数 (2)

385. 关于 C 语言，下列说法正确的是【 】。

A. 函数的形式参量在函数未调用时已分配存储空间。

B. 如果函数在 main 函数之前定义，则不必使用原型说明。

C. 若函数无 return 语句，则不返回值。

D. 一般函数的形式参量和实际参数类型应保持一致或兼容。

386. 关于 C 语言，在同一作用域范围内，下列说法正确的是【 】。

A. 存在函数名和参数类型相同，但参数名不同的函数。

B. 存在函数名和参数类型相同，但返回类型不同的函数。

C. 存在函数名相同，但参数类型和参数名不同的函数。

D. 不存在函数名和变量名相同。

387. What is the difference between an actual argument（实际参数） and a formal parameter（形式参量）?

388. 将复用的代码封装成函数是否会带来性能上的损失？比较以下代码的运行时间：

```
// func_time.c
// Author by: Jianming Huang
#include <stdio.h>
#include <sys/time.h>
#define TIMES 10000000
int add(int a, int b); // proto_type
int main()
{
    unsigned i, t, a = 10000, b = 99999; struct timeval
start, end; double time;
    gettimeofday(&start, NULL);
    // test the function mode for TIMES times
    for (i = 1; i <= TIMES; ++i)
    {
        t = add(a, b);
```

```
    }
    gettimeofday(&end, NULL);
    time = 1000000 * (end.tv_sec - start.tv_sec) +
end.tv_usec - start.tv_usec;
    printf("func mode: %.3lfms\n", time / 1000);
    gettimeofday(&start, NULL);
    // test the "+" mode for TIMES times
    for (i = 1; i <= TIMES; ++i)
    {
        t = a + b;
    }
    gettimeofday(&end, NULL);
    time = 1000000 * (end.tv_sec - start.tv_sec) +
end.tv_usec - start.tv_usec;
    printf("a+b mode: %.3lfms\n", time / 1000);
    return 0;
}
int add(int a, int b)
{
    return (a + b);
}
```

389. 以下程序的功能是用递归方法计算学生的年龄,已知第一位学生年龄最小,为 10 岁,其余学生一个比一个大 2 岁,求第 5 位学生的年龄.请填空.

递归公式如下:

$$age(n)=\begin{cases}10, & n=1;\\ age(n-1)+2, & n>1.\end{cases}$$

```
#include<stdio.h>
age(int n)
{int c;
if(n==1) c=10;
else c=【 】;
return【 】;
}
main()
{int n=5;
printf("age: %d\n",【 】);
}
```

390. 下面程序的功能是【 】，运行结果是【 】。

```
long fib(int g)
{switch(g)
{case 0: return0;
case 1: case 2: return1;}
return(fib(g-1)+fib(g-2));
}
main()
{long k;
k=fib(7);
printf("k=%d\n",k);
}
```

391. 下面程序的运行结果是【 】。

```
main()
{int i=5;
printf("%d\n",sub(i));
}
sub(int n)
{int a;
if(n==1) return 1;
a=n+sub(n-1);
return【 】;
}
```

392. 以下程序是应用递归算法求某数 a 的平方根，请填空。求平方根的迭代公式为 $x_1=\frac{1}{2}(x_0+\frac{a}{x_0})$。

```
#include<math.h>
double mysqrt(double a,double x0)
{double x1,y;
x1=【 】;
if(fabs(x1-x0)>0.00001) y=mysqrt(【 】);
else y=x1;
return y;
}
main()
{double x;
printf("Enter x:");scanf("%lf",&x);
printf("The sqrt of %f=%f\n",x,mysqrt(x,1.0));
}
```

393. 以下函数 p 的功能是用递归方法计算 x 的 n 阶勒让多项式的值.已有调用语句 p(n,x);请编写 p 函数.递归公式如下:

$$p_n(x)=\begin{cases}1, & n=0;\\ x, & n=1;\\ \dfrac{(2n-1)p_{n-1}(x)-(n-1)p_{n-2}(x)}{n}, & n>1.\end{cases}$$

```
float p(int n,int x)
{ }
```

394. 下列句子中，哪些语句是错误的？请更正后重新誊写。

```
#include <stdio.h>
int main()
{
    int iv;                     // Line 4
    int iv2 = 1024;             // Line 5
    int iv3 = 999;              // Line 6
    int &reiv;                  // Line 7
    int *pi;                    // Line 8
    *pi = 5;                    // Line 9
    pi = &iv3;                  // Line 10
    const double di;            // Line 11
    const double *pc = &di;     // Line 12
    return 0;
}
```

395. **选做** 用递归方法计算下列函数的值：

$$f(x,n)=x-x^2+x^3-x^4+\cdots(-1)^{n-1}x^n, n>0$$

396. **选做** 斐波那契数列公式如下：

$$F(n)=\begin{cases}F(n-1)+F(n-2), & n\geq 2;\\ n, & n<2.\end{cases}$$

其中， n∈N。请按以下写出程序，对给定 n 求出 F(n)：

a. 使用递归函数的思想实现；

b. 不使用递归函数的思想实现；

课件提到，使用递归可以处理反序问题，但为何最好避免这样做。请通过计算 这样的数，从时间上和代码简洁性方面比较以上两个程序的优缺点。（注：先尝试写出思路或画出草图，将其翻译为C语言，然后自行检查错误和验算。多试几次，尽量在编译器和系统帮助之外，手写出准确的程序。最后在 GCC 检查校对自己的答案，誊写到作业纸上，不要艺高人胆大。）

第12章 数组与指针 (1)

397. 衡量软件好坏的主要标准是看它的内聚性与耦合性，应（ ）内聚、（ ）耦合。

A. 高；高

B. 高；低

C. 低；高

D. 低；低

398. 若使用一维数组名作函数实参，则以下正确的说法是【 】。

A. 必须在主调函数中说明此数组的大小

B. 实参数组类型与形参数组类型可以不匹配

C. 在被调函数中，不需要考虑形参数组的大小

D. 实参数组名与形参数组名必须一致

399. 在 C 语言中，引用数组元素时，其数组下标的数据类型允许是【 】

A. 整型常量

B. 整型表达式

C. 整型常量或整型表达式

D. 任何类型的表达式

400. 以下对一维整型数组 a 的正确说明是【 】。

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];
```

401. 若有说明：int a[10]; 则对 a 数组元素的正确引用是【 】。

A. a[10]

B. a[3.5]

C. a(5)

D. a[10-10]

402. 在 C 语言中，一位数组的定义方式为：类型说明符 数组名【 】。

A. [常量表达式]

B. 整型表达式

C. [整型常量]或[整型表达式]

D. [整型常量]

403. 以下对一维长度为 10 的整型数组 a 进行声明并初始化的语句，不正确的是【 】。

A. int a[10]=(0,0,0,0,0);

B. int a[10]={};

C. int a[]={10*0};

D. int a[10]={10*1};

404. 以下对二维整型数组 a 的正确说明是【 】。

A. inta[3][];

B. float a(3,4);

C. double a[1][4];

D. float a(3)(4);

405. 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)

406. 若有说明：int a[3][4];则对 a 数组元素的非法引用是【 】。

A. a[0][2*1]

B. a[1][3]

C. a[4-2][0]

D. a[0][4]

407. 以下对二维整型数组 a 进行初始化的语句，不正确是【 】。

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

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

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

D. int a[][3]={{1,0,1},{},{1,1}};

408. 以下对二维整型数组 a 进行初始化的语句，不正确是【 】。

A. int a[2][3]={0};

B. int a[][3]={{1,2},{0}};

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

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

409. 若有说明：int a[3][4]={0};则下面正确的叙述是【 】。

A. 只有元素 a[0][0]可得到初值 0

B. 此说明语句不正确

C. 数组 a 中各元素都可得到初值，但其值不一定为 0

D. 数组 a 中每个元素均可得到初值 0

410. 若有说明：int a[][4]={0,0};则下面正确的叙述是【 】。

A. 数组 a 中每个元素均可得到初值 0

B. 二维数组 a 的第一维大小为 1

C. 因为二维数组 a 中第二维大小除以初值个数的商为 1，数组 a 的行数为 1

D. 只有元素 a[0][0]和 a[0][1]可得到初值 0，其余元素均得不到初值 0

411. 若有说明：int a[3][4];则数组 a 中各元素【 】。

A. 可在程序的运行阶段得到初值 0

B. 可在程序的编译阶段得到初值 0

C. 不能得到确定的初值

D. 可在编译或运行阶段得到初值 0

412. C 语言数组名做实参时，它和对应形参之间的数据传递方式是【 】。

A. 用户指定传递方式

B. 无传递

C. 单向值传递

D. 地址传递

413. 下面程序【 】。（每行程序前面的数字表示行号）。

```
main()
{
    int a[3]={3*0};
    int i;
    for(i=0;i<3;i++) scanf("%d",&a[i]);
    for(i=1;i<3;i++) a[0]=a[0]+a[i];
    printf("%d\n",a[0]);
}
```

A. 第 3 行有错误

B. 第 7 行有错误

C. 第 5 行有错误

D. 没有错误

414. 下面程序【 】。（每行程序前面的数字表示行号）。

```
main()
{
```

```
    float a[10]={0.0};
    int i;
    for(i=0;i<3;i++) scanf("%d",&a[i]);
    for(i=1;i<10;i++) a[0]=a[0]+a[i];
    printf("%f\n",a[0]);
}
```

A. 没有错误

B. 第 3 行有错误

C. 第 5 行有错误

D. 第 7 行有错误

415. 下面程序【 】。（每行程序前面的数字表示行号）。

```
main()
{
    int a[3]={1};
    int i;
    scanf("%d",a);
    for(i=1;i<3;i++) a[0]=a[0]+a[i];
    printf("a[0]=%d\n",a[0]);
}
```

A. 第 3 行有错误

B. 第 6 行有错误

C. 第 7 行有错误

D. 第 5 行有错误

416. 下面程序【 】。

```
main() {
 int a[3]={0};
 int i;
 for(i=0;i<3;i++)scanf("%d",&a[i]);
 for(i=1;i<4;i++)a[0]=a[0]+a[i];
 printf("%d\n",a[0]);
}
```

A. 没有错误

B. 第 3 行有错误

C. 第 5 行有错误

D. 第 6 行有错误

417. 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为【 】。

A. i*m+j

B. j*m+i

C. i*m+j-1

D. i*m+j+1

418. 对以下说明语句 int a[10]={6,7,8,9,10};的正确理解是【 】。

A. 将 5 个初值依次赋给 a[1]至 a[5]

B. 将 5 个初值依次赋给 a[0]至 a[4]

C. 将 5 个初值依次赋给 a[6]至 a[10]

D. 因为数组长度与初值的个数不同，所以此语句不正确

419. 以下不正确的定义语句是【 】。

A. double x[5]={2.0,4.0,6.0,8.0,10.0};

B. int y[5]={0,1,3,5,7,9};

C. char c1[]={'1','2','3','4','5'};

D. char c2[]={'\x10','\xa','\x8'};

420. 若有说明：int a[][3]={1,2,3,4,5,6,7};则 a 数组第一维的大小是【 】。

A. 2

B. 3

C. 4

D. 无确定值

421. 若二维数组 a 有 m 列，则在 a[i][j]前面的元素个数为【 】。

A. j*m+i

B. i*m+j

C. i*m+j-1

D. i*m+j+1

422. 定义如下变量和数组：

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

423. 则下面语句的输出结果是【 】。

424. for (k=0;k<3;k++) printf("%d",a[k][2-k]);

A. 3 5 7

B. 3 6 9

C. 1 5 9

D. 1 4 7

425. 若有以下程序段：

```
int a[]={4,0,2,3,1},i,j,t;
for(i=1;i<5;i++)
{t=a[i];j=i-1;
while(j>=0&&t>a[j])
{a[j+i]=a[j];j--;}
a[j+1]=t;}
```

426. 则该程序段的功能是【 】。

A. 对数组 a 进行插入排序（升序）

B. 对数组 a 进行插入排序（降序）

C. 对数组 a 进行选择排序（升序）

D. 对数组 a 进行选择排序（降序）

427. 以下不正确的定义语句是【 】。

A. `int a[1][4]={1,2,3,4,5};`

B. `float x[3][]={{1},{2},{3}};`

C. `long b[2][3]={{1},{1,2},{1,2,3}};`

D. `double y[][3]={0};`

428. 下面程序的运行结果是【 】。

```
main()
{int a[6][6].i.j;
for(i=1;i<6;i++)
for(j=1;j<6;j++)
a[i][j]=(i/j)*(j/i);
for(i=1;i<6;i++)
{for(j=1;j<6;j++)
printf("%2d",a[i][j]);
printf("\n";)}
}
```

A.

```
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
```

B.

```
0 0 0 0 1
0 0 0 1 0
0 0 1 0 0
0 1 0 0 0
1 0 0 0 0
```

C.

```
1 0 0 0 0
0 1 0 0 0
0 0 1 0 0
0 0 0 1 0
0 0 0 0 1
```

D.

```
1 0 0 0 1
0 1 0 1 0
0 0 1 0 0
```

```
0 1 0 1 0
1 0 0 0 1
```

429. 下面程序的运行结果是【 】。

```
main()
{int a[6],i;
for(i=1;i<6;i++)
{a[i]=9*(i-2+4*(i>3))%5;
printf("%2d",a[I]);
}
}
```

A. -4 0 4 0 4

B. -4 0 4 0 3

C. -4 0 4 4 3

D. -4 0 4 4 0

430. 在 C 语言中，二维数组的定义方式为【 】。

431. 在 C 语言中，二维数组元素的内存中的存放顺序是【 】。

432. 若有定义 double x[3][5]; 则 x 数组中行下标的下限为【 】，列下标的上限为【 】。

433. 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为：【 】（假设 a[0][0]位于数组的第一个位置上。）

434. 若有定义：int a[3][4]={{1,2},{0},{4,6,8,10}}; 则初始化后，a[1][2]得到的初值是【 】，a[2][1]得到的初值是【 】。

435. 下面程序以每行 4 个数据的形式输出 a 数组，请填空。

```
#define N 20
main()
{ int a[N],i;
for (i=0;i<N;i++)scanf("%d", 【 】);
for (i=0;i<N;i++)
{ if(【 】) 【 】
printf("%3d",a[i]);
}
```

```
printf ("\n");
}
```

436. 下面程序将二维数组 a 的行和列元素互换后存到另一个二维数组 b 中。请填空。

```
main()
{ int a[2][3]={{1,2,3},{4,5,6}};
int b[3][2],i,j;
printf("arrar a:\n");
for(i=0;i<=1;i++)
{for(j=0; 【 】;j++)
{printf("%5d",a[i][j]);
【 】;
}
printf("\n");
}
printf("array b:\n");
for(i=0; 【 】;i++)
{ for(j=0;j<=1;j++)
printf("%5d",b[i][j]);
printf("\n");
}
}
```

437. 下面程序可求出矩阵 a 的两条对角线上的元素之和。请填空。

```
main()
{ int a[3][3]={1,3,6,7,9,11,14,15,17},sum1=0,sum2=0,i,j;
    for (i=0;i<3;i++)
    for(j=0;j<3;j++)
    if(i==j)sum1=sum1+a[i][j];
    for(i=0;i<3;i++)
    for(【 】; 【 】;j--)
    if((i+j)==2) sum2=sum2+a[i][j];
    printf("sum1=%d,sum2=%d\n",sum1,sum2);
}
```

438. 下面程序的运行结果是【 】。

```
main()
{ int a[5][5],i,j,n=1;
for(i=0;i<5;i++)
for(j=0;j<5;j++)
```

```
a[i][j]=n++;
printf("The result is:\n");
for(i=0;i<5;i++)
{for(j=0;j<=i;j++)
printf("4%d",a[i][j]);
printf("\n");
}
}
```

439. 下面程序可求出矩形阵 a 的主对角线上的元素之和。请填空。

```
main()
{int a=[3][3]={1,3,5,7,9,11,13,15,17},sum=0,I,j;
for (i=0;i<3;i++)
for(j=0;j<3;j++)
if (【 】) sum=sum+【    】;
printf("sum=%d\n",sum);
}
```

440. 下面程序的功能是生成并打印某数列的前 20 项，该数列第 1，2 项分别为 0 和 1，以后每个奇数编号的项是前两项之和，偶数编号的项是前两项差的绝对值。生成的 20 个数存在一维数组 x 中，并按每行 4 项的形式输出。请填空。

```
main ()
{ int x[21] ,i ,j ;
x[1]=0; x[2]=1;
i=3;
do{
x[ i ]=【 】;
x[i+1]=【 】;
i=【 】;
} while (i <=20);
for (i=1;i<=20;i++)
{ printf("%5d",x[ i ]);
if( i %4= =0)
printf("\n");
}
}
```

441. 下面程序的运行结果是【 】。

```
main()
{
```

```
    int i, j, a[2][3] = { { 2,4,6 },{ 8,10,12 } };
    printf("The origianal array is :\n");
    for (i = 0; i <= 2; i++)
    {
        for (j = 0; j <= 3; j++)
            printf(" %4d", a[i][j]);
        printf("\n");
    }
    printf("\nthe result is : \n ");
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 2; j++)
            printf(" % 4d ", a[i][j]);
        printf(" \ n ");
    }
}
```

442. 下面程序的功能是输入 5 个整数，找出最大数和最小数所在的位置，并把二者对调，然后输出调整后的 5 个数。请填空。

```
main()
{int a[5],max,min,i,j,k;
for(i=0;i<5;i++)
scanf("%d",&a[i]);
min=a[0];
for(i=1;i<5;i++)
if(a[i]<min) {min=a[i];【     】;}
max=a[0];
for(i=1;i<5;i++)
if(a[i]>max) {max=a[i];【     】;}
[ a[j]=max;a[k]=min ]
print("\nThe position of min is:%3d\n",k);
printf("The position of max is:%3d\n",j);
for(i=0;i<5;i++)
printf("%5d",a[i]);
}
```

443. 下面程序的运行结果是【 】。

```
main()
{ int i,f[10];
f[0]=f[1]=1;
for(i=2;i<10;i++)
f[i]=f[i-2]+f[i-1];
```

```
for(i=0;i<10;i++)
{if(i%4==0) printf("\n");
printf("%3d",f[i]);}
}
```

444. 下面程序的功能是将二维数组 a 中每个元素向右移一列,最右一列换到最左一列,最后的数组存到另一数组 b 中，并按矩阵形式输出 a 和 b。请填空。

```
main()
{ int a[2][3]={4,5,6,1,2,3,},b[2][3];
inti,j;
printf("array a:\n");
for(i=0;i<=1;i++)
{for(j=0;j<3;j++)
{printf("%5d",a[i][j]);
【 】;
}
printf("\n");
}
for(【 】;i++) b[i]][0]=a[i][2];
printf("array b:\n");
for(i=0;i<2;i++)
{for(j=0;j<3;j++)
printf("%5d",b[i][j]);
【 】;
}
}
```

445. 下面程序的功能是统计年龄在 16～31 岁之间的学生人数。请填空。

```
main()
{int a[30],n,age,i;
for(i=0;i<30;i++)a[i]=0;
printf("enter the number of the students(<30)\n");
scanf("%d",&n);
printf("enter the age of each student:\n");
for(i=0;i<n;i++)
{scanf("%d",&age);【 】;}
printf("the result is\n");
printf("age number\n");
for(【 】i++)
printf("%3d %6d\n",i,a[i-16]);
}
```

446. 下面程序的功能是检查一个二维数组是否对称（即：对所有 i 和 j 都有 a[i][j]=a[j][i]）。请填空。

```
main()
{int a[4][4]={1,2,3,4,2,2,5,6,3,5,3,7,4,6,7,4};
int i,j,found=0;
for(j=0;j<4;j++)
for(【 】;i<4;i++)
if(a[j][i]!=a[i][j])
{【 】;bteak;}
if(found)
 printf("no");
else printf("yes");
}
```

447. 下面程序的运行结果是【 】。

```
main()
{int i=1,n=3,j,k=3;
int a[5]={1,4,5};
while (i<=n&&k>a[i])i++;
for(j=n-1;j>=i;j--)
a[j+1]=a[j];
a[i]=k;
for(i=0;i<=n;i++)
printf("%3d",a[i]);
}
```

448. 下面程序的运行结果是【 】。

