1、以下不能正确定义二维数组的选项是\_\_\_\_\_\_。

正确答案: D 你的答案: B (错误)

A、int a[2][2]={{1},{2}};

B、int a[][2]={1,2,3,4};

C、int a[2][2]={{1},2,3};

D、int a[2][]={{1,2},{3,4}};

对于二维数组，行数可以省略，但列数一定要指定，因为编译器根据列数来进行寻址

2、若有定义int（\*pt）[3]；则下列说法正确的是：

正确答案: D 你的答案: B (错误)

A、定义了基类型为int的三个指针变量

B、定义了基类型为int的具有三个元素的指针数组pt

C、定义了一个名为\*pt、具有三个元素的整型数组

D、定义了一个名为pt的指针变量，它可以指向每行有三个整数元素的二维数组的某一行

int (\*pt)[3]，首先看括号内，\*pt说明pt是一个指针，其指向的内容是int[3]，具有3个int元素的数组。

D选项说，可以指向每行有三个整数元素的二维数组，即int[][3]。

int(\*pt)[3] = NULL;

int arr[2][3] = {0};

pt = arr;

3、以下函数用法正确的个数是：

void test1()

{

unsigned char array[MAX\_CHAR+1],i;

for(i=0;i<=MAX\_CHAR;i++){

array[i]=i;

}

}

char\*test2()

{

char p[] = "hello world";

return p;

}

char \*p =test2();

void test3(){

char str[10];

str++;

\*str='0';

}

正确答案: A 你的答案: C (错误)

A、0

B、1

C、2

D、3

第一个问题：

重点不在于CHAR\_MAX的取值是多少，而是在于i的取值范围是多少。

一般char的取值范围是-128到127，而u char 则是0~255，所以i的取值范围是0~255.

所以当CHAR\_MAX常量大于255时，执行i++后，i不能表示256以上的数字，所以导致无限循环。

第二个问题：

重点在于函数中p的身份，他是一个指针，还是数组名；

如果是指针p，则p指向存放字符串常量的地址，返回p则是返回字符串常量地址值，调用函数结束字符串常量不会消失（是常量）。所以返回常量的地址不会出错。

如果是数组p，则函数会将字符串常量的字符逐个复制到p数组里面，返回p则是返回数组p，但是调用函数结束后p被销毁，里面的元素不存在了。

例子中p是数组名，所以会出错，p所指的地址是随机值。

若是把char p[]="hello";改成char \*p="hello";就可以了。

第三个问题：

重点在于str++;这实际的语句就是str=str+1;而str是数组名，数组名是常量，所以不能给常量赋值。（可以执行str+1，但是不能str=.）

4、下面程序段的输出结果是

char \*p1 = ”123”, \*p2 = ”ABC”, str[50] = “xyz”;

strcpy(str + 2, strcat(p1, p2));

printr(“%s\n”, str);

正确答案: D 你的答案: C (错误)

A、xyz123ABC

B、z123ABC

C、xy123ABC

D、出错

原代码有错：p1和p2都指向常量字符串，在常量区，所以不能对其进行操作；改为数组即可，但是用字符串初始化数组时要记得将数组长度加1，因为字符串默认的末尾有一个‘\0’；第二点要注意的是，strcat函数的p1要有足够的空间来容纳p1和p2连接后的串长。

修改为以下代码将可以：

char p1[7] = "123";

char p2[] = "ABC";

char str[50] = "xyz";

strcpy(str + 2, strcat(p1, p2));

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

5、C++语言不能定义字符串变量。

正确答案: B 你的答案: A (错误)

A、是

B、否

6、下面C程序的输出结果:

int i=0,a=1,b=2,c=3;

i=++a||++b||++c;

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

正确答案: A 你的答案: B (错误)

A、1 2 2 3

B、1 2 3 4

C、3 2 3 4

D、3 3 3 4

6、下列 C 代码中，不属于未定义行为的有：\_\_\_\_\_\_。

正确答案: C 你的答案: B (错误)

A、int i=0;i=(i++);

B、char \*p=”hello”;p[1]=’E’

C、char \*p=”hello”;char ch=\*p++

D、int i=0;printf(“%d%d\n”,i++ i--)

