一　选择题和填空题

数组定义和引用

数组长度不能为0

1. 以下能对一维数组进行正确初始化的是（A）。
   A. int a[] = {};　　B. int a[] = {0};　　C. int a[10] = {10 * 1};　D. int a[5] = {1, 2, 3, 4};
2. 对以下定义不正确的理解是（ ）
   int a[5] = {1, 2, 3};
   A. 将 3 个值依次赋值给 a[1] ～ a[3]
   B. 将 3 个值依次赋给 a[3] ～ a[5]
   C. 因为数组长度与初值个数不相同，所以语句不正确
   D. 将 3 个值依次赋值给 a[0] ～ a[2]
3. 若有定义语句：int m[] = {5, 4, 3, 2, 1}, i = 4; 则下列对 m 数组元素的引用中错误的是（C）。
   A. m[--i]　　B. m[2 * 2]　　C. m[m[0]]　　D. m[m[i]]

   越界
4. 若有定义 int x[7] = {1, 2, 3, 4, 5, 6, 7}, y = 6; 则正确的语句是（ ）。
   A. y = x[2] + 1;　　B. y = x[3];　　C. x[0] = y;　　D. y = x + 1;
5. 若有初始化语句 int a[][3] = {1, 2, 3, 4, 5, 6, 7}; 则以下正确的是（ ）。
   A. 数组中有 9 个元素　　B. 引用 a 数组时下标均不能超过 2
   C. 数组中有 7 个元素　　D. 数组第一维大小为 3
6. 以下可对二维数组 a 进行正确初始化的语句是（ ）。
   A. int a[2][3] = {{1, 2}, {3, 4}, {5, 6}};
   B. int a[3][4] = {0};
   C. int a[][3] = {1, 2, 3, 4, 5, 6};
   D. int a[][3] = {{1, 2}, {0}};
7. 对二维数组初始化错误的是（ ）。
   A. int b[3][3] = {{1, 2}, {4, 5, 6}, {0}};
   B. int b[2][] = {{1, 2, 3}, {4, 5, 6}};
   C. int a[][3] = {{1, 0, 1}, {}, {2, 3}};
   D. int b[][3] = {{1, 2, 3}, 4, 5, 6};
8. 若 float a[6][2]; 则数组 a 中各元素的值为（ ）。
   A. 不能得到确定的值
   B. 编译或者运行阶段初值都是 0
   C. 在程序编译阶段得到初值 0
   D. 在程序运行阶段得到初值 0
9. 二维数组 a 有 m 列，则计算任意元素 a[i][j] 在数组中的位置的公式为（ ）（设 a[0][0] 位于数组的第一个位置上）。
   A. j * m + i　　B. i * m + j + 1　　C. i * m + j - 1　　D. i * m + j
10. 已知有程序：

```
#include <stdio.h>
int main()
{
   int p[8] = {11, 12, 13, 14, 15, 16, 17, 18};
   int i = 0, j = 0;
```

```
    while(i++ < 7)
        if(p[i]%2)    j += p[i];
    printf("%d\n", j);
    return 0;
}
```

上面程序的输出结果是（ ）。

A. 42　　B. 45　　C. 60　　D. 56

11. 程序段

```
#include <stdio.h>
int main()
{
    int i, j = 50, a[] = {7, 4, 10, 5, 8};
    for( _________ )
        j += a[i];

    printf("%d", j-40);
    return 0;
}
```

若希望上面程序运行后输出 25，程序空白处的正确选项是（ ）。

A. i = 4; i > 2; i--　　B. i = 2; i < 4; ++i　　C. i = 1; i < 4; ++i　　D. i = 1; i < 3; ++i

12. 有程序段

```
#include <stdio.h>
void reverse(int a[], int n)
{
    int i, t;
    for(i = 0; i < n/2; i++)
    {
        t = a[i];
        a[i] = a[n-1-i];
        a[n-1-i] = t;
    }
}

int main()
{
    int b[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
    int i, s = 0;
    reverse(b, 10);
    for(i = 0; i < 3; i++)
        s += b[i];
    printf("%d\n", s);
    return 0;
}
```