```
main()
{int num_list[]={6,7,8,9},k,j,b,u=0,m=4,w;
    w=m-1;
    while(u<=w)
    {j=num_list[u];
        k=2; b=1;
        while(k<=j/2&&b)
        b=j%++k;
        if (b) printf("%d\n",num_list[u++]);
        else
        {num_list[u]=num_list[w];
            num_list[w--]=j;
        }
    }
```

```
}
```

449. 设数组 a 中的元素均为正数，以下程序是求 a 中偶数的个数和偶数的平均值。请填空。

```
main()
{int a[10]={1,2,3,4,5,6,7,8,9,10};
int k,s,i;
float ave;
for(k=s=i=0;i<10;i++)
{if(a[i]%2!=0)【    】;
s+=【    】;
k++;
}
if(k!=0){ave=s/k;printf("%d,%f\n",k,ave);}
}
```

450. 以下程序是求矩阵 a,b 的乘积，结果存入矩阵，请填空。

```
main()
{int a[3][2]={2,-1,-4,0,3,1};
int b[2][2]={7,-9,-8,10};
int i,j,s,c[3][2];
for(i=0;i<3;i++)
for(j=0;j<2;j++)
{for(【 】;k<2;k++)
s+=【 】;
c[i][j]=s;
}
for(i=0;i<3;i++)
{for(j=0;j<2;j++)
printf("%6d",c[i][j]);
【 】
}
}
```

451. 以下程序的功能是【 】。

```
main()
{ int num[10]={10,1,-20,-203,-21,2,-2,-2,11,-21};
int sum=0,i;
for (i=o;i<10;i++)
{if (num[i]>0)
sum=num[i]+sum;
```

```
}
printf("sum=%6d",sum);
}
```

452. 下面程序的运行结果是【 】。

```
main()
{int i,j,row,col,min;
int a[3][4]={{1,2,3,4},{9,8,7,6},{-1,-2,0,5}};
min=[0][0];
for (i=0;i<3;i++)
for (j=0;j<4;j++)
if (a[i][j]<min)
{ min=a[i][j]; row=i; col=j; }
printf("min=%d,row=%d,col=%d\n",min,row,col);
}
```

453. 若有以下输入 52↲，则下面程序的运行结果是【 】。

```
main()
{int a[8]={6,12,18,42,44,52,67,94};
int low=0,mid,high=7found,x;
found=0;
scanf("%d",&x);
while((low<=high)&&(found==0))
{mid=(low+high)/2;
if (x>a[mid])low=mid-1;
else if (x<a[mid])high=mid+1;
else
{found=1;break;}
}
if (found==1) printf("search successful ! the
index :%d\n",mid) ;
}
```

454. 下面程序的运行结果是【 】。

```
main()
{int a[9]={0,6,12,18,42,44,52,67,94};
int x=52, i, n=9,m;
i=n/2+1;
m=n/2;
while (m! =0)
{if (x<a[i])
{i=i-m/2-1;m=m/2;}
```

```
else if(x>a[i])
{i=i+m/2+1;m=m/2;}
else break;}
printf("the index is :%d",i);
}
```

455. 下面程序用"顺序查找法"查找数组 a 中是否存在某一关键字。请填空。

```
main()
{int a[8]={25,57,48,37,12,92,86,33};
int i,x;
scanf("%d",%x);
for (i=0;i<8;i++)
if (x==a[i])
{printf("found ! the index is : %d\n",--i);【 】;}
if (【 】)
{printf("can't found !");
}
```

456. 下面程序用"快速顺序查找法"查找数组 a 中是否存在某一关键。请填空.

```
main()
{int a[9]={25,57,48,37,12,92,86,33};
int i,x;
scanf("%d",&x);
【 】; i=0;
while（a[i]! =x）i++;
if（【 】）printf（"found! the index is : %d\n",i）;
else printf("can't found !\n");
}
```

457. 下面程序用插入法对数组 a 进行降序排序。请填空。

```
main()
{int a[5]={4，7，2，5，1};
int i,j,m;
for(i=1;i<5;i++)
{m=a[i];
j=【 】;
while(j>=0&&m>a[j])
{【 】;
j--;
}
【 】=m;
}
```

```
for(i=0;i<5;i++)
printf("%d",a[i]);
printf("\n");
}
```

458. 下面程序用"两路合并法"把两个已按升序排列的数组合并成一个升序数组。

请填空。

```
main()
{int a[3]={5,9,19};
int b[5]={12,24,26,37,48};
int c[10],i=0,j=0,k=0;
while(i<3&&j<5 )
if(【 】)
{c[k]=b[j];k++;j++;}
else
{c[k]=a[i];k++;i++}
while(【 】)
{c[k]=a[i];i++;k++}
while(【 】)
{c[k]=b[j];k++;j++}
for(i=0;i<k;i++)
printf("%3d",c[i]);
}
```

459. 下面程序的运行结果是【 】。

```
main()
{ int a[6][6],i,j;
    for(i=1;i<6;i++)
        for(j=1;j<6;j++)
            a[i][j]=(i/j)*(j/i);
    for(i=1;i<6;i++)
    { for(j=1;j<6;j++)
            printf("%2d",a[i][j]);
        printf("%\n");
    }
}
```

460. 若有以下输入 7 4 8 9 1 5↲,则下面程序的运行结果是【 】。

```
main()
{
    int a[6], i, j, k, m;
```

```
    for (i = 0, i< 6; i++)
        scanf("%d", &a[i]);
    for (i = 5; i >= 0; i--)
    {
        k = a[5];
        for (j = 4; j >= 0; j--)
            a[j + 1] = a[j];
        a[0] = k;
        for (m = 0; m< 6; m++)
            printf("%d", a[m]);
        printf("\n");
    }
}
```

461. 下面程序的运行结果是【 】。

```
main()
{int a[10]={1,2,3,4,5,6,7,8,9,10};
int k,s,i;
float ave;
for(k=s=i=0;i<10;i++)
{if (a[i]%2==0) continue;
s+=a[i];
k++;}
if(k!=0)
{ave=s/k;
printf("the number is :%d,the average is :%f\n",k,ave);}
}
```

462. 若有以下输入 3 1 2 3 2 2 2 1 1 3 0↲则下面程序的运行结果是【 】。

```
main()
{int a[4],x,i;
for(i=1;i<=3;i++) a[i]=0;
scanf("%d",&x);
while (x>0){ a[x]+=1;scanf("%d",&x);
for(i=1;i<=3;i++)printf("a[%2d]=%4d\n",i,a[i]);
}
```

463. 若有以下输入 5↲9 7 5 3 1 ↲5↲，则下面程的运行结果是【 】。

```
#define m 10
main()
{int a[m],x,i,n;
    printf("enter n (n<10):");
```

```
    scanf("%d",&n);
    for(i=1;i<=n;i++)
        scanf("%d",&a[i]);
    printf("enter n(n<10):");
    scanf("%d",&x);
    a[0]=x;a[i]=n;
    n=i++;
    for(i=1;i<=n;i++)
        printf("%3d",a[i]);
    printf("\n");
}
```

464. 下面程序的运行结果是【 】。

```
#define size 30
main( )
{ float a[size],b[size/5],sum;
    int i,k;
    for(k=2,i=0;i<size;i++)
    { a[i]=k; k+=2; }
    sum=0;
    for (k=0,i=0;i<=size;i++)
    {sum+=a[i];
    if((i-1)%5+1==5)
    {b[k]=sum/5;
    sum=0;k++;
    }
    }
    printf("the reasult is\n");
    for (i=0;i<size/5;i++)printf("%5.2f ",b[i]);
    printf("\n");
}
```

465. 求整型数组的最大值。

466. 找出二维数组的鞍点，即该位置上的元素是该行上的最大值，是该列上的最小值。二维数组也可能没有鞍点。

下面是 5×5 的螺旋方阵，编程生成 n×n 的螺旋方阵。 1 2 3 4 5 16 17 18 19 6 15 24 25 20 7 14 23 22 21 8 13 12 11 10 9

第13章 数组与指针 (2)

467. 经声明 int mat[10][30]后，以下说法正确的是（ ）

A. mat[3][7]与 mat[3][8]的内存地址相邻；

B. mat[3][7]与 mat[4][7]的内存地址相邻；

C. mat[3][30]与 mat[4][0]的内存地址不一致；

D. mat[4][0]与 mat[120]的内存地址一致。

468. 经声明 int arr[10]后，以下说法正确的是（ ）

A. arr（数组名）对应的值是 arr 数组的第一个元素的值；

B. arr（数组名）对应的值是 arr 数组的第一个元素所在地址；

C. arr（数组名）的地址是 arr 数组的第一个元素所在地址；

D. arr（数组名）对应的值是 arr[1]所在的地址。

469. 以下程序有错，错误的原因是【 】。

```
main()
{int *p,i;char *q,ch;
p=&i;
q=&ch;
*p=40;
*p=*q;
}
```

A. p 和 q 的类型不一致，不能执行*p=*q;语句

B. *p 中存放的是地址值，因此不能执行*p=40;语句

C. q 没有指向具体的存储单元，所以*q 没有实际意义

D. q 虽然指向了具体的存储单元，但该单元中没有确定的值，所以不能执行*p=*q;语句

470. 已有定义 int k=2;int *ptr1,*ptr2; 且 ptr1 和 ptr2 均已指向变量 k，下面不能正确执行的赋值语句是【 】。

A. k=*ptr1+*ptr2;

B. ptr2=k;

C. ptr1=ptr2;

D. k=*ptr1*(*ptr2);

471. 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

472. 若已定义 int a=5;下面对（1）int *p=&a;，（2）*p=a; 两个语句的正确解释是【 】。

A. 语句(1)和(2)中的 *p 含义相同，都表示给指针变量 p 赋值

B. (1)和(2)语句的执行结果，都是把变量 a 的地址值赋给指针变量 p

C. (1)在对 p 进行说明的同时进行初始化，使 p 指向 a；(2)将变量 a 的值赋给指针变量 p

D. (1)在对 p 进行说明的同时进行初始化，使 p 指向 a；(2)将变量 a 的值赋于*p

473. 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

474. 若有说明 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;

475. 若有说明 int *p1,*p2,m=5,n;以下均是正确赋值语句的选项是【 】。

A. p1=&m;p2=&p1;

B. p1=&m;p2=&n;*p1=*p2;

C. p1=&m;p2=p1

D. p1=&m;*p2=*p1;

476. 已有变量定义和函数调用语句：int a=25; print_value(&a);下面函数的正确输出结果是【 】。

```
void print_value(int *t)
{ print("%d\n",++*x);}
```

A. 23

B. 24

C. 25

D. 26

477. 设 p1 和 p2 是指向同一个字符串的指针变量，c 为字符变量，则以下不能正确执行的赋值语句是【 】。

A. c=*p1+p2;

B. p2=&c

C. p1=p2

D. c=*p1*(*p2);

478. 若有以下定义 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)

479. 若有以下定义 int a[5],*p=a;，则对 a 数组元素地址的正确引用是【 】。

A. p=5

B. *a+1

C. &a+1

D. &a[0]

480. 若有定义：int a[2][3]; 则对数组 a 的第 i 行第 j 列（假设 i，j 已正确声明并赋值）元数值的地址为【 】。

A. *(*(i+j)+j)

B. (a+i)[j]

C. *(a+i+j)

D. *(a+i)+j

481. 若有定义：int a[2][3]; 则对数组 a 的第 i 行第 j 列（假设 i，j 已正确声明并赋值）元数值的正确引用为【 】。

A. *(a[i]+j)

B. (a+i)

C. *(a+j)

D. a[i]+j

482. 若有以下定义和语句 int a[2][3],(*p)[3]; p=a;，则对 a 数组元素地址的不正确引用是【 】。

A. *(p+2)

B. p[2]

C. p[1]+1

D. (p+1)+2

483. 若有以下定义和语句 int a[2][3],(*p)[3]; p=a;，则对 a 数组元素地址的正确引用是【 】。

A. (p+1)[0]

B. *(*(p+2)+1)

C. *(p[1]+1)

D. p[1]+2

484. 若有定义 int a[5]; 则 a 数组中首元素的地址可以表示为【 】。

A. &a

B. a+1

C. a

D. &a[1]

485. 若有定义：int (*p)[4]; 则标识符 p【 】。

A. 是一个指向整型变量的指针

B. 是一个指针数组名

C. 是一个指针，它指向一个含有四个整型元素的一维数组

D. 定义不合法

486. 若有以下定义和赋值语句 int s[2][3]={0}，(*p)[3]; p=s;，则对 s 数组的第 i 行第 j 列（假设 i，j 已正确说明并赋值）元素地址的非法引用为【 】。

A. *(*(p+i)+j)

B. *(p[i]+j)

C. (p+i)+j

D. (*(p+i))[j]

487. 以下与 int *q[5];等价的定义语句是【 】。

A. int q[5];

B. int *q;

C. int *(q[5]);

D. int (*q)[5];

488. 阅读以下程序：

```
main()
{int a[10]={2,4,6,8,10,12,14,16,18,20}, *p;
p=a;
printf("%x\n",p);
printf("%x\n",p+9);
}
```

若第一个 printf 语句输出的是 ffca（16 位机器），则第二个 printf 语句的输出是【 】。

A. ffdd

B. ffdc

C. ffde

D. ffcd

489. 以下程序能找出数组 x 中的最大值和该值所在的元素下标，该数组元素从键盘输入。请选择填空。

```
main()
{int x[10], *p1, *p2,k;
for(k=0;k<10;k++) scanf("%d",x+k);
for(p1=x,p2=x;p1-x<10;p1++)
if(*p1>*p2) p2=【 】;
printf("MAX=%d,INDEX=%d\n",*p2,【 】);
}
```

490. 第 1 个空格应填写：

A. p1

B. p2[p1]

C. x[p2]

D. x-p1

491. 第 2 个空格应填写：

A. p1-x

B. p1

C. p2-x

D. x-p2

492. 请结合以下实例，说明*符号的三种用法，以及&符号的两种用法。（在 GCC 检查校对自己的答案，誊写到作业纸上。）

a. int *pointer;

b. value = *pointer;

c. value = value * 3;

d. pointer = &value;

e. **选做** value = value & 3;

493. 指针运算中，*和&互为逆运算的说法是否正确？如果正确，请举例说明；如果不正确，请举出反例。

494. 请结合课件bounds.c（第9页）实例，说明数组向上和向下越界的后果。（在 GCC 检查校对自己的答案，誊写到作业纸上。）

495. 经声明 int arr[10][30]后，（在 GCC 检查校对自己的答案，誊写到作业纸上。）

a. &arr、&arr[0]、&arr[-1][30]、&arr[0][0]与 arr 的值分别是否一致？为什么？

b. 上课谈到 arr 和&arr 的值是相同的，但二者仍有区别。若 ptarr = &arr; parr = arr;不报错且无告警，应如何声明指针 ptarr 和 parr？

c. 试比较**(arr+1)、*(*arr+1)、**arr+1 的不同。

第14章 数组与指针 (3)

496. 若有以下定义：int x[4][3]={1,2,3,4,5,6,7,8,9,10,11,12}; int(*p)[3]=x; 则能够正确表示数组元素 x[1][2]的表达式是【 】。

A. *((*p+1)[2])

B. (*p+1)+2

C. *(*(p+5))

D. *(*(p+1)+2)

497. 以下说法正确的是（ ）

A. 经声明 long double *p 后，运行 p++，则前后 p 的值相差 12；

B. 经声明 long double p[10]后，运行 p++，则前后 p 的值相差 12；

C. 函数声明 void print(double p[][])是合法的；

D. 函数声明 void print(double p[10][])是合法的。

498. 下面程序的运行结果是【 】。

```
main()
{int x[5]={2,4,6,8,10}, *p,**pp;
p=x;
pp=&p;
printf("%d",*(p++));
printf("%3d\n",**pp);
}
```

A. 4 4

B. 2 4

C. 2 2

D. 4 6

499. 下面程序的运行结果是【 】。

```
#include "stdio.h"
fun(int **a,int p[2][3])
```

```
{ **a=p[1][1]; }
main()
{ int x[2][3]={2,4,6,8,10,12},*p;
p=(int *) malloc(sizeof (int) );
fun(&p,x);
printf("%d\n",*p);
}
```

A. 10

B. 12

C. 6

D. 8

500. 以下程序输出的结果是【 】

```
void main()
{ int a=5,*p1,**p2;
p1=&a,p2=&p1;
(*p1)++;
printf("%d\n",**p2);
}
```

A. 5

B. 4

C. 6

D. 不确定

501. 设计做相互交换 a 和 b 的函数 swap(a, b)，分别以普通变量、指针、数组、字符串作为参量的类型，其运行结果有何不同？

502. 对整型数字 val 在使用 scanf 时需要加上 & 符号，而字符串 names 不得加上 & 符号，请解释原因。

503. 指针可以加、减、比较，是否能够乘除？为什么？

504. 对于 double *ptr1 = arr; double *ptr2 = &arr[2];, ptr2-ptr1 的值是多少？

505. 请说明表达式*p++等价于(*p)++还是*(p++)?

506. Suppose you have these declarations（声明）:

```
float rootbeer[10], things[10][5], *pf, value = 2.2;
int i = 3;
```

Identify each of the following statements as valid or invalid:

a. `rootbeer[2] = value;`

b. `scanf("%f", &rootbeer );`

c. `rootbeer = value;`

d. `printf("%f", rootbeer);`

e. `things[4][4] = rootbeer[3];`

f. `things[5] = rootbeer;`

g. `pf = value;`

h. `pf = rootbeer;`

507. Create an appropriate declaration for each of the following variables:

a. `rates` is an array of six floats .

b. `mat` is an array of three arrays of five integers.

c. `pstr` is a pointer to an array of 20 characters.

508. 经声明 `int arr[10];`与 `int arr2[39];`，并定义函数

```
void printsize(int ar[3])
{
    printf("%u ",sizeof ar);
}
```

后，运行 `printsize(arr)`和 `printsize(arr2)`，将如何输出？如果请将 `int ar[3]`替换为 `int *ar`，输出是否一样？请做解释。（在 GCC 检查校对自己的答案，誊写到作业纸上。）

509. 请说明函数声明 `int sum(const int arr[], int len);`中的 `const` 的作用。

第15章 字符串与字符串函数

510. 设有以下定义： char *cc[2]={"1234","5678"}; ，则正确的叙述是【 】。

A. cc 数组的两个元素中各自存放了字符串"1234"和"5678"的首地址

B. cc 数组的两个元素分别存放的是含有 4 个字符的一维字符数组的首地址

C. cc 是指针变量，它指向含有两个数组元素的字符型一维数组

D. cc 数组元素的值分别是"1234"和"5678"

511. 下面是对 S 的初始化，其中不正确的是【 】。

A. char s[5]={"abc"};

B. char s[5]={'a','b','c'};

C. char s[5]="";

D. char s[5]="abcdef";

512. 设有下面的程序段：chat s[]="china"; char *p; p=s;，则下列叙述正确的是【 】。

A. s 和 p 完全相同

B. 数组 s 中的内容和指针变量 p 中的内容相等

C. s 数组长度和 p 所指向的字符串长度相等

D. *p 与 s[0]相等

513. 下面判断正确的是【 】。

A. char * a="china"; 等价于 char *a;*a="china";

B. char str[10]={"china"}; 等价于 char str[10]; str[]={"china"};

C. char *s="china"; 等价于 char *s; s="china";

D. char c[4]="abc",d[4]="abc"; 等价于 char c[4]=d[4]="abc";

514. 设 char *s="\ta\017bc"; 则指针变量 s 指向的字符串所占的字节数是【 】。

A. 9

B. 5

C. 6

D. 7

515. 下面程序段中，for 循环的执行次数是【 】。

```
char *s="\ta\018bc";
for ( ;*s!='\0';s++) printf("*");
```

A. 2

B. 3

C. 6

D. 7

516. 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; scanf("%s",s);

517. 下面程序段的运行结果是【 】。

```
char *s="abcde";
s+=2; printf("%d",s);
```

A. cde

B. 字符'c'

C. 字符'c'的地址

D. 无确定的输出结果

518. 下面程序的运行结果是【 】。

```
char c[5]={'a','b','\0','c','\0'}; printf("%s",c);
```

A. 'a''b'

B. abc

C. ab c

D. ab

519. 设有一个名为 file1 的 C 程序，且已知命令行为为：FILE1 CHINA BEIJING SHANGHAI,则可得到以下运行结果 CHINA BEIJING SHANGHAI 的 C 程序为【 】。

A.