未定义行为（Undefined Behavior）是指C语言标准未做规定的行为。同时，标准也从没要求编译器判断未定义行为，所以这些行为有编译器自行处理，在不同的编译器可能会产生不同的结果，又或者如果程序调用未定义的行为，可能会成功编译，甚至一开始运行时没有错误，只会在另一个系统上，甚至是在另一个日期运行失败。当一个未定义行为的实例发生时，正如语言标准所说，“什么事情都可能发生”，也许什么都没有发生。一句话，未定义行为就是运行结果不确定

1.变量即是左边结果，又是右边的操作数，如a+=a++，a %= b ^= a ^= b ^= a

2.使用越界数组也是C的一个“未定义行为”

3.允许一个随便指的指针的读写。

4.使用未初始化的变量

等等

7、下面关于虚函数的描述，错误的是

正确答案: B 你的答案: C (错误)

A、在成员函数声明的前面加上virtual修饰，就可把该函数声明为虚函数

B、基类中说明了虚函数后，派生类中对应的函数也必须说明为虚函数

C、虚函数可以是另一个类的友元函数，但不能是静态成员函数

D、基类中说明的纯虚函数在其任何需要实例化的派生类中都必须实现

一旦在基类中指定某成员函数为虚函数，不管在公有派生类中是否给出virtual声明，派生类对其重载定义的成员函数均为虚函数，为增强可读性，通常在派生类中也加入virtual关键字。

8、函数参数使用的空间是在（）中申请的，malloc或new是在（）中申请空间的？

正确答案: B 你的答案: A (错误)

A、堆，栈

B、栈，堆

C、栈， 栈

D、堆，堆

函数内局部变量，形参都是在栈内存空间中申请的，随着函数的结束而消失

malloc和new关键字在堆内存申请空间，使用结束需要认为的释放

栈 系统维护

堆 自己申请释放

传参是系统操作的，故栈

malloc和new是由自己申请的，故堆

9、下面的程序可以从0....n-1中随机等概率的输出m个不重复的数。这里我们假设n远大于m

knuth(int n, int m)

{

srand((unsigned int)time(0));

for (int i = 0; i < n; i++) {

if ( ) {

cout << i << endl;

( );

}

}

}

正确答案: B 你的答案: C (错误)

A、rand()%(n-i)<=m m--

B、rand()%(n-i)<m m--

C、rand()%(n-i)>=m m++

D、rand()%(n-i)>m m++

上面有的回答思路很正确，但概率表述有问题，在这跟大家分享下

由这个for循环循环n次，且在满足条件时才输出i,可知，输出m个不同值的要求已满足，因为每次输出的都是i值，而i值每次都是不一样的，m--保证了程序在输出了m个值后就停止循环。

在i=0时，rand()%(n-i)的取值范围为0到n-1，共n个数，此时要输出0只需要rand()%(n-i)小于m，故i=0被输出的概率为m/n;

在i=1时，rand()%(n-i)的取值范围为0到n-2，共n-1个数，若i=0没有被输出，则m--未被执行，此时i=1被输出的概率为m/(n-1)，若i=0已经被输出了，则m变为m-1，此时i=1被输出的概率为(m-1)/(n-1)；由概率论的知识，可知此时i=1被输出的概率为

P=(1-m/n)\*(m/(n-1))+m/n\*((m-1)/(n-1))=m/n；以此类推，可知每个数被输出的概率都为m/n

10、有如下程序段：

class A

{

int \_a;

public:

A(int a): \_a(a)

{

}

friend int f1(A &);

friend int f2(const A &);

friend int f3(A);

friend int f4(const A);

};

以下调用哪个是错误的：

正确答案: A 你的答案: D (错误)

A、f1(0)

B、f2(0)

C、f3(0)

D、f4(0)