上面程序输出的结果是（ ）。

A. 6　　B. 21　　C. 9　　D. 27

13. 设有程序段：

```
int i;
int a[3][4] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};
for(i = 0; i < 3; i++)
    printf("%d ", a[i][2-i]);
```

则上面语句的输出结果是（ ）。

A. 4812　　B. 246　　C. 369　　D. 579

14. 设有程序

```
#include <stdio.h>
int main()
{
    int i, f[10];
    f[0] = 1;
    f[1] = 1;
    for (i = 2; i < 10; i++)
        f[i] = f[i-2] + f[i-1];
    for (i = 0; i < 10; i++)
        printf("%3d", f[i]);
    return 0;
}
```

斐波那契数列

上面程序输出的结果是 _____

一维数组和指针

15. 若有定义: int a[5], *p = a; 则对 a 数组元素地址的正确引用是（ ）。

A. *a + 1　　B. &a[0]　　C. p + 5　　D. &a + 1

越界　　移动到数组后了

16. 若有定义: int a[10], *p = a; 则 *(p+6) 和 p+6 分别表示（ ）。

A. 元素 a[6] 的值和元素 a[6] 的地址　　B. 元素 a[6] 的值和元素 a[7] 的地址

C. 元素 a[7] 的值和元素 a[6] 的地址　　D. 元素 a[7] 的值和元素 a[7] 的地

17. 若有定义 char s[10]; char *p = s; 则下面表达式中错误的是（ ）。

A. &s[1]　　B. &s[0] + 1　　C. s++　　D. s + 1

数组名是常量指针，不能++,--,赋值

指针变量不能加减

18. 已知：char s[10], *p=s，则在下列语句中，错误的语句是( )。

A) p=s+5；　B) s=p+s；　C) s[2]=p[4]；　D) *p=s[0]；

19. 已知：char s[6], *ps=s；则正确的赋值语句是 。

A) s="12345"；　B) *s="12345"；　C) ps="12345"；　D) *ps="12345"；

20. 若有定义：int *p[4]；则标识符关于 p 说法错误的是（ ）。

A. 定义不合法

B. 是一个指针，它指向一个含有四个整型元素的一维数组

C. 是一个指针数组名

D. 是一个指向整型变量的指针

21. 设有程序段

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

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

    return 0;
}
```

上面程序输出的结果是（ ）。

A. 5　　B. 7　　C. 4　　D. 6

22. 设有程序段

```
char str[] = "Hello";
char *ptr = str;
```

执行上面的程序段后, *(ptr+5) 的值（ ）。

A. 'e'　　B. 'l'　　C. '\0'　　D. 'o'

23. 设有程序段

```
#include <stdio.h>
int main()
{
    int a[] = {1, 2, 3, 4}, y, *p = &a[3];
    --p;
    y = *p;
    printf("y = %d\n", y);
    return 0;
}
```

上面程序输出的结果是（ ）。

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

二维数组和指针

24. 若有以下定义和语句: int a[2][3], (*p)[3]; p = a;，则对 a 数组元素的正确引用为（ ）。

A. *(p[1] + 1)　　B. p[1] + 2 地址　　C. *(*(p+2) + 1) 越界　　D. *(p+1)[0]

25. 若有定义：int a[2][3]; 则对 a 数组的第 i 行第 j 列（假设 i, j 已正确说明并赋值）元素值的正确引用为（ ）。

A. *a + 1　　B. &a + 1　　C. *(*(a+i)+j)　　D. &a[0]

26. 若有定义和语句: int a[3][3], (*p)[3]; p = a; 则对 a 数组元素地址的错误引用为（ ）。

A. p[2]　　B. *(p+2)　　C. p[1] + 1　　D. (p+1) + 2

27. 设有语句：int array[3][4]; 则在下面几种引用下标为 i 和 j 的数组元素的方法中，不正确的引用方式是( )

A）array[i][j]　　B）*(*(array + i) + j)

C）*(array[i] + j)　　D）*(array + i*4 + j)　　正确的是*(*array+i*4+j)

28. 若二维数组 a 有 m 列，则在 a[i][j]之前的元素个数为（ ）。

A) j*m+i　　B) i*m+j　　C) i*m+j-1　　D) i*m+j+1

29. 已知：static int a[3][4];则数组 a 中各元素（ ）。

A) 可在程序运行阶段得到初值 0

B) 可在程序编译阶段得到初值 0

C) 不能得到确定的初值

D) 可在程序的编译或运行阶段得到初值 0

30. 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. 说法不合法

C. 是一个指针数组名

D. 是一个指针，它指向一个含有四个元素的一维数组

31. 已知：int a[4][3]={1,2,3,4,5,6,7,8,9,10,11,12};

int (*ptr)[3]=a,*p=a[0];

则以下能够正确表示数组元素 a[1][2]的表达式是（ ） 。

A) *(*(a+1)+2)　　B) *(*(p+5))　　C) (*ptr+1)+2　　D) *((ptr+1)[2])

应该是 *(p+5)

32. 设有程序段