```
main（int argc, char *argv[]）
{while(--argc>0)
printf("%s%c",*++argv, (argv>1)?' ':'\n');
}
```

B.

```
main(int argx,char *argv[ ])
{while(argc-->1)
printf("%s\n",*argv);
}
```

C.

```
main(int argc,char *argv[ ])
{while(argc>0)
printf("%s%c\n",*++argv, (argc>1)?' ':'\n');
}
```

D.

```
main(int argc,char *argv[ ])
{while (argc>1)
{++argv;
printf ("%s\n", *argv);
--argc;
}
}
```

520. 下面程序的功能是按字典顺序比较两个字符串 s,t 的大小，如果 s 大于 t 则返回正值，等于则返回 0，小于则返回负值。请选择填空。

```
#include "stdio.h"
s(char *s, char * t)
{ for ( ; *s==* t; 【 】 ) 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);
}
```

第 1 个空格应填写：

A. s++

B. t++

C. s++;t++

D. t++,s++

521. 第 2 个空格应填写：

A. p,q

B. q,p

C. a,p

D. b,q

522. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{char str[]="cdalb";
abc(str);
puts(str);
}
abc(char *p)
```

```
{int i,j;
for(i=j=0;*(p+i)!='\0';i++)
if(*(p+i)>='d') {*(p+j)=*(p+i); j++;}
*(p+j)='\0';
}
```

A. dalb

B. cd

C. dl

D. c

523. 设有以下程序段：

```
char str[4][10]={"first", "second", "third", "fourth"},
*strp[4];
int n;
for（n=0；n<4；n++） strp[n]=str[n];
```

524. 若 k 为 int 型变量且 0≤k<4，则对字符串的不正确引用是【 】。

A. strp

B. str[k]

C. strp[k]

D. *strp

525. 若有说明：char *language[]={"FORTRAN","BASIC","PASCAL","JAVA","C"}; 则表达式*language[1]>*language[3] 比较的是【 】。

A. 字符 F 和字符 P

B. 字符串 BASIC 和字符串 JAVA

C. 字符 B 和字符 J

D. 字符串 FORTRAN 和字符串 PASCAL

526. 下面程序的功能是用递归法将一个整数存放到一个字符数组中。存放时按逆序存放。如 483 存放成"384"，请选择填空。

```
#include<stdio.h>
```

```
void convert (char *a, int n)
{ int i;
if((i=n/10)!=0) convert (【 】, i);
*a=【 】;
}
main()
{int number;
char str[10]=" ";
scanf("%d", & number);
convert(str,number);
puts(str);
}
```

527. 第 1 个空格应填写：

A. a++

B. a+1

C. a--

D. a-1

528. 第 2 个空格应填写：

A. n/10

B. n%10

C. n/10+'0'

D. n%10+'0'

529. 对两个数组 a 和 b 进行如下初始化

```
char a[]="ABCDEF";
char b[]={'A','B','C','D','E','F'};
```

530. 则以下叙述正确的是【 】。

A. a 和 b 数组完全相同

B. a 和 b 长度相同

C. a 和 b 中都存放字符串

D. a 数组比 b 数组长度长

531. 有两个字符数组 a、b，则以下正确的输入语句是【 】。

A. gets(a,b);

B. scanf("%s%s",a,b);

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

D. gets("a"),gets("b");

532. 下面程序的功能是从键盘输入一行字符，统计其中有多少个单词，单词之间用空格分隔。请选择填空。

```
#include<stdio.h>
main()
{char s[80],c1,c2=' ';
int i=0,num=0;
gets(s);
while(s[i]!='\0')
{c1=s[i];
if(i==0) c2=' ';
else c2=s[i-1];
if(【     】) num++;
i++;
}
printf("There are %d words.\n",num);
}
```

A. c1==' '&&c2==' '

B. c1!=' '&&c2==' '

C. c1==' '&&c2!=' '

D. c1!=' '&&c2!=' '

533. 下面程序的运行结果是【 】。

```
#include <stdio.h>
main()
{char ch[7]={"12ab56"};
int i,s=0;
for(i=0;ch[i]>='0'&&ch[i]<='9';i+=2)
s=10*s+ch[i]-'0';
printf("%d\n",s);
}
```

A. 1

B. 1256

C. 12ab56

D. 12

534. 以下程序的输出结果是【 】。

```
main() {int s[10]={1,2,3,4,5,6,7,8,9,10},*p=s; printf("%d\n",*(p+4)); }
```

A. 4

B. 5

C. 6

D. 14

535. 当运行以下程序时，从键盘键入：ab↲c↲def↲，则下面程序的运行结果是【 】。

```
#include <stdio.h>
#define N 6
main()
{char c[N];
int i=0;
for(;i<N;c[i]=getchar(),i++);
for(i=0;i<N;i++) putchar(c[i]);
}
```

A. abcdef↲b↲c↲

B. ab↲c↲d↲

C. ab↲cd↲f↲

D. ab↲c↲def↲

536. 当运行以下程序时，从键盘键入：AhaMA↲Aha↲，则下面程序的运行结果是【 】。

```
#include<stdio.h>
#define N 6
main()
{char s[80],c='a';
```

```
int i=0;
scanf("%s",s);
while(s[i]!=0)
{if(s[i]==c) s[i]=s[i]-32;
else if(s[i]==c-32) s[i]=s[i]+32;
i++;
}
put(s);
}
```

A. ahAMa

B. `AhAMa``

C. AhAMa ahA

D. ahAMa ahA

537. 下面程序的运行结果是【 】。

```
#include<stdio.h>
#include<string.h>
main()
{char a[80]="AB",b[80]="LMNP";
int i=0;
strcat(a,b);
while(a[i++]!='\0') b[i]=a[i];
puts(b);
}
```

A. LB

B. ABLMNP

C. AB

D. LBLMNP

538. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{char str[]="SSWLIA",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. SWW*

B. S*

C. SW*A

D. W*

539. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{ char a[ ]="morning",t;
int i,j=0;
for(i=1;i<7;i++) if(a[j]<a[i]) j=i;
t=a[j]; a[j]=a[7];
a[7]=a[j]; puts(a);
}
```

A. mogninr

B. mo

C. morning

D. morning

540. 下面程序的功能是从键盘接收一个字符串，然后按照字符顺序从小到大进行排序，并删除重复的字符。请选择填空。

```
#include<stdio.h>
#include<string.h>
main()
{ char string[100],*p,*q*,*r, c;
printf("Please input a string:");
gets(string);
for (p=string. *p; p++)
{for(q=r=p;*q;q++) if(【 】) r=q;
if (【 】) { c=*r; *r=*p; *p=c;}
```

```
}
for (p=string;*p;p++)
{for(q=p; *p==*q; q++);
strcpy(【 】,q );
}
printf("result:%s\n", string);
}
```

541. 第 1 个空格应填写：

A. *r>*q

B. *r>*p

C. r>q

D. r>p

542. 第 2 个空格应填写：

A. r==q

B. r!=q

C. p!=q

D. r!=p

543. 第 3 个空格应填写：

A. p++

B. p

C. p-1

D. p+1

544. 下面程序的运行结果是【 】。

```
#include<stdio.h>
#include<string.h>
main()
{char *p1,*p2,str[50]="abc";
p1="abc"; p2="abc";
strcpy(str+1,strcat(p1,p2));
printf("%s\n",str);}
```

A. abcabcabc

B. bcabcabc

C. aabcabc

D. cabcabc

545. 下面程序的运行结果是【 】。

```
#include<ctype.h>
fun(char*p)
{int i,t; char ts[81];
for(i=0,t=0;p[i]!='\0';i+=2)
if(! isspace(*(p+i))&&(*(p+i)!='a'))
ts[t++]=toupper(p[i]);
ts[t]='\0';
strcpy(p,ts);
}
main()
{char str[81]={"a b c d ef g"};
fun(str);
puts(str);
}
```

A. abcdeg

B. bcde

C. ABCDE

D. BCDE

546. 下面程序的功能是将一个整数字符串转换为一个整数，如"-1234"转换为-1234，请选择填空。

```
#include <stdio.h>
#include <stdio.h>
main()
{char s[6];
int n;
gets(s);
if(*s=='-')
n=-chnum(s+1);
else
n=chnum(s);
```

```
printf("%d\n",n);
}
chnum(char*p)
{int num=0,k,len,j;
len=strlen(p);
for(;【  】;p++)
{k=【  】;
j=(--len);
while(【  】)
{k=k*10;}
num=num+k;
}
return (num)
}
```

547. 第 1 个空格应填写：

A. *p!='\0'

B. *(++p)!='\0'

C. *(p++)!='\0'

D. len!=0

548. 第 2 个空格应填写：

A. *p

B. *p+'0'

C. *p-'0'

D. *p-32

549. 第 3 个空格应填写：

A. --j

B. j-->0

C. –len

D. len-->0

550. 下面程序的功能是统计子串 substr 在母串 str 中出现的次数。请选择填空。

```
#include <stdio.h>
main()
{char str[80],substr[80];
int n;
gets(str);
gets(substr);
printf("%d\n",count(str,substr));
}
count(char*str,char*substr)
{int i,j,k,num=0;
for(i =0; 【  】; i ++)
for(【  】,k=0;substr[k]==str[j];k++,j++)
if(substr[【  】]=='\0')
{num++;break;}
return(num);
}
```

551. 第 1 个空格应填写：

A. str[i]==substr[i]

B. str[i]!='\0'

C. str[i]=='\0'

D. str[i]>substr[i]

552. 第 2 个空格应填写：

A. j=j+1

B. j=i

C. j=0

D. j=1

553. 第 3 个空格应填写：

A. k

B. k++

C. k+1

D. ++k

554. 以下正确的叙述是【 】。

A. C 语言允许 main 函数带形参，且形参个数和形参名均可由用户指定

B. C 语言允许 main 函数带形参，形参名只能是 argc 和 argv

C. 当 main 函数带有形参时，传给形参的值只能从命令行列中得到

D. 若有说明：main(int argc,char *argv),则形参 argc 的值必须大于 1

555. main 函数的正确说明形式是【 】。

A. main(int argc,char *argv)

B. main(int abc,char **abv)

C. main(int argc,char argv)

D. main(int c,char v[ ])

556. **选做** 关于变长数组（VLA），下列说法正确的是：

A. 旧版本的 C 语言并不支持；

B. 数组声明后，其长度可以按需要改变；

C. 在声明求和函数时，可以使用 int sum(int ar[n], int n);，在函数中 sizeof(ar)为 n 的 4 倍；

D. 在声明求和函数时，可以使用 int sum(int ar[], int n);，在函数中 sizeof(ar)为 4（32 位系统）或 8（64 位系统）。

557. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{int i,j;
for(i=4;i>=1;i--)
{for(j=1;j<=i;j++) putchar('#');
for(j=1;j<=4-i;j++) putchar('*');
putchar('\n');
}
}
```

558. 下面程序从键盘输入的字符中统计数字字符的个数，用换行符结束循环。请填空。

```
int n=0,c;
c=getchar();
while(【    】)
{if(【    】) n++;
c=getchar();
}
```

559. 当运行以下程序时，从键盘键入 right?↲ ，则下面程序的运行结果是【 】。

```
#include<stdio.h>
main( )
{ char c;
while((c=getchar())! ='?') putchar(++c);
}
```

560. 分别写出下列情况 sizeof(str)和 strlen(str)的值，并作简要解释。（书写在草稿纸上，经GCC调试后将结果与同学讨论，修正并誊写在答题纸上）

- a. char str[40]="Good!";
- b. char str[4]="Good!";
- c. char str[4]="Good!"; str[4]='!';

561. 当执行语句 while (1) { scanf("%d ", &a); printf("%d\n", a); if (a < 0) return -1; }，输入字母 a，回车后输入 0 和回车，程序是否能够正确退出？

562. Create an appropriate declaration for each of the following variables:

a. psa is an array of 20 pointers to char.

563. 请写出表达式*"Hello"的值。

564. 请写出'$'和"$"分别需要多大存储空间。

565. 请写出 char fruit[3][7]={"Apple", "Pear", "Banana"};和 char* fruit [3]={"Apple", "Pear", "Banana"};的区别。

566. 设有字符串 char *src="Good";和 char *dst="Bad";，请解释字符串复制时，使用 strcpy(dst, src)和 dst=src 的区别。

567. 程序的命令行参数将写入 int main(int argc, char* argv[])函数的两个参数中，设 Windows 或 Ubuntu 下有可执行文件 map，说明分别运行 map 和 map Hello "Hello World!"时，argc 和 argv 的值。

568. 字符串"ab\n\012\\'"的长度是【 】。

569. 下面程序的功能是将字符串 a 中下标值为偶数的元素按从小到大排列，其他元素不变。请填空：

```
#include<stdio.h>
main()
{ char a[]="labchmfye",t ;
int i,j ;
for(i=0;i<7;i+=2)
for(j=i+2;j<9; 【     】 )
if (【     】)
{ t=a[i]; a[i]=a[j]; a[j]=t; j++; }
puts(a) ; printf("\n");
}
```

570. 下面程序的功能是在任意的字符串 a 中将与字符 c 相等的所有元素的下标值分别存放在整型数组 b 中. 请填空：

```
#include<stdio.h>
main()
{ char a[80];
int i,b[80],k=0;
gets(a) ;
for(i=0;a[i]!='\0' ;i++)
if( 【     】 ) { b[k]=i ; 【     】 ; }
for( i=0;i<k;i++) printf("%3d",b[i] );
}
```

571. 有十个字符串, 下面程序的功能是在每个字符串中,找出最大字符按一一对应的顺序放入一维数组 a 中,即第 i 个字符串中最大字符放入 a[i]中,输出每个字符串中的最大字符.请填空:

```
#include<stdio.h>
main()
{ char s[10][20];
int a[10],i, j;
for(i=0;i<10;i++) gets(s[i]);
for(i=0;i<10;i++)
{ 【     】 ;
for(j=1; s[i][j]!='\0' ;j++) if(a[i]<s[i][j] ) 【     】;
}
for(i=0;i<10;i++) printf("%d %c" , i, a[i] );
}
```

572. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{ int i ;
char a[]="{time" , b[]="tom" ;
for(i=0;a[i]!='\0'&&b[i]!='\0' ;i++)
if(a[i]==b[i])
if(a[i]>='a'&&a[i]<='z') printf("%c" ,a[i]-32);
else printf("%c",a[i]+32);
else printf("*");
}
```

573. 下面程序的运行结果是【 】。

```
#include<stdio.h>
main()
{ char a[2][6]={"sun","moon"};
int i,j,len[2];
for(i=0;i<2;i++)
{ for(j=0;j<6;j++)
if(a[i][j]=='\0')
{ len[i]=j ; break; }
printf("%6s:%d\n",a[i],len[i]);
}
}
```

574. 下面程序的运行结果是【 】。

```
#include <stdio.h>
main()
{int i,r;
char s1[80]="bus";
char s2[80]="book";
for(i=r=0;s1[i]!='\0'&&s2[i]!='\0';i++)
if(s1[i]==s2[i]) i++;
else {r=s1[i]-s2[i]; break;}
printf("%d",r);
}
```

575. 下面程序的运行结果是【 】。

```
#include"stdio.h"
#define len 4
main()
{int j,c;
static char n[2][len+1]={"8980","9198"};
for(j=len-1;j>=0;j--)
{c=n[0][j]+n[1][j]-2*'0';
n[0][j]=c%10+'0';
n[0][j]=c%10+'0';
}
for(j=0;j<=1;j++) puts(n[j]);
}
```

576. 当运行以下程序时,从键盘输入 aabd↲,则下面程序的运行结果是【 】。

```
#include "stdio.h"
main()
{char s[80];
int i=0;
gets(s);
while(s[i]!='\0')
{if(s[i]<='z'&&s[i]>='a')
s[i]='z'+'a'-s[i];
i++;
}
puts(s);
}
```

577. 当运行以下程序时,从键盘输入:book↲cut↲game↲page↲ ,则下面程序的运行结果是【 】。

```
#include<stdio.h>
#include<string.h>
main()
{int i;
char str[10],temp[10]="control";
for(i=0;i<4;i++)
{gets(str);
if(strcmp(temp,str)<0) strcpy(temp,str);
}
puts(temp);
}
```

第16章 存储类，链接与内存管理

578. 凡是函数中未指定存储类别的局部变量，其隐含的存储类别为【 】。

A. 自动(auto)

B. 静态(static)

C. 外部(extern)

D. 寄存器(register)

579. 在一个 C 程序文件中，若要定义一个只允许本源文件中所有函数使用的全局变量，则该变量需要使用的存储类别是【 】。

A. extern

B. register

C. auto

D. static

580. 以下说法不正确的是【 】。

A. 同一文件内的不同函数可以包含同名的 goto 标记；

B. 声明变量为 extern 时，不可以对变量进行初始化；

C. 声明变量为 static 时，在其外部不可以声明该变量为 extern 的。

D. const 和 extern 不可共用。

581. 对于 `int *p=(int *)malloc(40*sizeof(int));`:

a. 该段内存区域在何时被收回?

b. 执行上述声明语句时，使用了多少字节内存?（64 位应用程序）

c. 若想使用二维数组的形式访问 p，即：p[3][2]而非 p[3*4+2]访问数组的第14个元素，应如何修改该语句？如有余力，请提供2种方法。

582. Which functions know each variable in the following?（下列每个变量对哪些函数是可见的？） Are there any errors?