解释：

非常量引用的初始值必须为左值

要理解这个先得理解左值和右值的概念

一个区分左值与右值的便捷方法是：看能不能对表达式取地址，如果能，则为左值，否则为右值。

本题举例：

执行f1(0),实参0要传成A对象，那么执行

A &a1 = 0; //这是不行的。

执行f2(0),实参0要传成A对象，那么执行

const A &a2 = 0;//这是可行的。

选A。

非常量引用必须是左值

左值和右值都是针对表达式而言的，左值是指表达式结束后依然存在的持久对象，右值是指表达式结束时就不再存在的临时对象。

在标准C++语言中，临时量（术语为右值，因其出现在赋值表达式的右边）可以被传给函数，但只能被接受为const &类型。

函数形式参数是临时量，是右值。所以只能被接受为const &类型 。故A错。

11、下列关于虚函数的说法正确的是（）

正确答案: C D 你的答案: A B C (错误)

A、在构造函数中调用类自己的虚函数，虚函数的动态绑定机制还会生效

B、在析构函数中调用类自己的虚函数，虚函数的动态绑定机制还会生效

C、静态函数不可以是虚函数

D、虚函数可以声明为inline

由于类的构造次序是由基类到派生类，所以在构造函数中调用虚函数，这个虚函数不会呈现出多态； 相反，类的析构是从派生类到基类，当调用继承层次中某一层次的类的析构函数时往往意味着其派生类部分已经析构掉，所以也不会呈现出多态

因为静态成员函数没有this，也就没有存放vptr的地方，同时其函数的指针存放也不同于一般的成员函数，其无法成为一个对象的虚函数的指针以实现由此带来的动态机制。静态是编译时期就必须确定的，虚函数是运行时期确定的。

关知识点：什么函数不能声明为虚函数？

一个类中将所有的成员函数都尽可能地设置为虚函数总是有益的。

设置虚函数须注意：

1：只有类的成员函数才能说明为虚函数；

2：静态成员函数不能是虚函数；

3：内联函数不能为虚函数；

4：构造函数不能是虚函数；

5：析构函数可以是虚函数，而且通常声明为虚函数。

虚函数可以加 inline 关键字，只是编译器并不会真的生成 inline 。

12、有如下程序段，请问k的值是

enum {

    a, b=5, c, d=4, e

} k;

k =c;

正确答案: D   你的答案: D (正确)

A、3

B、4

C、5

D、6

enum中：首元素不赋值的话，默认为0；后一个元素不赋值的话比前一个元素大1

13、如下代码的输出结果是（）

int _tmain(int argc, _ TCHAR• argv[l) 
int 2, 3,4, S, 8,9, 10, 
int •ptr 
cout « •(ptr-3) « endl; 

正确答案: C   你的答案: B (错误)

A、3

B、4

C、9

D、10

数组名是数组这种类型的变量名，所以对数组名取地址是取的整个数组的地址，所以&a+1自然要跨过整个数组的长度，本题即跨过2\*5 = 10个int的长度。

有了上述概念，再来分析这道题，int \*ptr = (int \*)(&a+1)，p此时指向的地址应该是a[1][4]后面的地址，由于ptr是int型指针，ptr-3应该是ptr向前移动3个元素，即ptr-3指向a[1][2]，所以\*(ptr -3) = 9

14、以下描述正确的是？

正确答案: C   你的答案: D (错误)

A、虚函数是可以内联的，可以减少函数调用的开销提高效率

B、类里面可以同时存在函数名和参数都一样的虚函数和静态函数

C、父类的析构函数是非虚的，但是子类的析构函数是虚的，delete子类对象指针会调用父类的析构函数

D、以上都不对

对于A的话参看 <http://blog.csdn.net/zy1691/article/details/3606128>

《Effective C++》的第33款:

即使是最简单的虚函数调用，编译器的内联处理程序对它也爱莫能助。（这一点也不奇怪。virtual的意思是"等到运行时再决定调用哪个函数"，inline的意思是"在编译期间将调用之处用被调函数来代替"，如果编译器甚至还不知道哪个函数将被调用，当然就不能责怪它拒绝生成内联调用了）。

B  会冲突，实例调用的时候不知道调用哪个了。。

C  A\* a = new B  //B是A的子类

    delete a

    此时父类析构函数需要加virtual，如果不加virtual，那么最后只会调用A的析构函    数，而派生的部分，并没有得到析构，导致内存泄露。

    B\* b = new B

    delete b

    此时父类析构函数并不需要加virtual，就是一个很正常的构造析构过程

    A() -> B() -> ~B() -> ~A()

15、以下程序段执行后结果是()