```
#include   <stdio.h>
int main()
{
    int a[][3] = {{1, 2, 3}, {4, 5, 0}},(*pa)[3], i;

    pa = a;
    for(i = 0; i < 3; i++)
      if(i <= 2)
        *(*(pa+1) + i) = *(*(pa+1) + i) - 1;
      else
        *(*(pa+1) + i) = 1;
    printf("%d\n", *(*(pa+1) + 0) + *(*(pa+1) + 1) + *(*(pa+1) + 2));
    return 0;
}
```

p[1][i]

上面程序输出的结果是（ ）。

A. 6　　B. 8　　C. 无确定值　　D. 7

字符串和字符数组

33. 给出以下定义：

char x[]="abcdefg";

char y[]={'a','b','c','d','e','f','g'};

则正确的叙述为( )

A)数组 X 和数组 Y 等价　　B)数组 x 和数组 Y 的长度相同

C)数组 X 的长度大于数组 Y 的长度　　D)数组 X 的长度小于数组 Y 的长度

34.不能把字符串:Hello!赋给数组 b 的语句是（ ）

A) char b[10]={'H','e','l','l','o','!'};　　B) char b[10];b="Hello!";

没'\0'

C) char b[10];strcpy(b,"Hello!");　　D) char b[10]="Hello!";

35. 对字符数组 str 赋初值，str 不能作为字符串使用的一个是（　）。

A) char str[]="shanghai";

B) char str[]={"shanghai"};

C) char str[9]={'s','h','a','n','g','h','a','i'};

D) char str[8]={ 's','h','a','n','g','h','a','i'};　没'\0'

36. 判断字符串 s1 是否大于字符串 s2，应当使用（　）。

A)　if (s1 > s2)　　B)　if (strcmp(s1, s2))

C)　if (strcmp(s2, s1) > 0)　　D)　if (strcmp(s1, s2) > 0)

37.　下面不能正确进行字符串赋值操作的是（　）。

A. char *s; scanf("%s", s);　s没指向　　B. char *s = "ABCDE";

C. char s[5] = { 'A' , 'B' , 'C' , 'D' , 'E' };　没'\0'　D. char s[5] = {"ABCDE"};

38. 已知：　char　a[3][10]={"BeiJing","ShangHai","TianJin"}，*pa=a；不能正确显示字符串"ShangHai"的语句是　。

A) printf("%s",a+1);　　B) printf("%s",*(a+1));

C) printf("%s",*a+1);　　D) printf("%s",&a[1][0]);

39.　设有程序段