```
/* file 1 */
int daisy;
int main(void)
{
    int lily;
    ...;
}
int petal()
{
    extern int daisy, lily;
    ...;
}
/* file 2 */
extern int daisy;
static int lily;
int rose;
int stem()
{
    int rose;
    ...;
}
void root()
{
    ...;
}
```

583. **选作** 在 2048 中实践本节课所讲的 extern、static 等内容。

第17章 文件输入输出

584. 系统的标准输入文件是指【 】。

A. 键盘

B. 显示器

C. 软盘

D. 硬盘

585. 以下可作为函数 fopen 中第一个参数的正确格式是【 】。

A. "c:user\text.txt"

B. "c:\user\text.txt"

C. "\user\text.txt"

D. "c\\user\\text.txt"

586. 若执行 fopen 函数时发生错误，则函数的返回值是【 】。

A. 地址值

B. 0

C. 1

D. EOF

587. 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，则文件方式字符串应是【 】。

A. "ab+"

B. "wb+"

C. "rb+"

D. "ab"

588. 若以"a+"方式打开一个已存的文件，则以下叙述正确的是【 】。

A. 文件打开时，原有文件内容不被删除，位置指针移到文件末尾，可作添加 和读操作

B. 文件打开时，原有文件内容不被删除，位置指针移到文件末尾，可作重写和读操作

C. 文件打开时，原有文件内容被删除，只可作写操作

D. 以上各种说法皆不正确

589. 当顺利执行了文件关闭操作时，fclose 函数的返回值是【 】。

A. -1

B. TURE

C. 0

D. 1

590. 已知函数的调用形式：fread(buffer,size,count,fp);，其中 buffer 代表的是【 】。

A. 一个整型变量，代表要读入的数据项总数

B. 一个文件指针，指向要读的文件

C. 一个指针，指向要读入数据的存放地址

D. 一个存储区，存放要读的数据项

591. fscanf 函数的正确调用形式是【 】。

A. fscanf(fp,格式字符串，输出表列);

B. fscanf(格式字符串，输出表列，fp);

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);

592. fwrite 函数的一般调用形式是【 】。

A. fwrite(buffer,count,size,fp);

B. fwrite(fp,size,count,buffer);

C. fwrite(fp,count,size,buffer);

D. fwrite(buffer,size,count,fp);

593. fgetc 函数的作用是从指定文件读入一个字符，该文件的打开方式必须是【 】。

A. 只写

B. 追加

C. 读或读写

D. 答案 B 和 C 都正确

594. 若调用 fputc 函数输出字符成功，则其返回值是【 】。

A. EOF

B. 1

C. 0

D. 输出的字符

595. 阅读以下程序及对程序功能的描述，其中正确的描述是【 】。

```
#include <stdio.h>
main()
{
    FILE *in, *out ;
    char ch,infile[10],outfile[10] ;
    scanf("%s",infile) ;
    printf("Enter the infile name:\n") ;
    scanf("%s",outfile) ;
    if(in=fopen(infile,"r"))==NULL)
    {
        printf("cannot open infile\n") ;
        exit(0) ;
    }
    if((out=fopen(outfile,"w"))==NULL)
    {
        printf("cannot open outfile\n") ;
        exit(0) ;
    }
```

```
    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和二为一的功能

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且在屏幕上输出

596. 函数调用语句: fseek(fp,-20L,2); 的含义是【 】。

A. 将文件位置指针移动到距离文件偷 0 个字节出

B. 将文件位置指针从当前位置向后移动 20 个字节

C. 将文件位置指针从文件末尾出向后退 20 个字节

D. 将文件位置指针移动到离当前位置 20 个字节处

597. 利用 fseek 函数的正确调用形式是【 】。

A. 改变文件的位置指针

B. 文件的顺序读写

C. 文件的随机读写

D. 以上答案均正确

598. fseek 函数的正确调用形式是【 】。

A. fseek(文件类型指针,起始点,位置量);

B. fseek(fp,位置量,起始点);

C. fseek(位置量,起始点,fp);

D. fseek(起始点,位置量,文件类型指针);

599. 函数 rewind 的作用是【 】。

A. 使位置指针重新返回文件的开头

B. 将位置指针指向文件中所要求的特定位置

C. 使位置指针指向文件的末尾

D. 使位置指针自动移动到下一个字符位置

600. 函数 ftell(fp)的作用是【 】。

A. 得到流式文件的当前位置

B. 移动流式文件的位置指针

C. 初始化流式文件的位置指针

D. 以上答案均正确

601. 下面程序实现人员登录.即每当键盘接收一个姓名,便在文件"member.dat"中寻找.若此姓名已经存在,则显示相应信息;若文件中没有该姓名,字将其存入文件(若文件"member.dat"不存在,应该在磁盘上建立一个新文件).当输入姓名按回车键或处理过程中出现错误时程序结束.请从下面对应的一组选项中选出正确的内容填入。

```
#include <stdio.h>
main()
{
FILE *fp;
int flag;
char nname[20],data[20];
if((fp=fopen("member.dat",【    】))==NULL)
{
printf("Open file error\n");
exit(0);
}
do
{
    printf("Enter name:");
    【    】;
    if(strlen(name)==0)
    break;
    strcat(name,"\n");
    rewind(fp);
    flag=1;
    while(flag&&((fgets(data,30,fp)!=NULL)))
    if(strcmp(data,name)==0)
        flag=0;
```

```
    if(flag)
        fputs(name,fp);
    else
        printf("tThis name has been existed!\n");
}
while(【 】);/*读写正确就循环*/
fclose(fp);
}
```

上述第 1 个空格填写【 】。

A. "w"

B. "w+"

C. "r+"

D. "a+"

602. 上述第 2 个空格填写【 】。

A. fgets(name)

B. gets(name)

C. scanf(name)

D. getc(name)

603. 上述第 3 个空格填写【 】。

A. ferror(fp)==0

B. ferror(fp)==1

C. ferror(fp)!=0

D. !(ferror(fp)==0)

604. 在执行 fopen 函数时，ferror 函数的初值是【 】。

A. TURE

B. -1

C. 1

D. 0

605. 以下 read 函数的调用形式中，参数类型正确的是【 】。

A. read(int fd,char*buf,int count)

B. read(int*buf,int fd,int count)

C. read(int fd,int count,char*buf)

D. read(int count,char*buf,int fd)

606. 函数 fseek 用来移动文件的位置指针，其调用形式是【 】。

A. fseek(位移方向,位移量,文件号);

B. fseek(文件号,位移量,起始点);

C. fseek(文件号,起始点,位移量);

D. fseek(文件号,位移方向,位移量);

607. 在 C 程序中，文件可以用【 】方式存取，也可以用【 】方式存取。

608. 在 C 语言中，文件的存取是以【 】为单位的，这种文件被称作【 】文件。

609. 函数调用语句：fgets(buf,n,fp);从 fp 指向的文件中读入【 】个字符放到 buf 字符数组中。函数值为【 】。

610. feof(fp)函数用来判断文件是否结束，如果遇到文件结束，函数值为【 】，否则为【 】。

611. What's the difference among the following?

```
printf("Hello, %s\n", name);
fprintf(stdout, "Hello, %s\n", name);
fprintf(stderr, "Hello, %s\n", name);
```

612. 设存在可执行文件 a.out，命令行 a.out hello > u.txt 中，main(int argc, char *argc[])函数第一个参量 argc 的值是多少？

613. 请在【 】处填入适当内容。

```
#include<stdio.h>
main(int argc,char*argv[(argv[1],"rb")])
{
```

```
    FILE*old,*new;
    char ch;
    if(argc!==3)
    {
    printf("You fougot to enter a filename\n);
    exit(0);
    }
    if((old=fopen【 】)==NULL)
    {
    printf("cannot open inlile\n");
    exit(0);
    }
    while(!feof(old))fputc(【 】,new);
    fclose(old);
    fclose(new);
}
```

614. 下面程序用变量 count 统计文件中字符的个数。请在【 】处填入适当内容。

```
#include<stdio.h>
main()
{FILE*fp;long count=0;
    if((fp=fopen("letter.dat",【 】))==NULL)
    {printf("cannot open file\n"); exit(0);}
    while(! feof(fp)) {【 】;【 】;}
    printf("count=%ld\n",count);
    fclose(fp);
}
```

615. 下面程序由终端键盘输入字符，存放到文件中，用！结束输入。请在【 】处填入适当内容。

```
#include<stdio.h>
main()
{FILE*fp;
    char ch,fname[10];
    printf("Input name of file\n"); gets(fname);
    if((fp=fopen(fname,"w"))==NULL)
    {printf("cannot open\n");exit(0);}
    printf("Enter data:\n");
    while(【    】!='!')fputc(【    】);
    fclose(fp);
}
```

616. 以下程序的功能是显示磁盘文件内容的十六进制代码和对应的字符，若代码对应的字符是非显示字符时则显示"."。请在【 】处填入适当内容。

```
#include<stdio.h>
main(int argc, char *argv[])
{
char letter[17];
int c,i,cnt;
FILE *fp;
if(【 】)
{
puts("Usage:dumpf filename");
exit();
}
if((fp=fopen(argv[1],"r"))==NULL)
{
printf("file %s can't opened\n",argv[1]);
exit();
}
cnt=0;
do
{
i=0;
printf("%06x:",cnt *16);
while((c=getc(fp))!=EOF)
{
printf("%02x",

c. ;
if(c< ' '\|\|c>0x7e)

letter[i]='.';
else

letter[i]=c;
if(++i==16)break;
}
letter[i]='\0';
if(i!=16)

for(;i<16;i++)
```

```
printf(" ")

printf("%s\n",【 】);
【 】;
}while(c!=EOF);
fclose(fp);
}
```

617. 磁盘上还有以下 C 语言源程序，经编译，连接后生成可执行文件，文件名为 ex12.exe。

```
#include<stdio.h>

main(int argc, char *argv[])
{FILE *fp;
    void sub();
    int i=1;
    while(--argc>0)
    if((fp=fopen(argv[i++],"r"))==NULL)
    {printf("Cannot open file!\n");
        exit(1);
    }
    else
    { sub(fp); fclose(fp); }
}
void sub(FILE *fp)
{char c;
    while((c=getc(fp))!='!' putchar(c+1);
}
```

618. 若在 DOS 提示符下键入：ex12 fi f2 f3，则程序的运行结果是【 】。

619. 以下程序的功能四将 C 语言源程序文件 exam.c 中用斜杠与星号括起来的非嵌套注释删除，然后存入文件 exam.out 中。请在【 】处填入适当的内容。

```
#include<stdio.h>

void delcomm(FILE *fp1,FILE *fp2)
{ int c,i=0;
while ((【 】)!=EOF)
if(c=='\n')
fprintf(fp2,"\n");
```

```
else
switch(i)
{case(0):
if(c=='/')i=1;
else fprintf(fp2,"%c",c);
break;
case(1):
if(c=='*')i=2;
else
{fprintf(fp2,"%c",c);
i=0;
}
break;
case (2):
if (c=='*') i=3;
break;
case(3):
I=(c=='/')【 】;
break;
}
}
main()
{FILE *fp1,fp2;
fp1=fopen("exam.c","r");
fp2=fopen("exam.out","w");
delcon(【 】);
fcloseall( );
}
```

620. 有 n 个班级参加 ns 比赛。下面的程序从文件 t.in 读入 n(<=30),ns(<=10)和全部班级各项得分，计算出各班总分,并按总分降序次序每个班级总分及各项目的得分输出到文件 t.out 中。为了避免排序可能要交换 score[i][k]和 socre[j][k](0<=k<=ns)。程序另引入数组 order[]，改上述交换为 order[i]和 order[j]的交换。请在【 】处填入适当的内容。

```
#include<stdio.h>
#define Nember 30
#define Terms 10
#define INF "t.in"
#define OUTF "t.out"
```

```
it score[Number][Terms];
int total[Number], order[Number];
main()
{inti,j,n,ns,,t;
FILE *fpt;
if((ftp=fopen(INF,"r"))==MULL){
printf("Can't pen file %s\n",INF);
exit(1);}
fsanf(ftp,"%d","%d",\&n,&ns);
for(i=0;i<n;i++){
for(j=0;j<ns;j++);
fsanf(fpt,"%d",sore[i]+j);
for(t=j=0;j<=ns;j++)
t+=score[i][j];
total[i]=t;
【 】;}
fclose(fpt);
for(I=0;i<=n-1;i++)
for(j=【 】;j<n;j++)
if(【 】){
t=order[i];
order[i]=order[j];
order[i]=t;}
fpt=fopen(OUTF,"w");
for(i=0;i<n;i++);
fprintf(fpt,"%4d %7d:",I+1,total[order[]]);
for(j=0;i<n;I+=)
fprintf(fpt,"%3d",score[order[i][j]];
fprintf(fpt"\n");}
fclose(fpt);
}
```

621. 假设以下程序执行前文件 gg.txt 的内容为:sample.程序运行后的结果是【 】。

```
#include<stdio.h>
void main(void)
{FILE *fp;
    long position;
    fp=fopen("gg.txt","a");
    position=ftell(fp);
    printf("position=%ld\n",position);
    printf(fp,"sample data\n");
```

```
    position=ftell(fp);
    printf("position=%ld\n",position);
    fclose(fp);
}
```

622. 以下程序的功能是将文件 file1.c 的内容输出到屏幕上并复制到文件 file2.c 中。请将【 】处添入适当的内容。

```
#include<stdio.h>
main()
{
    FILE【 】;
    fp1=fopen("file1.c","r");
    fp2=fopen("file2.c","w");
    while(! Feof(fp1)) putchar(grtchar(fp1));
    【 】
    while(! Feof(fp1)) putc(【 】);
    fclose(fp1);
    fclose(fp2);
}
```

623. 以下程序的功能是用"追加"的形式打开 gg.txt 查看文件指针的位置；然后向文件中写入"data"再查看文件指针的位置。其中 ftell(*FILE)返回 long 型的文件指针位置。

程序执行前 gg.txt 内容为：sample。请在【 】处填入适当的内容。

```
#include<stdio.h>
void main(void)
{【 】
    long position;
    fp=fopen(【 】);
    position=ftell(fp);
    printf("position=%ld\n",position);
    fprintf(【 】);
    position=ftell(fp);
    printf("position=%ld\n",position);
    fclose(fp);
    fclose(fp);
}
```

624. 当调用函数 read 从磁盘文件中读数据时，若函数的返回值为 10，则表明读入了 10 个字符；若函数的返回值为 0，则是【 】；若函数的返回值为-1，则意味着【 】。

625. What's wrong with this program?

```
int main(void)
{
int * fp;
int k;
fp = fopen("gelatin");
for (k = 0; k < 30; k++)
    fputs(fp, "Nanette eats gelatin.");
fclose("gelatin");
return 0;
}
```

626. 请编写程序：从键盘输入一个字符串，将其中的小写字母全部转换成大写字母，输出到磁盘文件"upper.txt"中保存。输入的字符串以："！"结束。然后再将文件 upper.txt 中的内容读出显示在屏幕上。

第18章 结构体与其他数据格式 (1)

627. 已知学生记录描述为

```
struct student
{int no;
char name[20];
char set;
struct
{int year;
int month;
int day;
}birth;
};
struct student s;
```

628. 设变量 s 中的"生日"应是"1984 年 11 月 11 日"，下列对生日的正确赋值方式是【 】.

A. year=1984; month=11; day=11;

B. birth.year=1984; birth.month=11; birth.day=11;

C. s.year=1984; s.month=11; s.day=11;

D. s.birth.year=1984; s.birth.month=11; s.birth.day=11;

629. 当说明一个结构体变量时系统分配给它的内存是【 】.

A. 各成员所需内存量的总和

B. 结构中第一个成员所需内存量

C. 成员中占内存量最大者所需的容量

D. 结构中最后一个成员所需内存量

630. 以下对结构体类型变量的定义中不正确的是【 】.

A.

```
#define STUDENT struct student
STUDENT
{int num;
```

```
float age;
}std1;
```

B.

```
struct student
{int num;
float age;
}std1;
```

C.

```
struct
{int num;
float age;
}std1;
```

D.

```
struct
int num;
float age;
}student;
struct student std1;
```

631. 设有以下说明语句

```
struct stu
{int a;
float b;
}stutype;
```

632. 则下面的叙述不正确的是【 】.

A. struct 是结构体类型的关键字

B. struct stu 是用户定义的结构体类型

C. stutype 是用户定义的结构体类型名

D. a 和 b 都是结构体成员名

633. C 语言结构体类型变量在程序执行期间【 】..

A. 所有成员一直驻留在内存中

B. 只有一个成员驻留在内存中

C. 部分成员驻留在内存中

D. 没有成员驻留在内存中

634. 在 32 位 IBM-PC 机上使用 C 语言，若有如下定义：

```
struct data
{int i;
char ch;
double f;
}b;
```

635. 则结构变量 b 占用内存的字节数是【 】.

A. 4

B. 8

C. 16

D. 32

636. 以下程序的运行结果是【 】.

```
#include "stdio.h"
main()
{struct data
{int year,month,day;
}today;
printf("%d\n",sizeof(struct data) );
}
```

A. 6

B. 8

C. 10

D. 12

637. 根据下面的定义，能打印出字母 M 的语句是【 】.

```
struct person{char name[9];
int age;
};
struct person class[10]={"John",17,
"Paul",19,
"Mary",18,
```

```
"adam",16
};
```

A. printf("%c\n",class[3].name);

B. printf("%c\n",class[3].name[1]);

C. printf("%c\n",class[2].name[1]);

D. printf("%c\n",class[2].name[0]);

638. 下面程序的运行结果是【 】.

```
main()
{
struct cmplx {int x;
int y;
} cnum[2]={1,3,2,7};
printf("%d\n",cnum[0].y/cnum[0].x*cnum[1].x);
}
```

A. 0

B. 1

C. 3

D. 6

639. 若有以下定义和语句:

```
struct student
{int age;
int num ;
};
struct student stu [3]={{1001,20 },{1002,19},{1003,21}};
main()
{struct student *p;
p=stu;
...
}
```

640. 则以下不正确的引用是【 】。

A. (p++)->num

B. p++

C. (*p).num

D. p=&stu.age

641. 以下 scanf 函数调用语句中对结构体 变量成员的不正确引用是【 】。

```
stuct pupil
{char name[20 ];
int age ;
int sex;
}pup[5],*p;
p=pup;
```

A. scanf("%s",pup[0].name);

B. scanf("%d",&pup[0].age);

C. scanf("%d",&sex);

D. scanf("%d",p->age);

642. 有以下定义和语句，则以下引用形式不合法的是【 】。

```
struct s
{int i1;
struct s *i2,i0;
};
static struct s
a[3 ]={2,&a[1],'\0',4,%a[2],&a[0],6,'\0',&a[1]},
*ptr;
ptr=a;
```

A. ptr->i1++

B. *ptr->i2

C. ++ptr->i0

D. ptr->i1

643. 设有如下定义：

```
struct sk
{int n;
float x;
} data,*p;
```

644. 若要使 P 指向 data 中的 n 域， 正确的负值语句的是【 】。

A. p=&data.n;

B. *p=data.n;

C. p=(struct sk*)&data.n;

D. p=(struct sk*)data.n;

645. 若有以下说明和语句：

```
struct student
{int age;
int num;
}std,*p;
p=&std;
```

646. 则以下对结构体变量 std 中成员 age 的引用方式不正确的是【 】。

A. std.age

B. p->age

C. (*p).age

D. *p.age

647. 若以下程序段：

```
struct dent
{ int n;
int *m;
};
int a=1, b=2,c=3;
struct dent s[3]={{101<&a},{102<&b},{103,&c};
main()
{
struct dent *p;
p=s;
...
}
```

648. 则以下表达中值为 2 的是--D---。

A. (p++)->m

B. *(p++)->m

C. (*p).m

D. *(++p)->m

649. 若有以下说明和语句，则对中域的正确引用方式是【 】。

```
struct pupil
{char name [20];
int sex;
}pup,*p;
p=&pup;
```

A. p.pup.sex

B. p->pup.sex

C. (*p).pup.sex

D. (*p).sex

650. 设有以下语句：

```
struct st
{int n;
struct st *next;
};
static struct st a[3] ={5,&a[1],7,&a[2],9,'\0'},*p;
p=&a=[0];
```

651. 则以下表达式的值为 6 的是【 】。

A. p++->n

B. p->n++

C. (*p).n++

D. ++p->n

652. 以下程序的输出结果是【 】。

```
struct stu
{int x;
int *y;
} *p;
int dt[4] ={10,20,30,40};
```

```
struct stu a[4]={50,&dt[0],60,&dt[1],70,&dt[2],80,&dt[3]
};
main()
{p=a;
printf("%d,",++p->x);
printf("%d,",(++p)->x);
printf("%d\n",++(*p->y));
}
```

A. 10,20,20

B. 50,60,21

C. 51,60,21

D. 60,70,31

653. 若有以下说明和语句，则下面表达式中值为 1002 的是【 】。

```
struct student
{int age;
int num;
};
struct student stu[3]={{1001,20},{1002,19},{1003,21};
struct student *p;
p=stu;
```

A. (p++)—>num

B. (p++)—>age

C. (*p).num

D. (*++p).age

654. 以下对结构体变量 stul 中成员 age 的非法引用是【 】。

```
struct student
{int age;
int num;
}stul,*p;
p=&stul;
```

A. stul.age

B. student.age

C. p—>age

D. (*p).age

655. 以下程序的功能是：读入一行字符（如：a,….y,z）,按输入时的逆序建立一个链接式的结点序列，即先输入的位于链表尾（如下图），然后再按输入的相反顺序输出，并释放全部结点。请选择正确的内容填入【 】中。

```
#include<stdio.h>
#define getnode(type) 【    】malloc(sizeof(type))
main()
{struct node
{char info;
struct node*link;
}*top,*p;
char c;
top=NULL;
while((c=getchar())【    】)
{p=getnode(struct node);
p—>info=c;
p—>link=top;
top=p;
}
while(top)
{【    】;
top=top—>link;
putchar(p—>info);
free(p);
}
}
```

656. 其中第 1 个空格应该填写【 】。

A. (type)

B. (type*)

C. type

D. type*

657. 其中第 2 个空格应该填写【 】。

A. =='\0'

B. !='\0'

C. =='\n'

D. ='\n'

658. 其中第 3 个空格应该填写【 】。

A. top=p

B. p=top

C. ==top

D. ==p

659. 若要利用下面的程序片段使指针变量 p 指向一个存储整型变量的存储单元，则应填入的内容是【 】。

```
int*p;
P=【    】malloc(sizeof(int));
```

A. int

B. int*

C. (*int)

D. (int*)

660. 当说明一个共用体变量时系统分配给它的内存是【 】。

A. 各成员所需内存量的总和

B. 结构中第一个成员所需内存量

C. 成员中占内存量最大者所需的容量

D. 结构中最后一个成员所需内存量

661. 以下对 C 语言中共用体类型数据的叙述正确的是【 】。

A. 可以对共用体变量名直接赋值

B. 一个共用体变量中可以同时存放其所有成员

C. 一个共用体变量中不可能同时存放其所有成员

D. 共用体类型定义中不能出现结构体类型的成员

662. 若有以下定义和语句：

```
union data
{int i;
char c;
float f;
}a;
int n;
```

663. 则以下语句正确的是【 】。

A. a=5;

B. a={2,'a',1.2};

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

D. n=a;

664. 设有以下说明，则下面不正确的叙述是【 】。

```
union data
{ int i;
char c;
float f;
}un;
```

A. un 所占的内存长度等于成员 f 的长度

B. un 的地址和它的各成员地址都是同一地址

C. un 可以作为函数参数

D. 不能对 un 赋值，但可以在定义 un 时对它初始化

665. C 语言共用体型变量在程序运行期间【 】。

A. 所有成员一直驻留在内存中

B. 只有一个成员驻留在内存中

C. 部分成员驻留在内存中

D. 没有成员驻留在内存中

666. 以下程序的运行结果是【 】。

```
#include "stdio.h"
main()
```

```
{ union {long a;
int b;
char c;
}m;
printf("%d\n",sizeof(m));
}
```

A. 2

B. 4

C. 6

D. 8

667. 下面程序的运行结果：

```
#include "stdio.h"
main()
{union {int a[2];
long b;
char c[4];
}s;
s.a[0]=0x39;
s.a[1]=0x38;
printf("%lx\n",s.b) ;
printf("%c\n",s.c[0]);
}
```

668. 第 1 行输出是【 】。

A. 390038

B. 380039

C. 3938

D. 3839

669. 第 2 行输出是【 】。

A. 39

B. 9

C. 38

D. 8

670. 下面程序的运行结果:

```
#include "stdio.h"
main()
{int j;
union {int a;
long b;
unsigned char c;
}m
m.b=0x12345678;
printf("%x\n",m.a;)
printf("%x\n",m.c;)
}
```

671. 第 1 行输出是【 】。

A. 1234

B. 5678

C. 12345678

D. 0

672. 第 2 行输出是【 】。

A. 12

B. 78

C. 1234

D. 5678

673. 在 16 位 IBM-PC 机上使用 C 语言,若有如下定义:

```
union data
{int i;
char ch;
double f;
}b;
```

674. 则共用体变量占用内存的字节数是【 】。

A. 1

B. 2

C. 8

D. 11

675. 以下程序的运行结果是【 】。

```
#include "stdio.h"
union pw
{int i;
char ch[2];
}a;
main()
{
a.ch[0]=13;
a.ch[1]=0;
printf("%d\n",a.i);
}
```

A. 13

B. 14

C. 208

D. 209

676. 使用 typedef 定义一个新类型的正确步骤是【 】.