#include<stdio.h>

void main()

{

    short \*p,\*q;

    short arr[15]={0};

    p=q=arr;

    p++;

    printf("%d,",p-q);

    printf("%d,",(char\*)p-(char\*)q);

    printf("%d",sizeof(arr)/sizeof(\*arr));

}

正确答案: D   你的答案: C (错误)

A、1,0,15

B、0,2,1

C、1,1,15

D、1,2,15

指针自增、自减每次移动的偏移量是指针所指向对象的字节大小，所以p++与q的偏移量是2个字节。

指针相减的值是指针地址的偏移除以指针每次移位的大小；

1)p-q=1;偏移量为2个字节，每次移动2个字节，所以为1

2)(char \*)p-(char \*)q，指针的偏移没变，但是每次指针移位是按照（char\*）类型移动，即每次移动1个字节，所以是2

3）数字每次元素2个字节，所以sizeof(arr)为30，sizeof(\*arr)为2。

16、有这样一个类：

class Eye

{

    public:

    void Look(void);

};

现在希望定义一个Head类，也想实现Look的功能，应该使用()方法，实现代码重用。

正确答案: B   你的答案: A (错误)

A、继承

B、组合

C、模板

D、过滤

组合是在新类中以原有类的对象作为数据成员，继承是在不改变现有的类的基础上，采用现有类的形式并在其中添加新代码，组合一般用于在新类中使用现有类的功能而不是他的接口的情况，就是新类用户看到的只是为新类所定义的接口。而继承则是用于在新类需要向基类转化的情况（多态），这也是组合和继承使用的最清晰的判断方法。

       结合本题分析，我们只需让眼睛作为头的一个成员即可，而让头去继承眼睛的特性是没有必要的。

      因此这道题选择B

17、用多态是指

正确答案: D   你的答案: B (错误)

A、强制多态和包含多态

B、重载多态和强制多态

C、参数多态和重载多态

D、包含多态和参数多态

重载多态和强制多态是 指特定多态。

参数多态和包含多态是指通用多态。

18、针对以下代码，

const char str1[]=”abc”;

const char str2[]=”abc”;

const char \*p1 = “abc”;

const char \*p2 = “abc”;

判断下列说法哪个是正确的(注意是地址)：\_\_\_\_\_\_。

正确答案: A   你的答案: D (错误)

A、str1和str2地址不同，P1和P2地址相同。

B、str1和str2地址相同，P1和P2地址相同。

C、str1和str2地址不同，P1和P2地址不同。

D、str1和str2地址相同，P1和P2地址不同。

E、4个地址都相同

str1和str2毫无干系，他们都是字符常量数组，恰巧存储的值相同而已。"abc"和str1 str2都存储在常量区。str1 str2的地址就是str1[0] str2[0]的地址，即数组首元素所在的地址。

后面的两个“abc”存储在常量区，因为他们是字面值（常量），但是p1 p2存储在栈中。其中p1和p2的值不同，但是他们指向同一块静态存储区域。也就是"abc"的地址。

19、

#include

void test(void \*data) {

    unsigned int value = （此处应填入）

    printf("%u", value);

}

using namespace std;

int main() {

    unsigned int value = 10;

    test(&value);

    return 0;

}

正确答案: D   你的答案: C (错误)

A、\*data

B、(unsigned int)(\*data)

C、(unsigned\*)data

D、\*((unsigned int \*)data)

void test(void \*data)

 注意，参数类型是void,  所以先要进行指针转换：（unsigned int \*)然后再取值。

20、下面程序的输出结果是

char \*p1= “123”, \*p2 = “ABC”, str［50］= "xyz";

strcpy（str+2,strcat（p1,p2））;

cout << str;

正确答案: D   你的答案: C (错误)

A、xyz123ABC

B、z123ABC

C、xy123ABC

D、出错

char\* strcat(char \*,const char\*)//第一个参数所指向的内容必须可以修改，可以赋值为在栈上分配的数组

strcat（p1,p2）试图修改p1的内容，p1指向文字常量区，其指向的内容无法修改

21、

下面四个类A,B,C,D,在32位机器上sizeof(A),sizeof(B),sizeof(C),sizeof(D)值分别为()

class A{

};

class B{

char ch;

int x;