```
char a[] = "abcdefgh";
char *p = a;
p += 3;
printf("%d\n", strlen(strcpy(p, "ABCD")));
```

拷贝后a变成"abcABCD",但strcpy返回的是p，从也就是i会从str【3】显示字符串

上面程序段的运行结果（　）。

A. 8　　B. 4　　C. 12　　D. 7

40.　设有程序段

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

程序段运行结果的是（　）。

A. cde　　B. 字符　c　　C. 字符　c　的地址　　D. 无确定的输出结果

41.　若有定义　char str[] = "ABCDE\0FGH ", *p = str;，则语句　printf("%c", *p+7);　的输出结果是 _H_____，语句　printf("%s", str);　的输出结果是 _ABCD___。

42.　下面程序段的输出结果是“ a b c d e”，请填空。

```
char array[6] = {"ABCDE"}, *p = array;
int i;
for (i = 0; i < sizeof(array)/sizeof(char); _i++____ )
    printf("%c ", *p + 32);
```

43.　设有程序段

```
char str[] = "ABC", *p = str;
printf("%d\n", *(p+3));
```

上面程序段的运行结果（　）。

A.　字符　C　　B.　字符　C　的地址　　C. 0　　D. 67

44.　设有程序段

```
#include <stdio.h>
#include <string.h>
int main()
```

```
{
    char *s1 = "AbDeG";
    char *s2 = "AbdEg";
    s1 += 2; s2 += 2;
    printf("%d\n", strcmp(s1, s2));
    return 0;
}
```

上面程序输出的结果是（ ）。

A. 零　　B. 负数　　C. 正数　　D. 任意值

45. 设有程序段

```
char *s = "\t\018bc";
for(; *s! = '\0' ; s++) {（ ）}
```

长度为5，‘ \018是两个字符,8不是8进制的字符，\01是一个字符，8是一个字符

上面程序段中 for 循环的执行次数是（ ）。

A. 5　　B. 6　　C. 7　　D. 9

46. 设 char *s = "\ta\017bc"；则指针变量 s 指向字符串的所占据的字节数是（ ）。 \0也占字节

A. 9　　B. 5　　C. 6　　D. 7

47. 若有语句：char s1[] = "string", s2[8], *s3, *s4 = "string2"；则对库函数 strcpy 的正确调用是（ ）。

A. strcpy(s1, "string");　　B. strcpy(s4, "string1");　　s4保存的是文字常量区的地址，无法往其中写入

C. strcpy(s3, "string1");　　s3没初始化　　D. strcpy(s1, s2);　　s2没有初始化，不能当字符串用

48. 设有下面程序段,则下列叙述中错误的是（ ）。

```
char s[] = "china";
char *p; p = s;
```

A. *p 与 s[0] 值相等

B. s 数组长度和 p 所指向的字符串长度相等　　数组长度要算\0的，字符串长度不算

C. 数组 s 中的内容和指针变量 p 中的内容相等　　p的内容是地址

D. s 和 p 完全相同　　s是常量指针

49. 若有定义 char s[10]; 则下面表达式中不表示 s[1] 的地址的是（ ）。

A. s + 1　　B. s ++　　C. &s[0] + 1　　D. &s[1]

50. 设有程序段

```
#include <string.h>
int main()
{
    char *p1, *p2, str[50] = "abc";
    p1 = str;
    p2 = str;
    strcpy(str+1, strcat(p1, p2));
    printf("%s\n", str);
    return 0;
}
```

上面程序输出的结果是（ ）。

A. cabcabc　　B. aabcabc　　C. bcabcabc　　D. abcabcabc

51. 设有定义:char *cc[2] = {"1234", "5678"}; 则正确的叙述是（ ）。

A. cc 数组元素的值分别是 1234 和 5678

B. 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

C. 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 算上\0是5个字符

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

52. 若有说明：char *language[] = {"FORTRAN", "BASIC", "PASCAL", "JAVA", "C"}; 则 language[2] 的值是（ ）。

A. 一个地址　　B. 一个不定值　　C. 一个字符串　　D. 一个字符

二、程序题目

以下是数组这章练习题目

课后题：3、4、5、6、10、11

程序作业：第 11 次作业，第 14 次作业的第 2、3；第 15 次作业的第 3 题；第 19 次作业