677. 把变量名换成新类型名.

678. 按定义变量的方法写出定义体.

679. 用新类型名定义变量.

680. 在最前面加上关键字.

A. 2,4,1,3

B. 1,3,2,4

C. 2,1,4,3

D. 4,2,3,1

681. 下面对 typedef 的叙述中不正确的是【 】。

A. 用 typedef 可以定义各种类型名,但不能用来定义变量

B. 用 typedef 可以增加新类型

C. 用 typedef 只是将已存在的类型用一个新的标识符来代表

D. 用 typedef 有利于程序的通用和移植

682. 以下程序的运行结果是【 】。

```
typedef union {long a[2];
int b[4];
char c[8];
}TY
TY our;
main()
{printf ("%d\n",sizeof(our));
}
```

A. 32

B. 16

C. 8

D. 24

683. 以下程序的运行结果是【 】. .

```
struct n{
int x;
char c;
};
main()
{struct n a={10,'x'};
func(a);
printf("%d,%c",a.x,a.c);
}
func(struct nb)
{b.x=20;
b.c='y';
}
```

684. 以下程序的运行结果是【 】。

```
main()
{struct EXAMPLE{struct{
int x;
inty;
}in;
int a;
int b;
}e;
e.a=1;e.b=2;
e.in.x=e.a*e.b;
e.in.y=e.a+e.b;
printf("%d,%d",e.in.x,e.in.y);
}
```

685. 以下程序用以输出结构体变量 bt 所占内存单元的字节数,请在( )内填上适当内容.

```
struct ps
{
double i;
char arr[20];
};
main()
{ struct ps bt;
printf("bt size : %d\n",(sizeof(struct ps));
}
```

686. 以下程序用来按学生姓名查询其排名和平均成绩.查询可连续进行,直到键入 0 时结束,请在【 】内填入正确内容.

```
#include <stdio.h>
#include <string.h>
#define NUM 4
struct student
{ int rank;
char * name;
float score;
};
【   】stu[]={ 3,"Tom",89.3,
4,"Mary",78.2 ,
1,"Jack",95.1,
2,"Jim",90.6,
};
```

```
main()
{ char str[10];
int i;
do { printf("Enter a name:");
scanf("%s",str);
for(i=0;i<NUM;i++)
if( 【    】 )
{ printf("name : %8s\n,"stu[i].name);
printf("rank :%3s\n",stu[i].rank);
printf("average :0.5s\n",stu[i].score);
【    】;
}
if(i>=NUM) printf("Not found\n");
} while(strcmp(str,"0")! = 0);
}
```

687. 设有三人的姓名和年龄存在结构数组中,以下程序输入三人中年龄居中者的姓名和年龄,请在【 】内填入正确内容.

```
static struct man
{
char name[20];
int age;
} person[]={"li-ming",18,
"wang-hua",19,
"zhang-ping",20
};
main()
{ int i,j,max,min;
max=min=person[0].age;
for(i=1;i<3;i++)
if(person[i].age<max) ( 【    】 );
else if(person[i].age<min) ( 【      】 );
for(i=0;i<3;i++)
if(person[i].age! = max 【    】 person[i].age! = min)
{ printf("%s %d\n",person[i].name,person[i].age);
break;
}
}
```

688. 以下程序用"比较计数"法对结构数组 a 按字段 num 进行降序排列."比较计数"法的基本思想是:通过另一字段 con 记录 a 中小于某一特定关键字的元

素的个数.待算法结束,a[i].con 就是 a[i].num 在 a 中的排序位置.请在( )内填入正确内容.

```
#define N 8
struct c
{
int num;
int con;
} a[16];
main()
{int i,j;
for(i=0;i<N;i++)
{scanf("%d",&a[i].num);
a[i].con=0;
}
for(i=N-1;i>=1;i--)
for(j=i-1;j>=0;j--)
if(a[i].num<a[j].num)
【 】;
else 【 】;
for(i=0;i<N;i++)
printf("%d,%d\n"a[i].num,a[i]con);
}
```

689. 若已定义:

```
struct num
{int a;
int b;
float f;
} n={1,3,5.0};
struct num *pn=&n;
```

690. 则表达式 pn->b/n.a*++pn->b 的值是【 】，表达式 (*pn).a+pn->f 的值是【 】

691. 以下程序的功能是计算并打印复数的差。请在【 】内填入正确内容。

```
struct comp
{float re;
float im;
};
struct comp *m(x,y)
struct comp *x,*y;
```

```
{【 】;
z=(struct comp * )malloc(sizeof(struct comp));
z->re=x->re-y->re;
z->im=x->im-y->im;
return(【 】);
}
main()
{struct comp *t;
struct comp a,b;
a.re=1; a.im=2;
b.re=3; b.im=4;
t=m(【 】);
printf("z.re=%f , z.im=%f",t->re,t->im);
}
```

692. 以下程序调用 readrec 函数把 10 名学生的学号、姓名、四项成绩以及平均分放在一个结构体数组里,学生的学号、姓名、和四项成绩由键盘输入,然后计算出平均分放在结构体对应的域中,调用 writerec 函数输出 10 名学生的记录.请在【 】内填入正确内容.

```
#include <stdio.h>
struct stud
{ char num[5] , name[10];
int s[4];
int ave;
};
main()
{struct stud st[30];
int i , k;
for(k=0;k<10;k++) readrec(&st[k]);
writerec(st);
}
readrec( struct stud * rec)
int i , sum; char ch;
gets(rec->num);
gets(rec->name);
for(i=0;i<4;i++) scanf("%d", 【 】);
ch=getchar();
sum=0;
for(i=0;i<4;i++ ) sum +=【 】;
rec->ave=sum/4.0;
}
```

```
writerec(struct stud * s)
{ int k,j;
for(k=0;k<10;k++)
{ printf("NUM:%s MANE:%s\n",(*(s+k)).num,(*(s+k)).name );
for(i=0;i<4;i++)
printf(" MARK:%5d", 【 】);
printf("AVE:%5d\n",(*(s+k)).ave);
}
}
```

693. 以下程序的运行结果是【 】。

```
struct ks
{int a;
int *b;
} s[4],*p;
main()
{
int n=1 , i;
printf("\n");
for(i=0;i<4;i++)
{ s[i].a=n;
s[i].b=&s[i].a;
n=n+2;
}
p=&s[0];
p++;
printf("%d,%d\n",(++p)->a,(p++)->a);
}
```

694. 结构数组中存有三人的姓名和年龄,以下程序输出三人中最年长者的姓名和年龄。请在【 】内填入正确内容。

```
static struct man
{
char name[20];
int age;
} person[ ]={"li-ming",18,
"wang-hua",19,
"zhang-ping",20
};
main()
{ struct man *p,*q;
int old=0;
```

```
p=person;
for(;p【 】; p++)
if (old<p->age)
{q=p;【 】;}
printf("%s%d",【 】);
}
```

695. 以下程序运行结果为【 】。

```
struct s{
int a;
float b;
char *c;
}
main()
{
static struct s x={19,83.5,"zhang"};
struct s *px=&x;
printf("%d%.1f%s\n",x.a,x.b,x.c);
printf("%d%.1f%s\n",px->a, (*px).b, px->c);
printf("%c%s\n",*px->c-1,&px->c[1]);
}
```

696. 以下程序是用来统计学生成绩。其功能包括输入学生姓名和成绩，按成绩从高到低排列打印输出，对前 70%的学生定为合格（pass），而后 30%的学生定为不合格（fail）。请在【 】内填入正确内容.

```
#include<stdui.h>
#include<string.h>
typedef struct
{ char name[30];
int grade;
}student;
student class[40];
void sortclass();
void swap();
main()
{
int ns,cutoff,I;
printf("number of student: \n");
scanf("%d",&ns);
printf("Enter name and grade for each student: \n");
for(i=0;i<ns;i++)
```

```
scanf("%s%d",【 】);
sortclass(class,ns);
cutoff=(ms*7)/10-1;
printf("\n");
for(i=0;i<ns;i++)
{printf("%-6s%3d",class[i],name,class[i],grade);
if(i<=cutoff) printf(" pass \n");
else printf(" fail \n");
}
}
void sortclass(student st[],int nst)
{
int i,j,pick;
for(i=0:i<(nst-1);i++)
{ pick=i;
for(【 】;j++)
if(st[j].grade>st[pick].grade)
pick=j;
swap(【 】);
}
}
void swap(ps1,ps2)
student *ps1,ps2;
{ student temp;
strcpy(temp.name,ps1->name);
temp.grade=ps1->grade;
strpcy(ps1->name,ps2->name);
ps1->grade=ps2->grade;
strcpy(ps->name,temp.name);
ps2->grade=temp.grade;}
```

697. 以下程序输入若干人员的姓名(六位字母)机器电话号码(七位数字),以字符#结束输入。然后输入姓名,查找该人的电话号码。数据从 s[1]开始存放。请在【 】内填入正确内容。

698. 设有以下定义和语句，请在 printf 语句的【 】中填上能够正确输出的变量及相应的格式说明。

```
union
{ int n;
double x;
} num;
```

```
num.n=10;
num.x=10.5;
printf("【 】",【 】);
```

699. 以下程序的运行结果是【 】。

```
main()
{ struct EXAMPLE
{ union {
int x;
int y;
} in;
int a;
int b;
} e;
e.a=1; e.b=2;
e.in.x=e.a*e.b;
e.in.y=e.a+e.b;
printf("%d,%d",e.in.x,e.in.y);
}
```

700. 以下程序用以读入两个学生的情况存入结构数组。每个学生的情况包括：姓名，学号，性别。若是男同学，则还登记视力正常与否（正常用 Y，不正常用 N）；对女生则还登记身高和体重。请在【 】内填入正确内容。

```
struct
{ char name[10];
int number;
char sex;
union body
{ char eye;
struct
{ int hength;
int weight;
}f;
}body;
}per[2];
main()
{ int i;
for (i=0;i<2;i++)
{ scanf("%s %d %c",&per[i].name,&per[i].number,&per[i].sex);
if(per[i].sex=='m')
scanf("%c",【 】);
```

```
else if(per[i].sex=='f')
scanf("%d %d",【 】,【 】);
else printf("input error\n");
}
```

701. 以下程序的运行结果是【 】。

```
union ks
{int a;
int b;
};
union ks s[4];
union ks *p;
main()
{
int n=1,i;
printf("\n");
for(i=0;i<4;i++)
{ s[i].a=n;
s[i].b=s[i].a+1;
n=n+2;
}
p=&s[0];
printf("%d,",p->a);
printf("%d,",++p->a);
}
```

702. 以下程序的运行结果是【 】。

```
main()
{union EXAMPLE
{sturuct
{int x;
int y;
}in;
int a;
int b;
}e;
e.a=1;
e.b=2;
e.in.x=e.a*e.b;
e.in.y=e.a+e.b;
printf("%d %d",e.in.x,e.in.y);
}
```

703. 以下程序的运行结果为【 】。

```
#include "stdio.h"
struct w
{char low;
char high;
};
union u
{sturct w bute;
int word;
}uu;
main()
{uu.word=0x1234;
printf("Word value: %04x\n",uu.word);
printf("High value: %02x\n",uu.byte.high);
printf("Low value: %02x\n",uu.byte.low);
uu.byte.low=0xff;
printf("Word value: %04x\n",uu.word);
}
```

704. 写出下列程序的输出结果【 】。

```
enum coin {
penny,nickel,dime,quarter,half_dollar,dollar};
char
*name[]={"penny","nickel","dime","quarter","hal_fdollar","do
llar"};
main()
{
enum coin money1,money2;
money1=dime;
money2=dollar;
printf("%d %d\n",money1,money2);
printf("%s %s\n",name[(int)money1],name[(int)money2]);
}
```

705. 以下程序对输入的两个数字进行正确性判断,若数据满足要求则打印正确信息,并计算结果,否则打印出相应的错误信息并继续读数,直到输入正确为止. 请在【 】内填入正确内容.

```
enum ErrorData {Right,Less0,Great100,MinMaxErr};
char *ErrorMessage[]={
"Enter Data Right",
"Data<0 Error",
```

```
"Data>100 Error",
"x>y Error"
};
main()
{int status,x,y;
do {printf("please enter two number(x,y)");
scanf("%d%d",&x,&y);
status=【 】;
printf(ErrorMessage[【 】]);
}while(status!=Right);
printf("Result=%d",x*x+y*y);
}
int error (int min,int max)
{if(max<min) return MinMaxErr;
else if (max>100) ruturn Great100;
else if(min<0) return Less0;
else【 】;
}
```

706. 以下程序的输出结果是【 】。

```
typedef int INT;
main()
{INT a,b;
a=5;
b=6;
printf("a=%d\tb=%d\n",a,b);
{
float INT;
INT=3.0;
printf("2*INT=%.2f\n",2*INT);
}
}
```

第19章 结构体与其他数据格式 (2)

707. 若有以下定义和说明：

```
#include"stdio.h"
struct std
{ char num[6];
char name[8];
float mark[4];
}a[30];
FILE *fp;
```

708. 设文件中以二进制形式存有 10 个班的学生数据，且已正确打开，文件指针定位于文件开头。若要从文件中读出 30 个学生的数据放入 a 数组中，以下不能此功能的语句是【 】。

A.

```
for(i=0;i\<30;i++)
    fread( &a[i], sizeof( struct std ), 1L, fp );
```

B.

```
for(i=0; i\<30; i++, i++)
    fread( a+i, sizeof( struct std ), 1L，fp );
```

C. `fread( a, sizeof( struct std ), 30L，fp );`

D.

```
for( i=0; i\<30; i++ ) fread( a[i], sizeof( struct std ),
1L，fp );
```

709. 设有以下结构体类型：

```
struct st
{ char name[8];
int num;
float s[4];
} student[50];
```

并且结构体数组 student 中的元素都已有值，若要将这些元素写到硬盘文件 fp 中，以下不正确的形式是【 】。

A. `fwrite( student, sizeof( struct st ), 50, fp );`

B. fwrite(student, 50*sizeof(struct st), 1, fp);

C. fwrite(student, 25*sizeof(struct st), 25, fp);

D. for(i=0;i<50; i++) fwrite(student+i, sizeof(struct st), 1, fp);

710. 下面程序以字符形式读入一个文件,从文件中检索出六种 C 语言的关键字,并统计,输出每种关键字在文件中出现的次数.为简化运算,程序中规定:单词是以空格或'\t','\n'结束的字符串.请从下面对应的一组选项中选出正确的内容填入.

```
#include <stdio.h>
#include <string.h>
FILE *cp;
char fname[20],buf[100];
int num;
struct key
{char word[10];
int count;
} keyword[]={"if",0,"char",0,"int",
0,"else",0,"while",0,"return",0};
char *getword(FILE *fp)
{int i=0;
char c;
while ((c=getc(fp))!=EOF && (c==' '||c=='t'||c=='\n'));
if (c==EOF)return(0);
else buf[i++]=c;
while (【  】 &&c!=' '&&c!='t'&&c!='\n')
buf[i++]=c;
buf[i]='\0';
}
lookup(char *p)
{int i;
char *q,*s;
for (i=0;i<num;i++)
{q=&keyword[i].word[0];
s=p;
while(*s&&(*s==*q))
{s++;q++; }
if(*s==*q)
{keyword[i],count++;
```

```
break;
}
return;
}
main()
{int i;
char *word;
printf("Enter file name:");
scanf("%s",fname);
if((cp=fopen (fname,"r"))==NULL)
{printf("file open error!");
exit(0);
}
num=sizeof(keyword)/sizeof(struct key);
while (【】)
lookup(word);
for(i=0;i\<num;i++)
printf("keyword=%s,count=%d\n",keyword[i],word,keyword[i].co
unt);
}
```

上述第 1 个空格填写【 】。

A. (c=fgetc(fp))!=EOF

B. (c=fgetc(fp))==EOF

C. (c=fgetc(fp))

D. (c=fgetc(fp))<>EOF

711. 上述第 2 个空格填写【 】。

A. (word==getword(cp))!=NULL

B. word==getword(cp)!=NULL

C. (word==getword(cp))!=NULL

D. (word=getword(cp))!=NULL

712. 设有以下说明语句

```
typedef struct {
    int n;
    char ch[8];
```

```
} PER;
```

713. 则下面叙述中正确的是

A. PER 是结构体变量名

B. PER 是结构体类型名

C. typedef struct 是结构体类型

D. struct 是结构体类型名

714. 有以下程序

```
#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];
}
int 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]);
}
```

715. 程序的运行结果是

```
A. `Qian,f,95,92` B. `Qian,m,85,90` C. `Zhao,f,95,92` D.
`Zhao,m,85,90`
```

716. 有下列程序段

```
struct st
{
    int x; int *y;
} *pt;
```

```
int a[] = { 1, 2 }, b[] = { 3, 4 };
struct st c[2] = { 10, a, 20, b };
pt = c;
```