};

class C{

public:

void Print(void){}

};

class D

{

public:

virtual void Print(void){}

};

正确答案: C 你的答案: B (错误)

A、0,5,0,0

B、1,8,4,4

C、1,8,1,4

D、1,5,1,4

答案是C

类A空类型的实例虽然不包含任何信息,但是必须在内存中占一定的空间,否则无法使用这些实例,一般都是1

类B因为内存对齐所以为8,

类C里面虽然有函数,但是只需要知道函数的地址即可,而这些函数的地址只与类型相关,而

与类型的实例无关,编译器不会因为函数而在内存中多添加任何的额外信息.所以还是1

类D因为有虚函数,C++的编译器一旦发现一个类型中有虚函数,就会为该类型生成虚函数表,并在该类型的

每一个实例中添加一个指向虚函数表的指针.因为多了一个指针,所以在32位机器为4,64位机器为8

22、两个圆，半径分别为1cm,3cm,小圆在大圆外，绕大圆圆周一周，请问小圆一个旋转了多少圈？

正确答案: A 你的答案: D (错误)

A、4

B、1

C、2

D、3

        把大圆剪断拉直.小圆绕大圆圆周一周,就变成从直线的一头滚至另一头.因为直线长就是大圆的周长,是小圆周长的3倍,所以小圆要滚动3圈.

但是现在小圆不是沿直线而是沿大圆滚动,小圆因此还同时作自转,当小圆沿大圆滚动1周回到原出发点时,小圆同时自转1周.当小圆在大圆内部滚动时自转的方向与滚动的转向相反,所以小圆自身转了2周.当小圆在大圆外部滚动时自转的方向与滚动的转向相同,所以小圆自身转了4周.

       如何更通俗的理解呢，就是看圆心距离，在圆外时圆心距离是4，且小球滚动圆心距离是不变的，然后路径是4πR，又可知，圆心走过的距离和小圆边走过的距离一定是相等的，故也是4πR，所以要转的圈数是4πR/ πR=4圈，同理可知如果小圆在圆内只要2圈就够了

*来自 <*[*https://www.nowcoder.com/test/question/done?tid=10238876&qid=26002#summary*](https://www.nowcoder.com/test/question/done?tid=10238876&qid=26002#summary)*>*

23、一块不均匀的主板，从头烧到尾要2个小时，最少需要几块相同款型的主板，才能准确的衡量出2小时30分钟？

正确答案: B   你的答案: B (正确)

A、2根

B、3根

C、4根

D、5根

三根就可以。 首先拿一根从两头烧。是一个小时。 再拿一根从两头烧，又是一个小时。 第三根和第二根同时烧，不过第三根是从一头烧，等第二根烧完了第三根就烧了一个小时，还剩一个小时。然后将第三根剩余的从两头烧，就是半个小时了。 所以总共就是3根了。

24、有一个小白程序员，写了一个只能对5个数字进行排序的函数。现在有25个不重复的数字，请问小白同学最少调几次该函数，可以找出其中最大的三个数？

正确答案: C 你的答案: B (错误)

A、5

B、6

C、7

D、8

先分成五组

测试五次  得到每组前三

五组里面的 第一（假设是 A  B  C  D  E） **测一次**    得到序列 （假设比较好的序列是  A  B  C  D  E）

再从得到最大那一组(A组)的里面取出第二大  X  和第三大 Y ，和第二大数那一组(B组) 取出第二大 Z，把最大五个排好序(A  B   C   D  E)里面的 最大（A）  和倒数两个（D   E）换掉

现在就是（B  Z  C  X   Y）**调用一次**

5+1+1=7

25、关于ARP表，以下描述中正确的是（）

正确答案: C 你的答案: C (正确)

A、用于在各个子网之间进行路由选择

B、提供常用目标地址的快捷方式来减少网络流量

C、用于建立IP地址到MAC地址的映射

D、用于进行应用层信息的转换

ARP是地址解析协议，工作在网络层，简单语言解释一下工作原理。

1：首先，每个主机都会在自己的ARP缓冲区中建立一个ARP列表，以表示IP地址和MAC地址之间的对应关系。

2：当源主机要发送数据时，首先检查ARP列表中是否有对应IP地址的目的主机的MAC地址，如果有，则直接发送数据，如果没有，就向本网段的所有主机发送ARP数据包，该数据包包括的内容有：源主机 IP地址，源主机MAC地址，目的主机的IP 地址。