717. 下列选项中表达式的值为 11 的是（ ）。

A. *pt->y

B. pt->x

C. ++pt->x

D. (pt++)->x

718. 以下程序的功能是输入 4 名学生的姓名，年龄，住址并存入磁盘文件。请在【 】处填人适当内容。

```
#include<stdio.h>
#define SIZE 4
struct student_type
{
    char name[10];
    int num;
    int age;
    char addr[15];
}
stud[SIZE];
void save()
{
    FILE*fp;
    int i;
    if((fp=fopen("stu_list","wb"))= =NULL)
    {
        printf("cannot open file\n");
        return;
    }
    for{i=0;i<SIZE;i++}
    if(fwrite(【     】)!=1)
        printf("file while error\n");
}
main()
{ int i;
    for(i=0,i<SIZE;i++)
```

```
    scanf("%s,%d,%d,%s",.【    】,【    】,
【    】,stud[i].addr);
    save();
}
```

719. 阅读以下程序：

```
#include<stdio.h>
#define SIZE 4
struct student_type
{
    char name[10];
    int num;
    int age;
    char addr[15];
}stud[SIZE];
void save()
{ FILE*fp;
    int i;
    if((fp=fopen("stu_list","wb"))= =NULL)
    {
    printf("cannot open file\n");
    return;
    }
    for(i=0,i<SIZE;i++)
        if(fwrite(&stud[i],sizeof(struct
student_type),1,fp)!=1)
    printf("file white error\n");
}
void load()
{
    FILE*fp;
    int i;
    if ((fp=fopen("stu_dat","rd"))==NULL)
    {
    printf("cannot open file\n");
    return;
    }
    {
    printf("cannot open file\n");
    return;
    }
    for(i=0,i<SIZE;i++)
```

```
        if(fread(&stud[i],sizeof(struct
student_type),1,fp)!=1)
        {
            if(feof(fp))return;
            printf("file read error\n);
        }
}
main()
{
    load();
    save();
}
```

此程序功能为：将磁盘文件【 】中【 】数据，并输出到【 】文件中。

720. 以下程序的功能只将文件 stud_dat 中第 i 个学生的姓名，学号年龄，性别输出。请在【 】处填入适当的内容。

```
#include&lt;stdio.h>
struct _student_type
{
char name [10];
int num;
int age;
char sex;
} stud[10];
main()
{
    int i;
    FILE 【  】;
    if((fp=fopen("stud_dat","rb"))==NULL)
    {
        printf("cannot open file\n");
        exit(0) ;
    }
    scanf("%d",\&i);
    fseek(【        】);
    fread(【    】,sizeof(struct_student_type),1fp);

printf("%s%d%d%c\n",stud[i].name,stud[i].num,stud[i]age,stud
[i].sex);
    fclose(fp);
}
```

721. 设有以下结构体类型：

```
struct st
{char name[8];
int num;
float s[4];
}student[50];
```

并且结构体数组 student 中的元素都已有值，若要将这些元素写到硬盘文件 fp 中，请将以下 fwrite 语句补充完整。

```
fwrite(student,【 】,1,fp);
```

722. 下面程序从一个二进制文件中读入结构体数据，并把结构体数据显示在终端屏幕上。请在[]处填入适当内容。

```
#include>stdio.h>
struct rec
{
    int num;
    float total;
}
main()
{FILE *f;
    f=fopen("bin.dat","rb");
    reout(f);
    fclose(f);
}
reout(【】)
{struct rec r ;
    while(! feof(f))
    {fread(&r,【    】,1,f);
    printf("%d,%f\n",[r.num,r.total]);
    }
}
```

723. Consider the following declarations:

```
struct fullname {
    char fname[20];
    char lname[20];
};
struct bard {
    struct fullname name;
    int born;
```

```
    int died;
};
struct bard willie;
struct bard *pt = &willie;
```

a. Identify（标记） the born member（成员） of the willie structure（结构体） using the willie identifier（标识符）.

b. Identify the born member of the willie structure using the pt identifier.

c. Use a scanf() call（调用） to read in a value for the born member using the willie identifier.

d. Use a scanf() call to read in a value for the born member using the pt identifier.

e. Use a scanf() call to read in a value for the lname member of the name member using the willie identifier.

f. Use a scanf() call to read in a value for the lname member of the name member using the pt identifier.

g. Construct（构造） an identifier for the third letter of the first name of someone described（描写） by the willie variable（变量）.

h. Construct an expression（表达式） representing（表示） the total number of letters in the first and last names of someone described by the willie variable.

i. 对 willie 成员进行完整赋值，使得 willie 的 name 成员的 fname 值为"Abraham"，lname 值为"Lincoln"，born 值为 1809，died 值为 1865。

724. Given the following typedef, declare a 10-element array of the indicated structure. Then, using individual member assignment (or the string equivalent), let the third element describe a Remarkatar lens with a focal length of 500 mm and an aperture of f/2.0.

```
typedef struct lens { /* lens descriptor */
    float foclen; /* focal length,mm */
    float fstop; /* aperture */
    char brand[30]; /* brand name */
```

```
} LENS;
```

725. Repeat it, but use an initialization list with a designated initializer in the declaration rather than using separate assignment statements for each member.

726. Declare an enumeration（枚举） with the tag（标记） `choices` that sets the enumeration constants `no`, `yes`, and `maybe` to 0, 1, and 2, respectively（分别）.

727. Declare a pointer to a function that returns a pointer-to-`char` and that takes a pointer-to-`char` and a `char` as arguments.

728. 设文件 STUDENT.DAT 中存放着一年级学生的基本情况，这些情况由以下结构体来描述：

```
struct student
{long int num; /*学号*/
char [10]; /*姓名*/
int age; /*年龄*/
char sex; /*性别*/
char speciality[20]; /*专业*/
char addr[40]; /*住址*/
};
```

729. 请编写程序，输出学号在 970101~970135 之间的学生学号，姓名，年龄和性别。

730. 默写冒泡排序法的函数。（学有余力的同学可以在冒泡排序法内自由发挥）

```
void bubble_sort_1(double arr[], const unsigned int len)
{
    unsigned int size, i;
    for (size = len; size > 1; size--) // 排序的范围从全数组慢慢收缩至停止
    {
        for (i = 1; i < size; i++) // 单趟排序
        {
            if (arr[i - 1] > arr[i]) // 需要发生交换的条件
            {
                double temp; // temp 的唯一正确用法
                temp = arr[i - 1];
                arr[i - 1] = arr[i];
                arr[i] = temp;
```

```
            }
        }
    }
}
```

或者

```
void bubble_sort_2(double arr[], const unsigned int len)
{
    unsigned int size, i; // size 是排序的范围，i 是索引（index）的缩写
    for (size = 0; size < len; size++) // 排序的范围从全数组慢慢收缩至停止，虽然此处是递增的，但是在下面使用时有减号
    {
        for (i = 1; i < len - size; i++) // 单趟排序
        {
            if (arr[i - 1] > arr[i]) // 需要发生交换的条件
            {
                double temp; // temp 的唯一正确用法
                temp = arr[i - 1];
                arr[i - 1] = arr[i];
                arr[i] = temp;
            }
        }
    }
}
```

或者

```
void bubble_sort_3(double arr[], const unsigned int len)
{
    unsigned int size, i; // size 是排序的范围，i 是索引（index）的缩写
    for (size = len; size > 1; size--) // 排序的范围从全数组慢慢收缩至停止
    {
        int is_swapped = 0; // 标记是否交换
        for (i = 1; i < size; i++) // 单趟排序
        {
            if (arr[i - 1] > arr[i]) // 需要发生交换的条件
            {
                double temp; // temp 的唯一正确用法
                temp = arr[i - 1];
                arr[i - 1] = arr[i];
                arr[i] = temp;
```

```
                is_swapped = 1; // 标记已经经过交换
            }
        }
        if (!is_swapped) // 如果未经过交换，说明已经排好序，可以
退出，不必再排序
            break;
    }
}
```

第20章 位操作

731. 请读程序片段:

```
int x=20;
printf("%d\n", ~x);
```

732. 上面程序片段的输出结果是【 】。

A. 02

B. –20

C. -21

D. -11

733. 表达式~0x13 的值是【 】。

A. 0xFFEC

B. 0xFF71

C. 0xFF68

D. 0xFF17

734. 在位运算中,操作数每右移一位,其结果相当于【 】。

A. 操作数乘以 2

B. 操作数除以 2

C. 操作数除以 4

D. 操作数乘以 4

735. 在位运算中,操作数每左移一位,其结果相当于【 】。

A. 操作数乘以 2

B. 操作数除以 2

C. 操作数除以 4

D. 操作数乘以 4

736. 设有以下语句:

```
char x=3,y=6,z;
z=x^y<<2;
```

737. 则 z 的二进制值是【 】。

A. 00010100

B. 00011011

C. 00011100

D. 00011000

738. 请读程序:

```
struct bit
{
unsigned a_bit:2;
unsigned b_bit:2;
unsigned c_bit:1;
unsigned d_bit:1;
unsigned e_bit:2;
unsigned word:8;
};
main()
{
struct bit *p;
unsigned int modeword;
printf("Enter the mode word (HEX):");
scanf("%x".&modeword);
p=(struct bit *)&modeword;
printf("\n");
printf("a_bit: %d\n",p ->a_bit);
printf("b_bit: %d\n",p ->b_bit);
printf("c_bit: %d\n",p ->c_bit);
printf("d_bit: %d\n",p ->d_bit);
printf("e_bit: %d\n",p ->e_bit);
}
```

若运行时从键盘输入: 96↲

则以上程序的运行结果是【 】。

A.

```
a_bit: 1
b_bit: 2
c_bit: 0
d_bit: 1
e_bit: 2
```

B.

```
a_bit: 2
b_bit: 1
c_bit: 0
d_bit: 1
e_bit: 2
```

C.

```
a_bit: 2
b_bit: 1
c_bit: 1
d_bit: 0
e_bit: 2
```

D.

```
a_bit: 1
b_bit: 2
c_bit: 2
d_bit: 0
e_bit: 1
```

739. 设有以下说明:

```
struct packed
{ unsigned one:1;
unsigned two:2;
unsigned three:3;
unsigned four:4;
} data;
```

则以下位段数据的引用中不能得到正确数值的是【 】。

A. data.one=4

B. data.two=3

C. data.three=2

D. data.four=1

740. 设位段的空间分配由右到左,则以下程序的运行结果是【 】。

```
struct packed_bit
{unsigned a:2;
unsigned b:3;
unsigned c:4;
int i;
} data;
main()
{data.a=8; data.b=2;
printf("%d\n",data.a+data.b) ;
}
```

A. 语法错

B. 2

C. 5

D. 10

741. 设 int a=04, b; 则执行 b=a>>1;语句后，b 的结果是【 】。

A. 04

B. 4

C. 10

D. 2

742. 以下 for(x=0,y=0;(y=123) && (x<4);x++); ，for 循环的执行次数是【 】。

A. 是无限循环

B. 循环次数不定

C. 执行 4 次

D. 执行 3 次

743. 在 C 语言中,&运算符作为单目运算符时表示的是【 】;作为双目运算符时表示的是【 】运算.

744. 与表达式 a&=b 等价的另一书写形式是【 】。与表达式 x^=y-2 等价的另一书写形式是【 】。请读程序片段:

```
int a=1,b=2;
if(a&b) printf("***\n");
else printf("$$$\n");
```

745. 以上程序片段的输出结果是【 】。

746. 设有 char a,b;若要通过 a&b 运算屏蔽掉 a 中的其它位,只保留第 2 和第 8 位(右起为第 1 位),则 b 的二进制数是【 】。测试 char 型变量 a 第六位是否为 1 的表达式是【 】(设最右位是第一位)设二进制数 x 的值是 11001101,若想通过 x&y 运算使 x 中的低 4 位不变,高 4 位轻零,则 y 的二进制数是【 】。

747. 请读程序片段:

```
int a=-1;
a=a | 0377;
printf("%d,%o\n",a,a);
```

748. 以上程序片段的输出结果是【 】。

749. 设 x 是一个整数(16bit),若要通过 x|y 使 x 低 8 位置 1,高 8 位不变,则 y 的八进制数是【 】。

750. 设 x=10100011,若要通过 x\^y 使 x 的高 4 位取反,低 4 位不变,则 y 的二进制数是【 】。

751. 请读程序片段:

```
int m=20,n=025;
if (m&n==m) printf("m\n");
else printf("n\n");
```

752. 以上程序片段的输出结果是【 】。

753. 请读程序片段:

```
int x=1;
printf("%d\n",x);
```

754. 上面程序片段的输出结果是【 】。

755. 以下程序的运行结果是【 】。

```
main()
{
unsigned a,b;
a=0x9a;
b=a;
printf("a:%x\nb:%x\n",a,b) ;
}
```

756. 以下程序的运行结果是【 】。

```
main()
{
char a=-8;unsigned char b=248;
printf("%d,%d",a>>2,b>>2);
}
```

757. 以下程序的运行结果是【 】。

```
main()
{
unsigned char a,b;
a=0x1b;
printf("0x%x\n",b=a<<2);
}
```

758. 请读程序片段:

```
unsigned a=16;
printf("%d,%d,%d\n",a>>2,a=a>>2,a);
```

759. 以上程序片段的输出结果是【 】。

760. 若 x=0123，则表达式(5+(int)(x))&(2)的值是【 】

761. 下面程序的运行结果是【 】。

```
main()
{
unsigned char a,b;
a=0x9d;
b=0xa5;
printf("a AND b:%x\n",a&b);
```

```
printf("a OR b:%x\n",a|b);
printf("a NOR b:%x\n",a^b);
}
```

762. 下面程序的运行结果是【 】。

```
main()
{unsigned a=0112,xmy,z;
x=a>>3; printf("x=%o,",x);
y=~(1<<4); printf("y=%o,",y);
z=x&y; printf("z=%o\n",z);
}
```

763. 下面程序的运行结果是【 】。

```
main()
{unsigned a=0361,x,y; int n=5;
x=a<<(16-n);printf("x=%o\n",x);
y=a>>n; printf("y1=%o\n",y);
y|=x; printf("y2=%o\n",y);
}
```

764. 把 int 类型变量 low 中的低字节及变量 high 中的高字节放入变量 s 中的表达式是【 】。

765. 以下程序的运行结果是【 】。

```
main()
{char a=0x95,b,c;
b=(a&0xf)<<4;
c=(a&0xf0)>>4;
a=b|c;
printf("%x\n",a);
}
```

766. 下面程序的功能是实现左右循环移位，当输入位移的位数是一正整数时循环右移，输入一负整数时循环左移。请在【 】出填入正确内容。

```
main()
{unsigned a;
int a;
printf("请输入一个八进制数:");
scanf("%o",&a);
printf("请输入位移的位数:");
scanf("%d",&n);
```

```
if【 】
{moveright(a,n);
printf("循环右移的结果为:%o\n"，moveright(a,n));}
else
{【 】;
moveleft(【 】);
printf("循环左移的结果为:%o\n",moveleft(a,n));
}
}
moveright(unsigned value,int n)
{unsigned z;
z=(value>>n)|(value<<(16-n));
return(z);
}
moveleft(unsigned value,int n)
{unsigned z;
【 】;
return(z);
}
```

767. 以下函数的功能是计算所用计算机中 int 型数据的字长(即二进制位)的位数。int 型数据所分配的长度是不同的，该函数具有可移植性)。请在【 】处填入正确内容。

```
wordlength()
{int i;
unsigned int v=【 】; /*将 int 型单元各二进制位置 1*/
for(i=1;(v=v>>1)>0;i++); /*计算 int 单元中的位数*/
return(【 】);
}
```

768. 请读以下函数:

```
getbits(unsigned x,unsigned p,unsigned n)
{ x=((x<<(p+1-n))&~((unsigned) ~0>>n));
return(x);
}
```

769. 假设机器的符号整数字长为 16 位。若调用此函数时 x=0115032,p=7,n=4，则函数返回值的八进制数是【 】。

770. 下面程序的行结果是【 】。

```
#define PB(X) printf((X)?"1":"0")
```

```
#define UI unsigned int
struct byte
{UI a0:1;UI a1:1;UI a2:1;UI a3:1;UI a4:1;UI a5:1;UI a6;UI
a7;};
union bits{char ch;struct byte bit;};
void decode(union bits);
main()
{char c;union bits asc;
for(c='A';c<'C';c++)
{asc.ch=c;
printf("%o:",c);
decode(asc);
}
}
void decode(union bitsb)
{PB(b.bit.a7);PB(b.bit.a6);PB(b.bit.a5);PB(b.bit.a4);
PB(b.bit.a3);PB(b.bit.a2);PB(b.bit.a1);PB(b.bit.a0);
printf("\n");
}
```

771. 阅读以下程序:

```
struct bit
{
unsigned bund_bit:2;
unsigned char_bit:2;
unsigned pari_bit:1;
unsigned even_bit:1;
unsigned stop_bit:2;
unsigned dummybit:8;
}
main()
{
unsigned int *pmidwird;
struct bit mod;
mod.bund_bit=input("bund bit(0-3)");
mod.char_bit=input("char bit(0-3)");
mod.pari_bit=input("pari bit(0,1)");
mod.even_bit=input("even bit(0,1)");
mod.stop_bit=input("stop bit(0-3)");
mod.dummybit=0;
pmodword=(unsigned int *)&mod;
printf("\n The mode word is %x\n",*pmodword);
```

```
}
input(char *s)
{
int bits;
printf("Enter %s:",s);
scanf("%d",&bits);
return(bits);
}
```

若运行时输入以下数据:

```
Enter bund bit(0-3):2↲
Enter char bit(0-3):2↲
Enter pari bit(0,1):1↲
Enter even bit(0,1):0↲
Enter stop bit(0-3):1↲
```

则运行结果是【 】。

772. 设位段的工件分配由右到左，则以下程序的运行结果是【 】。

```
struct packed_bit
{unsigned a:2;
unsigned b:3;
unsigned c:4;
int i;
}data;
main()
{data.a=1; data.b=2; data.c=3; data,i=0;
printf("%d\n",data);
}
```

773. 请编程序:从终端读入 16 进制无符号整数 m，调用函数 rightrot 将 m 中的原始数据循环右移 n 位。并输出移位前后的内容。

774. 请编写函数 getbits 从一个 16 位的单元中取出以 n1 开始至 n2 结束的某几位，起始位和结束位都从左向右计算。同时编写主函数调用 getbits 进行验证。

第21章 C 预处理器与 C 库

775. 以下叙述中不正确的是【 】。

A. 预处理命令行都必须以#号开始

B. 在程序中凡是以#号开始的语句行都是预处理命令行

C. C 程序在执行过程中对预处理命令行进行处理

D. 以下是正确的宏定义：#define IBM-PC

776. 以下叙述中正确的是【 】。

A. 在程序的一行上可以出项多个有效的预处理命令行

B. 使用带参的宏时,参数的类型应与宏定义时的一致

C. 宏替换不占用运行时间,只占编译时间

D. 在以下定义中 C R 是称为"宏名"的标识符 #define C R 045

777. 设有说明：float x, y, z; 则不正确使用 C 语言库函数的赋值语句是【 】。

A. z=exp(y)+fabs(x);

B. y=log10(y)+pow(y);

C. z=sqrt(y-z);

D. x=(int)(atan2((double)x,y)+exp(y-0.2));

778. 若有代数式 yx+lny，则正确的 C 语言表达式是【 】。

A. sqrt(fabs(pow(y,x)+log(y)))

B. sqrt(abs(pow(y,x)+long(y)))

C. sqrt(fabs(pow(x,y)+log(y)))

D. sqrt(abs(pow(x,y)+long(y)))

779. 请读程序:

```
#define ADD(x) x+x
```

```
main()
{
int m=1,n=2,k=3;
int sum=ADD(m+n)*k;
printf("sum=%d",sum);
}
```

上面程序的运行结果是【 】。

A. sum=9

B. sum=10

C. sum=12

D. sum=18

780. 以下程序的运行结果是【 】。

```
#define MIN(x,y) (x)<(y)?(x):(y)
main()
{int i=10,j=15,k;
    k=10*MIN(i,j);
    printf("%d\n",k);
}
```

A. 10

B. 15

C. 100

D. 150

781. 在宏定义 #define PI 3.14159 中,用宏名 PI 代替一个【 】。

A. 常量

B. 单精度数

C. 双精度数

D. 字符串

782. 以下程序的运行结果是【 】。

```
#include "stdio.h"
#define FUDGE(y) 2.84+y
#define PR(A) printf("%d",(int)(A))
```

```
#define PRINT1(A) PR(A); putchar('\n')
main()
{int x=2;
    PRINT1(FUDGE(5)*x);
}
```

A. 11

B. 12

C. 13

D. 15

783. 以下有关宏替换的叙述不正确的是【 】。

A. 宏替换不占用运行时间

B. 宏名无类型

C. 宏替换只是字符替换

D. 宏名必须用大写字母表示

784. C 语言的编译系统对宏命令的处理是【 】。

A. 在程序运行时进行的

B. 在程序连接时进行的

C. 和 C 程序中的其它语句同时进行编译的

D. 在对源程序中的其它语句同时进行编译的

785. 若有宏定义如下:

```
#define X 5
#define Y X+1
#define Z Y*X/2
```

786. 则执行以下 printf 语句后,输出结果是【 】。

```
int a;a=Y;
printf("%d\n",Z);
printf("%d\n",--a);
```

A. 7↲6

B. 12↲6

C. 12 ↲ 5

D. 7 ↲ 5

787. 若有以下宏定义:

```
#define N 2
#define Y(n) ((N+1)*n)
```

则执行语句 z=2*(N+Y(5)); 后的结果是【 】。

A. 语句有错误

B. z=34

C. z=70

D. z 无定值

788. 若有宏定义: #define MOD(x,y) x%y 则执行以下语句后的输出为【 】。

```
int z,a=15,b=100;
z=MOD(b, a);
printf("%d\n",z++);
```

A. 11

B. 10

C. 6

D. 宏定义不合法

789. 以下程序的运行结果是【 】。

```
#define MAX(A, B) (A>B?A:B)
#define PRINT(Y) printf("Y=%d\t",Y)
void main()
{int a=1,b=2,c=3,d=4,t;
t=MAX(a+b,c+d);
PRINT(t);
}
```

A. Y=3

B. 存在语法错误

C. Y=7

D. Y=0

790. 以下程序段中存在错误的是【 】。

A.

```
#define S 100
int S[S];
```

B.

```
#define PI 3.14159
#define S(r) PI*(r)*(r)
...
int S[10]={0};
S[0]++;
```

C.

```
#define PI 3.14159
#define S(r) PI*(r)*(r)
...
int S=1;
S++;
```

D.

```
#define PI 3.14159
#define S(r) PI*(r)*(r)
...
double S(double r)
{
    return PI*r*r;
}
area=S(3.2);
```

791. 请读程序

```
#include <stdio.h>
#define MUL(x,y) (x)*y
main()
{int a=3,b=4,c;
c=MUL(a++,b++);
printf("%d\n", c);
}
```

792. 上面程序的输出结果是【 】。

A. 12

B. 15

C. 20

D. 16

793. #define 能做简单的替代，用宏替代计算多项式 4xx+3*x+2 之值的函数 F，正确的宏定义是【 】。

A. #define f(x) 4*x*x+3*x+2

B. #define f 4*x*x+3*x+2

C. #define f(a) (4*a*a+3*a+2)

D. #define (4*a*a+3*a+2) f(a)

794. 对下面程序段

```
#define A 3
#define B(a) ((A+1)* a)
...
x=3*(A+B(7));
```

795. 正确的判断是【 】。

A. 程序错误，不许嵌套宏定义

B. x=93

C. x=21

D. 程序错误，宏定义不许有参数

796. 以下程序中，第一个输出值【 】。

```
#include<stdio.h>
#define M 3
#define N (M+1)
#define NN N*N/2
main()
{printf("%d\n",NN);
printf("%d",5*NN);
}
```

A. 3

B. 4

C. 6

D. 8

797. 第二输出值是【 】.

A. 17

B. 18

C. 30

D. 40

798. 以下程序的输出结果为【 】。

```
#include <stdio.h>
#define F(y) 3.84+y
#define PR(a) printf("%d",(int)(a))
#define PRINT(a) PR(a);putchar('n')
main()
{int x=2;
PRINT(F(3)*x));
}
```

A. 8

B. 9

C. 10

D. 11

799. 以下程序的输出结果为【 】。

```
#define PT 5.5
#define S(x) PT*x*x
main()
{inta=1,b=2;
printf("%4.1\n",S(a+b) );
}
```

A. 12.0

B. 9.5

C. 12.5

D. 33.5

800. 以下在任何情况下计算平方数时都不会引起二义性的宏定义是【 】。

A. #define POWER(x)x*x

B. #define POWER(x)(x)*(x)

C. #define POWER(x)(x*x)

D. #define POWER(x)((x)*(x))

801. 在"文件包含"预处理语句的使用形式中，当#include 后面的文件名用 "" 括起时，寻找被包含文件的方式是【 】。

A. 直接按系统设定的标准方式搜索目录

B. 先在源程序所在目录搜索，再按系统设定的标准方式搜索

C. 仅仅搜索源程序所在目录

D. 仅仅搜索在前目录

802. 在"文件包含"预处理语句的使用形式中，当#include 后面的文件名用 <> 括起时，寻找被包含文件的方式是【 】。

A. 仅仅搜索当前目录

B. 仅仅搜索源程序所在目录

C. 直接按系统设定的标准方式搜索目录

D. 先在源程序所在目录搜索，再按系统设定的标准方式搜索

803. 请读程序

```
#define LETTER 0
main()
{char str[20]="C Language",c;
int i;
i=0;
while((c=str[i]!='\0')
{i++;
```

```
#if LETTER
if(c>='a'&&c<='Z')
c=c-32;
#else
if(c>='A'&&c<='Z')
c=c+32;
#endif
printf("%c", c);
}
}
```

804. 上面程序的运行结果是【 】。

A. C Language

B. c language

C. C LANGUAGE

D. c LANGUAGE

805. 以下正确的描述是【 】。

A. C 语言的预处理功能是指定完成宏替换和包含文件的调用

B. 预处理指令只能位于 C 程序文件的首部

C. 凡是 C 程序中行首以 # 标识的控制行都是预处理指令

D. C 语言的编译预处理就是对源程序进行初步的语法检查

806. C 语言提供的预处理功能包括条件编译，其基本形式为：

```
#XXX 标识符
程序段 1
#else
程序段 2
#endif
```

807. 这里 XXX 可以是【 】。

A. define 或 include

B. ifdef 或 include

C. ifdef 或 ifndef 或 define

D. ifdef 或 ifndef 或 if

808. 下面程序的功能是将从键盘输入的偶数写成两个素数之和。请选择填空。

```
#include<stdio.h>
#include<math.h>
main()
{ int a,b,c,d;
scanf("%d",&a);
for(b=3;b <= a/2;b+=2)
{ for(c=2;c <= sqrt(b);c++) if(b%c==0) break;
if(c > sqrt(b)) d=【 】; else break;
for(c=2;c <= sqrt(d);c++) if(d%c==0) break;;
if(c>sqrt(d)) printf("%d=%d+%d\n",a,b,d);
}
}
```

A. a+b

B. a-b

C. a*b

D. a/b

809. 若运行以下程序时，从键盘输入 3.6 2.4↲，则下面程序的运行结果是【 】。

```
#include<math.h>
#include<stdio.h>
main()
{ float x,y,z;
scanf("%f%f",&x,&y);
z=x/y;
while(1)
{ if(fabs(z)>1.0) {x=y;y=z;z=x/y;}
else break;
}
printf("%f\n",y);
}
```

A. 1.500000

B. 1.600000

C. 2.000000

D. 2.400000

810. 若 x 和 y 都是 double 型变量，且 x 的初值为 3.0,y 的初值为 2.0，则表达式 pow(y,fabs(x))的值为【 】。

811. 设有以下宏定义：

```
#define WIDTH 80
#define LENGTH WIDTH+40
```

812. 则执行赋值语句： v=LENGTH*20;（v 为 int 型变量）后，k 的制式【 】。

813. 设有以下宏定义：

```
#define WIDTH 80
#define LENGTH (WIDTH+40)
```

814. 则执行赋值语句：k=LENGTH*20;（k 为 int 型变量）后，k 的值是【 】。

815. 下面程序运行结果是【 】。

```
#define DOUBLE(r) r*r
main()
{int x=1 ,y=2,t;
t=DOUBLE(x+y);
printf("%d\n",t);
}
```

816. 下面的运行结果是【 】。

```
#define MUL(z) (z)*(z)
main()
{
    printf("%d\\n",MUL(1+2)+3);
}
```

817. 下面程序的运行结果是【 】。

```
#define POWER(x) ((x)*(x))
main()
{ int i=1;
while(i<=4)printf("%d\t",POWER(i++));
printf("\n");
}
```

818. 下面程序的运行结果是【 】。

```
#define EXCH(a，b) {int t；t=a；a=b; b=t;}
main()
{int x=5，Y=9;
EXCH(x,y);
printf("x=%d,y=%d\\n",x,y);
}
```

819. 下面程序的运行结果是【 】。

```
#define MAX(a，b, c)
((a)>(((a)>(b)?(a):(b)))?(a):((a)>(b)?(a):(b)))
main()
{int x，y，z}
x =l; y=2; z =3;
printf("%d，",MAX(x，y，z));
printf(%d,",MAX(x+y,y, y+ x)),
printf(%d\n"，MAX(x，y+z,z));
```

820. 下面程序的运行结果是【 】。

```
#define SELECT(a,b) a<b?a:b
main()
{int m=2 ,n= 4;
printf("%d\n",SELECT(m,n));
}
```

第22章 高级数据表示

821. 有下列程序段：

```
typedef struct node{int data; struct node *next;}*NODE;
NODE p;
```

下列叙述中正确的是【 】。

A. p 是指向 struct node 结构变量的指针的指针；

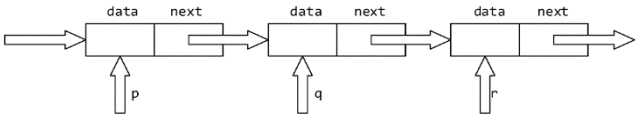
B. NODE p;语句出错；

C. p 是指向 struct node 结构变量的指针；

D. p 是 struct node 结构变量。

822. 有下列结构体说明和变量定义，如图所示，指针 p、q、r 分别指向此链表中的三个连续结点。

```
struct node { int data; struct node *next; }*p, *q, *r;
```



823. 现要将 q 所指结点从链表中删除（但不释放内存），同时要保持链表的连续，下列不能完成指定操作的语句是（ ）。

A. p->next = q->next;

B. p->next = p->next->next;

C. p->next = r;

D. p = q->next;

824. 下列程序运行后的输出结果是【 】。

```
struct NODE
{
    int num; struct NODE *next;
};

main()
{
```

```
    struct NODE s[3] = { { 1, '\0' }, { 2, '\0' }, { 3,
'\0' } }, *p, *q, *r;
    int sum = 0;
    s[0].next = s + 1; s[1].next = s + 2; s[2].next = s;
    p = s; q = p->next; r = q->next;
    sum += q->next->num; sum += r->next->next->num;
    printf("%d\n", sum);
}
```

825. 下列程序运行后的输出结果是【 】。

```
stuct NODE
{ int k;
struct NODE *link; };
main()
{
    struct NODE m[5], *p = m, *q = m + 4;
    int i = 0;
    while (p!= q)
    {
        p->k = ++i; p + +;
        q->k = i + +; q--;
    }
    q->k = i;
    for (i = 0; i<5; i + +) printf("%d", m[i].k);
    printf("\n");
}
```

826. Why can the linked list（链表） in Listing 17.2 (films2.c) be traversed（遍历） in only one direction（方向）? How could you modify（修改） the struct film definition（定义） so that the list could be traversed in both directions（双向）?

827. 以知 head 指向一个带头结点的单向链表,链表中每个结点包含整型数据域(data)和指针域(next). 链表中各结点按数据域递增有序链接,以下函数删除链表中数据域值相同的结点,使之只保留一个(去偶). 请在【 】内填入正确内容.

```
typedef int datatype;
typedef struct node
{datatype data;
```

```
struct node *next;
}linklist;
...
PURGE(linklist *head)
{linklist p,q;
q=head->next;
if(q= =NULL) return;
p=q->next;
while (p!=NULL)
if (p->data= =q->data)
{【 】;free(p);p=q->next;}
else
{q=p;【 】;}
}
```

828. 以下程序实现带有头结点的单链表的建立,链表中每个结点包含数据域data(字符型)和指针域 next. 所建立的头指针由参数 phd 传回调用程序. 请在【 】内填入正确内容.

```
#include "stdio.h"
#include "stdlib.h"
typedef char datatype;
typedef struct node
{ datatype data;
struct node * next;
} linklist;

void CREATLIST(【    】)
{char ch;
linklist * s, *r;
*phd= malloc(sizeof(linklist));
r= *phd;
ch= getchar();
while(ch! = '\$')
{s=malloc(sizeof(linklist));
s->data=ch;
r->next=s;
r=s;
ch=getchar()
}
r->next=【     】;
}
```

```
main()
{linklist * head;
head=NULL;
CREATLIST([&head]);
...
}
```

829. 已知 head 指向一个不带结头的环链表,链表中每个结点包含数据域(num)和指针域(link).数据域存放整数,第 i 个结点的数据域值为 i.

以下函数利用环形链表模拟猴子选大王的过程:从第一个结点开始循环"报数",每遇到 C 的整数倍,就将相应的结点删除(编号为 C 的猴子被淘汰).如此循环直到链表中剩下一个结点,就是猴王.

请在【 】内填入正确内容.

```
typedef int datatype;

typedef struct node
{ datatype data;
struct node*next;
} linklist;
...
int selectking(linklist *head,int c)
{linklist * p, *q; int t;
p=head; t=0;
do
{ t++;
if((t%c) !=0)
{q=p;
【    】;
}
else
{
q->next=【     】;
p=p->next;
}
}
while(【     】);
returen (p->data) ;
}
```

830. 以下函数实现双向链表的建立。链表中每个结点包括数据域（info）、指向后继元素的指针域（NEXT）和指向前趋元素的指针域（pre）。链表的头、尾指针分别在指针数组 a 的两个元素中。链表结点的数据通过键盘输入，当输入数据为-1 时，表示输入结束。请在[ ]内填入正确内容。

```
#include"stdio.h"
#include"stdio.h"
#define LEN sizeof(struct student)
struct student
{int info;
struct student*pre;
struct student*next;
};
int n;
struct student*creat(struct student*a[2])
{ struct student*head,*tail;
struct student*p,*q;
n=0;
p=q=( struct student*)malloc(LEN);
scanf("%d",&p->info);
head=NULL;
while([p->info!=-1])
{n++;
if(n==1) {[head=p];p->pre=p->next=NULL;}
else
{[q->next=p;p->pre=q]}
q=p;
p=( struct student*)malloc(LEN);
scanf("%d",&p->info);
}
q->next=NULL;
tail=q;
a[0]=head;
a[1]=tail;
}
```

831. 以下程序的功能是统计链表中结点的个数，其中 first 为指向第一个结点的指针（链表不带结点）。请在[ ]内填入正确内容。

```
stuct link
{char data;
stuct link*next;
```

```
};
...
stuct link*p,*first;
int c=0;
p=first;
while(【    】)
{【    】;
p=【    】;
}
```

832. 已知 head 指向一个带头结点的单向链表，链表中每个结点包含数据域（data 和指针域（next），数据域为整型。以下函数求出链表中所有链接点域的和值，作为函数值返回。请在【 】内填入正确内容。

```
stuct link
{ int data; stuct link*next}
main()
{ stuct link *head;
sum(head);
}
sum(【    】);
{ stuct link *p; ints=0;
p=head->next;
while(p) {s +=【    】;p=【    】;}
return 0;
}
```

833. 已知 head 指向一个带头结点的单向链表，链表中每个结点包含数据域（data）和指针域（next），数据域为整型。以下过程求出链表所有连接点中，数据域中值最大的结点的位置，有指针变量 s 传回调用程序。请在【 】内填入正确内容。

```
struct link
{int data; stusct link next;}
main()
{ struct link *head, *q;
fmax(head,&q); printf("max=%d\n",q->data);
}
fmax(struct link *head,【    】)
{ struct link *p;
p=(*head).next; *s=p;
while(p!=NULL)
```

```
{p=【    】;
if((*p).data>【    】) *s=p;
}
}
```

834. 已知单链表的第一个结点，以下程序调用函数 print 输出这一单向链表。请在【 】内填入正确内容。

```
#include"stdio.h"
#include"stdio.h"
struct student
{int info;
struct student*link;
};
void print(struct student*head)
{ struct student*p;
printf("\n the linklist is:");
p=head;
if(head!=NULL)
do
{printf("%/d",[p->info]);
p=【    】;
}
while([p!=NULL]);
}
main()
{ struct student * head;
...
print(head);
...
}
```

835. 以下 min3 函数的功能是：计算循环单链表 first 中每 3 个相邻结点数据域中的和，并返回其中最小值。请在【 】内填入正确内容。

```
struct node{int data;
struct node*link;
};
int min3(struct node*first)
{struct node*p=first;
int m,m3=p->data+p->link->data+p->link->link->data;
for(p=p->link;p!=first;p=【    】)
{ m=p->data+p->link->data+p->link->link->data;
```

```
if(【    】) m3=m;}
return【    】;
}
```

836. 已知 head 指向单链表的第一个结点，以下函数完成往降序单向链表中插入一个结点，插入后链表仍有序。请在【 】内填入正确内容。

```
#include "stdio.h"
struct student
{ int info;
struct student*link;
};
struct student*insert(struct student*head,struct
student*stud)
{ struct student*p0,*p1,*p2;
p1=head;
p0=stud;
if(head==NULL)
{ head=p0; p0->link=NULL;}
else
while((p0->info<p1->info)&&(p1->link!=NULL))
{ p2=p1; p1=p1->link;}
if(p0->info>=p1->info)
{ if(head==p1){【    】;head=p0;}
else {p2->link=p0;【    】;}
}
else
{ p1->=link=p0;【    】;}
return 【    】 ;
}
```

837. 以下函数完成往降序链接的双向链表中插入一个结点，待插结点的地址由调用程序传给参数 stud。链表中每个结点包含数据(info)、后继元素指针域(next)和前趋元素指针域(pre)。 链表的头、尾指针分别放在指针数组 a 的两个元素中。请在（ ）内填入正确内容。

```
#include "stdio.h"
#include "stdlib.h"
struct student
{ int info;
struct student*pre;
struct student*next;
```

```
};
void insert(struct student*a[2],struct student*stud)
{ struct student*p0,*p1,*p2;
struct student*head,*tail;
head=a[0];
tail=a[1];
p1=head;
p0=stud;
if(head==NULL)
{ head=p0;【    】}
else
while((p0->info<p1->info)&&(p1->next!=NULL))
{ p2=p1;p1=p1->next;}
if(p0->info>=p1->info)
if(p1==head)
{ p0->next=head;head->pre=p0;【    】p0->pre=NULL;}
else
{ p2->next=p0;p0->next=p1;p1->pre=p0;【    】}
else
{p1->next=p0;p0->pre=p1;tail=p0;p0->next=NULL;}
a[0]=head;
a[1]=tail;
}
```

838. 已知 head 指向单链表的第一个结点，以下函数 del 完成从单向链表中删除值为 num 的第一个结点。请在【 】内填入正确内容。

```
#include "stdio.h"
struct student
{ int info;
struct student*link;
};
struct student*del(struce student*head,int num)
{ struce student*p1,*p2;
if(head==NULL)
printf("\nlist null! \n");
else
{ p1=head;
while(【    】)
{ p2=p1;p1=p1->link;}
if(num==p1->info)
{ if(p1==head) 【    】;
else【    】;
```

```
printf("delete:%d\n",num);
}
else printf("%d not been found! \n",num);
}
return【    】;
```

839. **选做** 以下函数完成从双向链表中删除值为 num 的一个结点。链表中没个结点包含数据域（info）、后继元素指针域（next）和前趋元素指针域（pre）。链表的头、尾指针分别放在指针数组 a 的两个元素中。请在【 】内填入正确内容。

```
#include"stdio.h"
struct student
{ int info;
struct student *pre;
struct student *next;
};
void del (struct student *a[2], int num)
{ struct student *p,*q;
struct student *head,*tail;
head=a[0];
tail=a[1];
if(head==NULL) printf("\n list null ! \n");
else
{
p=head;
while (【 】)
{q=p;p=p->next;}
if(num==p->info)
{if(p==head)
{head=p->next;【 】}
else if (p==tail)
{q->next=NULL;【 】}
else
{p=p->next;q->next=p;p->pre=q;}
}
else printf ("%d not been found! \n",num);
}
a[0]=head;
a[1]=tail;
}
```

840. **选做** 编写程序实现单向链表的：从头部插入法新建链表、从尾部插入法新建链表、按位置查找链表的指定位置、按内容查找指定值在链表的位置、插入链表、删除链表、链表长度、链表显示等。

841. 设链表的定义为：

```
typedef struct Node {
    int data;
    struct Node *next;
} Node, *LinkList;
```

具体地，需要实现下列函数（用课上所述的方法，先阅读群共享中的数据结构(第 2 版)\02\LinkList.h，作图，再手写到草稿纸上，人工验证无误后，并上机验证后誊写在作业纸上。注意：应尽量处理输入错误的情况。）：

a. 编写 LinkList CreatFromHead()函数，实现输入多个元素，将新元素插入到头部（按顺序输出时表现为反序），输入以 EOF 结束。

b. 编写 LinkList CreatFromTail()函数，实现输入多个元素，将新元素插入到尾部（按顺序输出时表现为顺序），输入以 EOF 结束。

c. 编写 Node *Get(LinkList LL, int loc)函数，实现查找 LL 链表中的第 loc 个元素，并返回该元素的指针。

d. 编写 Node *Locate(LinkList LL, int e, int *loc)函数，实现查找 LL 链表中的第一个满足值为 e 的元素，将其位置保存至指针 loc 中，并返回该元素的指针。

e. 编写 int InsList(LinkList *LL, int loc, int e)函数，实现在 LL 链表中的第 loc 个元素后插入一个元素，该元素值为 e，并返回运行是否正确。

f. 编写 int DelList(LinkList *LL, int loc, int *e)函数，实现在 LL 链表中删除第 loc 个元素，并将其值保存至指针 e 指向的位置（用于避免误删除），并返回运行是否正确。注意：删除后应随手释放内存。

g. 编写 int ListLength(LinkList LL)函数，实现计算 LL 链表的长度，并返回。

h. 编写 int Display(LinkList LL)函数，实现依次遍历 LL 链表，显示其每个元素的 data 成员的值。

i. 编写 void Destory(LinkList LL)函数，实现释放 LL 链表中各个元素的内存空间。

842. 请默写用尾插法新建链表的程序，输入若干个二位平面上的坐标，以 q 结尾，并逐一显示。数据定义如下：

```
typedef struct point_list {
    double x;
    double y;
    struct point_list * next;
} point_list;
```

843. 应默写的程序如下：

```
#include <stdio.h>
#include <stdlib.h>

typedef struct point_list {
    double x;
    double y;
    struct point_list * next;
} point_list;

point_list * create_linklist_from_tail();
void show_linklist(point_list *head);

int main()
{
    point_list * point;

    point = create_linklist_from_tail();
    show_linklist(point);
    return 0;
}

point_list * create_linklist_from_tail()
{
    point_list * head = NULL, * current = NULL, * prev =
NULL;
    double x, y;
```

```
    puts("Enter points (end with 'q'):");
    while (scanf("%lf %lf", &x, &y) == 2)
    {
        current = (point_list *)malloc(sizeof(point_list));
        current->x = x;
        current->y = y;
        current->next = NULL;
        if (head == NULL)
            head = current;
        else
            prev->next = current;
        prev = current;
    }
    return head;
}

void show_linklist(point_list * head)
{
    point_list *cur = head;
    while (cur)
    {
        printf("(%.2lf, %.2lf)\n", cur->x, cur->y);
        cur = cur->next;
    }
}
```

第23章 复习课 (1)

特别说明：这份试题为《C 程序设计期中模拟测试题》，设：□为空格；↲为回车。下同。

一、单项选择题（60 分）

844. 以下叙述中正确的是【 】。

A. C 程序中的注释只能出现在程序的开始位置和语句的后面。

B. C 程序书写格式严格，要求一行内只能写一个语句。

C. C 程序书写格式自由，一个语句可以写在多行上。

D. 用 C 语言编写的程序只能放在一个程序文件中。

845. 一个结构完整的 C 程序代码并非必须有【 】。

A. 函数

B. 语句

C. 表达式

D. 变量

846. 语句包括选择语句、循环语句、跳转语句，但不包括【 】。

A. 声明语句

B. 标记语句

C. 表达式语句

D. 预定义语句

847. 下列选项中，不是合法变量名的是【 】。

A. `$1`

B. `printf`

C. `@a`

D. main

848. 对于 char str[40]="Good!"; , sizeof(str) 和 strlen(str) 的值分别为【 】。

A. 40;5

B. 6;5

C. 40;6

D. 5;5

849. 设 x 为双精度浮点型且值为 256.7 , 运行 x=(char)x-(double)5.2; 之后 x 值为【 】。

A. −5.2

B. 251.5

C. 250.8

D. 251

850. 设 x 为双精度浮点型，则运行 x=(0/0, 10/4); 之后， x 值为【 】。

A. 2

B. 2.5

C. 3

D. 编译错误

851. 若 , , w=3.2，x=3.5，y=6.7 , 表达式 (w)?(--x):(y+1) 的值为【 】。

A. 0

B. 1

C. 2.5

D. 7.7

852. 使用 #define PI 3.141519 预编译指令，下列语句正确的是【 】。

A. PI=3.14;

B. 2*PI;

C. PIPI/PI;

D. PI*=2;

853. 运行 int x=5; char c; res=scanf("%d\n", &x); c=getchar(); ，输入 u↲，则【 】。

A. c 不能被正确赋值

B. 回车后输入数字方可正确赋值 x

C. x 的值不确定，程序等待输入

D. x 仍为 5 ， c 的值为 117（ u 的 ASCII 码为 117 ）

854. 运行 res=scanf("%*d %*d %d", &x); 并在运行期间输入 1 ，则 res 值为【 】。

A. 0

B. 1

C. 2

D. 3

855. 运行 float x=1e38;之后，x 的值为【 】。

A. 1×10+38 以内的数

B. 编译错误

C. 1×10+38

D. +∞

856. 表达式 -7/-4-7.0/4 的值为【 】。

A. 0

B. −0.75

C. 0.25

D. 0.75

857. 下列选项中可作为 C 语言合法常量的是【 】。

A. -.80

B. -080

C. -8e1.0

D. -80.0e

858. 执行 int i = 3; int ans = (++i)+(++i)+(++i); 之后， ans 值为【 】，i 值为【 】。

A. 16; 6

B. 18; 6

C. 不确定; 6

D. 不确定; 不确定

859. 执行 int x=3, y=6; printf("%d, %d, %d", x*y, ++x, y--); 的输出为【 】。

A. 18, 3, 6

B. 18, 4, 5

C. 20, 4, 5

D. 20, 4, 6

860. 执行 int x=-1, y=6; printf("%d", (++x?y--:++x*y)); 的输出为【 】。

A. 0

B. 5

C. 6

D. -6

861. 若变量已正确定义，在 if (W) printf("%d\n", k);中,以下不可替代 W 的是【 】。

A. return 0

B. ch=getchar()

C. a==b+c

D. 3

862. 若有定义语句 int a, b; double x;，则下列选项中没有错误的是【 】。

A.

```
switch((int)x%2) { case 0: a++; break; case 1: b++; }
```

863. B.

```
switch((int)x/2.0) {case 0: a++; break; case 1: b++; }
```

864. C.

```
switch(x%2) { case 0: a++; break; case 1: b++; }
```

865. D.

```
switch((int)(x)/2) {case 0.0: a++; break; case 1.0: b++; }
```

866. 执行 int x=100; while (x++<101) printf("%d ",x); printf("%d",x); 的输出是【 】。

A. 100 101

B. 100 101 102

C. 101

D. 101 102

867. 关于 C 语言使用 int a[20]; a[i]++; 的说法正确的是【 】。

A. i 应在[0,19]之间

B. a[1]为首个元素

C. 可声明 i 为实数

D. i 可以为负整数

868. 执行 `char *a="1113"; short *p=(short*)a; printf("%d\n",*p);` 的输出是【 】。（提示：字符'1'的 ASCII 码为 49。）

A. 49

B. 1113

C. 12593

D. 0

869. 执行 `printf("%#o\n",*"1113");` 的输出是【 】。

A. 061

B. 61

C. 1

D. 0

870. 设有程序片段 `int arr[10][3]={0}; printf("%d\n",&arr[3][2]-arr);`，则【 】。

A. 输出 11

B. 输出 32

C. 输出 2

D. 编译错误无法执行

871. 在 Ubuntu Server 18.04 的 x64 版本上将某 C 代码编译为 64 位程序，运行时 `sizeof(void*)` 的值为【 】。

A. 0

B. 4

C. 8

D. 64

872. 设有语句 `int a[10]={0}, *p=a;`，则【 】是对 a 数组元素的错误引用，0≤i<10 。

A. a[p-a]

B. *(*(a+i))

C. p[i]

D. *(&a[i])

873. 设有语句 int a[10]={0}, *p=a; 【 】 可以通过编译。

A. a++;

B. a-1;

C. p*=2;

D. a+p

874. 在 Ubuntu Server 18.04 中，标准输入流是【 】的。

A. 有缓存有回显

B. 有缓存无回显

C. 无缓存有回显

D. 无缓存无回显

875. 在 Ubuntu 中执行可执行程序 ex，通过【 】可将标准输出流附加文件尾。

A. ./ex < 1.txt

B. ./ex << 1.txt

C. ./ex > 1.txt

D. ./ex >> 1.txt

876. 对字符型变量 x，有【 】种取值可使 if (0<=x<1) putchar('*'); 输出星号。

A. 1

B. 127

C. 128

D. 255

877. 以下叙述中正确的是【 】。

A. int 型和 double 型变量对其值超过上限（溢出）处理方法类似。

B. 所有 scanf 格式字符串都能忽略标准输入中的多余空白字符。

C. 语句 do a++; while (a); 总比 while (a) a++; 的循环体多执行一次。

D. C 程序表达式之间可以相互替换和嵌套，语句之间也类似，但二者不可混用。

878. 对于程序 main() { int i=5; do { if (i%3==1) if (i%5==2) { printf("%d",i); break;} i++; } while(i); printf("\n"); }，输出为【 】。

A. 7

B. 35

C. 5

D. 以上答案都不对

879. 若有定义 int x=0,*p=&x; ,则语句 printf("%d\n",*p); 的输出结果是【 】。

A. 随机值

B. 0

C. x 的地址

D. p 的地址

880. 有下列程序片段 void sum(int a[]) { a[0]=a[-1]+a[1]; }，执行 int a[10]={2,4,6,8,10,12,14,16,18,20}; sum(&a[2]); printf("%d\n",a[2]); 后得到输出【 】。

A. 12

B. 14

C. 10

D. 18

881. 设有函数 void swap(x, y) { int t=x; x=y; y=t; }，则执行 x=5; y=6; swap(x,y++); 后 x，y 的值为【 】。

A. 5,6

B. 5,7

C. 6,5

D. 7,5

882. 关于 C 语言的函数说法正确的是【 】。

A. 函数不能没有参量

B. 函数必须有返回语句

C. 函数可以有任意多个参量

D. 函数必须做到高耦合低内聚

883. 设有函数 int f(int x) { int y; if (x==0 || x==1) return (3); y=x*x-f(x-2); return y; }，则 f(3) 的值为【 】。

A. 3

B. 6

C. 8

D. 9

884. 设有整型变量 p，q，表达式“scanf("%d,%d", &p, &q)”在键盘输入【 】时值为 2。

A. 7Ω3

B. 3Ω,5

C. 5↲3

D. 5,↲9

885. 以下错误的定义语句是【 】。

A. int x[ ][3]={{0},{1},{1,2,3}};

B. int x[4][3]={{1,2,3},{1,2,3},{1,2,3},{1,2,3}};

C. int x[ ][3]={1,2,3,4};

D. int x[4][ ]={{1,2,3},{1,2,3},{1,2,3},{1,2,3}};

886. 以下叙述中正确的是【 】。

A. 标记为 const 的常量，在程序运行期间其值不变。

B. 程序中可以同时存在 int sum(int arr[], int n); 和 int sum(int *arr, int n); 。

C. 程序中可以同时存在 int sum(int arr[]); 和 int sum(int *arr, int n); 。

D. 变长数组在运行期间，长度视情况随时改变。

二、填空题（30 分，每题 2 分）

887. 以下函数 sstrcat() 实现了字符串的连接，即将 from 所指字符串复制到 to 所指字符串的尾部。例如：字符串 to 为 "ab"，字符串 from 为 "eh"，调用后 to 为 "abeh"，from 不变。

```
#include <string.h> void sstrcat(char *to, char *from) { int toLen = strlen(to); while (*(to+toLen)= 【41】 ) { from++, to++; } }
```

888. 以下程序的输出结果是【42】。

```
#include <stdio.h> void swap(int *a,int *b) { int *t; t=a; a=b; b=t; } int main() { int i=3,j=5,*p=&i,*q=&j; swap(p,q); printf("%d %d\n",*p,*q); }
```

889. 用 C 语言书写表达式

890. s={v0t+12at2,t≥0;0,t<0.

891. 为【43】。

892. 古希腊哲学家芝诺说：一支箭从弓箭手手里射出去，理论上这只箭永远到达不了靶子上。因为这支箭永远是在上一时刻靶子和终点的中间位置，永远到不了靶子。请填空。

```
#include <stdio.h> int main(void) { int t_ct; // term count double time, power_of_2; int limit; printf("Enter the number of terms you want: "); scanf("%d", 【44】 ); for ( 【45】 ; t_ct <= limit; 【46】 ) { time += 1.0/power_of_2; printf("time = %f when terms = %d.\n", time, t_ct); } return 0; }
```

893. 字符型十进制数 −84 转换二进制为【47】，转换十六进制为【48】，有符号字符型取值范围为【49】。

894. 声明 int a=5,b=2; 表达式 !(a=2)&&b==1&&a+b 的值是为【50】。

895. 声明 double x=-0.536; 请书写 printf 语句【51】，得到输出 -0.54ΩΩ，并使得当 x=81.6 时，输出为 +81.60Ω。

896. 设函数 fun() 将十进制正整数 m 转换成 k 进制数，并按位输出。下列程序无法获得正确输出，请修正三处错误：【52】，【53】，【54】（书写该行修正后的语句。不得增删语句或改变程序结构），并对编程风格提出修改意见【55】。

```
void fun(int m, int k); { int aa[20], i; for (i = 0; m; i++) { aa[i] = m / k; m /= k; } for (; i; i--) printf("%d", aa[i]); }
```

三、程序设计题（10 分）

897. 不借助字符串库函数，重新书写字符串复制函数 strcpy()，应力求完整、易用。

898. 【56】

第24章 复习课 (2)

899. 函数 printf()中用到格式符%5s。如果字符串长度大于 5，则输出按方式（ ）

A. 从左起输出该字串，右补空格；

B. 按原字符长从左向右全部输出；

C. 右对齐输出该字串,左补空格；

D. 输出错误信息。

900. 以下叙述不正确的是（ ）

A. 一个 C 源程序可由一个或多个函数组成；

B. 一个 C 源程序必须包含一个 main 函数；

C. C 程序的基本组成单位是函数；

D. 在 C 程序中，注释只能位于一条语句的后面。

901. 若有程序段 for (m=0;m<=60;m++) { scanf("%d",&y); if(y<0) continue; printf("%3d",m); }，正确的描述是（ ）。

A. 当 y 小于 0 时整个循环结束；

B. 当 y 小于 0，则不输出；

C. 函数 printf()函数永远不执行；

D. 最多允许输出 60 个非负整数。

902. 用 C 语言书写表达式 v=4πr23，应为（ ）.

903. C 语言程序经过（ ）得到目标文件，再经过（ ）得到可执行程序。

904. int a=5, b=2;表达式!(a=2)&&b=1&&0 的值是（ ）。

905. 语句 printf("u\103Fx\n");的输出结果是（ ）。

906. 假设所有变量均为整型,则表达式(a=2,b=5,b++,a+b)的值是（ ）。

907. 在 if (x) y();语句中与条件表达式 x 等价的是（ ）。

908. 若有以下类型说明语句：char a; int b; float c; double d;则表达式 a*b+d-c 的结果类型为（ ）。

909. 请写出以下运算符的优先级顺序、并写出方向为从右向左的运算符：

910. 1 逗号、2 赋值、3 加减、4 乘除、5 尺寸（sizeof）、6 模、7 大（小）于号、8 大（小）于等于号、9 等号、10 不等号、11 小括号、12 中括号（用于数组下标）、13 负号、14 自增减（如：+=）、15 自乘除、16 逻辑与（或）、17 逻辑非、18 条件运算符、19 自增减量（如：++）

911. 请按优先级顺序排列，不同级别之间用逗号隔开，同一级别的以括号表示。如：9 (10)。

912. 如 float x=4.3; int a; a=4*x+(int)x;则 a 的值为（ ）。

913. 找出一个小于给定整数 m 且紧随 m 的素数，请填空。

```
#include <stdio.h> int fun(int m) { int i,k; for (____[1]____;i>=2;i--)
{ for(k=2;k<i;k++) if(i%k==0) ____[2]____; if(k==i) return(i); } } main()
{ int n; printf("\nplease enter n:"); scanf("%d", &n);
printf("max=%d\n",fun(n)); }
```

914. 请写出如下程序的输出结果（ ）。

```
main() { char c,lc=' ',string[81]={"VHS SVHS V8 Hi8"}; int i,j=0;
for(i=0;(c=string[i])!='\0';i++) { if(c!=' '&& lc==' ') j++; lc=c; }
printf("%d\n",j); }
```

915. C 语言程序中可以有多少个头文件？多少个主函数？多少个.c 文件？多少个变量？

916. 请写出下列情况下，应声明何种类型的变量，内存使用率最高（如：对于学生是否及格，选择 char、int 和 long double 都可以，但 char 的内存使用率为 1/8，最节省内存）：

a. 学生成绩（满分为 100 分）；

b. 体温计的读数；

c. 一个汉字；

d. 我院学生学号；

917. e. 我院 C 语言分班人数；

918. f. 我国人口（概数）；

919. g. 一次评审评分（-2、-1、0、1、2）。

920. 请写出在第 2 题所述情况下，应声明何种类型的 scanf 语句。

921. 字符串的长度是否为字符串数组尺寸（sizeof）减一?

922. 循环判定的表达式 X 不变，do A; while (X);与 while(X) A;的循环次数是何种大小关系?

923. 设有如下程序片段：

```
switch (ch) { default: printf("D"); case 'a': printf("C"); break; case 'b': printf("S"); }
```

924. 在 GCC 下，请在 switch 语句前分别加一条语句，使得程序：

a. 输出 D；

b. 输出 DC；

c. 输出 DCS；

d. 输出 C；

e. 输出 S；

f. 不输出。

925. 设主程序包含预编译指令#define S(x,y) x*y，则程序中 int a=2,b=5;S(a+b,a-b)的值是多少?

926. 设有语句 if (A) B; else if (C) if (D) E; else F;，请写出在何种情况下运行语句 F?

927. 设有语句 index = 1; while(--index < 5) A;，则当 index 分别为 int、float 和 unsigned short 时，语句 A 各被执行多少遍?

928. 请写出：-7./4.、-7./4、-7/4.、-7/4 的值。（注意数字后是否有圆点）

929. 编写以下四个函数，当主程序从键盘输入一串二进制数据（0或1的序列，长度小于 64），可以分别调用该函数，求出：

a. 字符串的长度；

b. 字符串中指定字符（0 或者 1）的个数；

c. 求该二进制数的值；

d. 校验该序列中输入出错的情况，并忽略之；

930. 编写命令行，将上述程序编译、运行，再运行一次将输入输出重定向到文件中。

931. 素数（即：质数）指在一个大于 1 的自然数中，除了 1 和此整数自身外，无法被其他自然数整除的数。输入两个非负整数 m 和 k（假设 k 不超过 100），请从 m 开始（包含 m）计算其后的 k 个素数，并按照从大到小的顺序输出这 k 个素数。重复上述运算过程，程序直到输入的 m 或 k 其中之一为 0 时，运行结束。