3：当本网络的所有主机收到该ARP数据包时，首先检查数据包中的IP地址是否是自己的IP地址，如果不是，则忽略该数据包，如果是，则首先从数据包中取出源主机的IP和MAC地址写入到ARP列表中，如果已经存在，则覆盖，然后将自己的MAC地址写入ARP响应包中，告诉源主机自己是它想要找的MAC地址。

4：源主机收到ARP响应包后。将目的主机的IP和MAC地址写入ARP列表，并利用此信息发送数据。如果源主机一直没有收到ARP响应数据包，表示ARP查询失败。

广播发送ARP请求，单播发送ARP响应。

所以，答案应该选择C选项。

26、边长1米的正方体共计2100个，堆成一个实心的长方体，它的高是10米，长，宽都大于高，问长方体的长与宽的和是多少米？

正确答案: D 你的答案: B (错误)

A、41

B、31

C、37

D、29

答案是D

高是10米,则长乘以宽=2100/10=210

将210分解质因数,得到210=2\*3\*5\*7

到了这里,因为长和宽都大于10 ,所以只能取2\*7和3\*5

所以长为15,宽为14

27、一条tcp连接，主动关闭的一方不可能出现的连接状态是()

正确答案: A 你的答案: B (错误)

A、CLOSE\_WAIT

B、FIN\_WAIT2

C、TIME\_WAIT

D、FIN\_WAIT1

介绍一下TCP连接建立与关闭过程中的状态。TCP连接过程是状态的转换，促使状态发生转换的因素包括用户调用、特定数据包以及超时等，具体状态如下所示：

CLOSED ：初始状态，表示没有任何连接。

LISTEN ： Server 端的某个 Socket 正在监听来自远方的 TCP 端口的连接请求。

SYN\_SENT ：发送连接请求后等待确认信息。当客户端 Socket 进行 Connect 连接时，会首先发送 SYN 包，随即进入 SYN\_SENT 状态，然后等待 Server 端发送三次握手中的第 2 个包。

SYN\_RECEIVED ：收到一个连接请求后回送确认信息和对等的连接请求，然后等待确认信息。通常是建立 TCP 连接的三次握手过程中的一个中间状态，表示 Server 端的 Socket 接收到来自 Client 的 SYN 包，并作出回应。 ESTABLISHED ：表示连接已经建立，可以进行数据传输。

FIN\_WAIT\_1 ：主动关闭连接的一方等待对方返回 ACK 包。若 Socket 在 ESTABLISHED 状态下主动关闭连接并向对方发送 FIN 包（表示己方不再有数据需要发送），则进入 FIN\_WAIT\_1 状态，等待对方返回 ACK 包，此后还能读取数据，但不能发送数据。在正常情况下，无论对方处于何种状态，都应该马上返回 ACK 包，所以 FIN\_WAIT\_1 状态一般很难见到。

FIN\_WAIT\_2 ：主动关闭连接的一方收到对方返回的 ACK 包后，等待对方发送 FIN 包。处于 FIN\_WAIT\_1 状态下的 Socket 收到了对方返回的 ACK 包后，便进入 FIN\_WAIT\_2 状态。由于 FIN\_WAIT\_2 状态下的 Socket 需要等待对方发送的 FIN 包，所有常常可以看到。若在 FIN\_WAIT\_1 状态下收到对方发送的同时带有 FIN 和 ACK 的包时，则直接进入 TIME\_WAIT 状态，无须经过 FIN\_WAIT\_2 状态。

TIME\_WAIT ：主动关闭连接的一方收到对方发送的 FIN 包后返回 ACK 包（表示对方也不再有数据需要发送，此后不能再读取或发送数据），然后等待足够长的时间（ 2MSL ）以确保对方接收到 ACK 包（考虑到丢失 ACK 包的可能和迷路重复数据包的影响），最后回到 CLOSED 状态，释放网络资源。

CLOSE\_WAIT ：表示被动关闭连接的一方在等待关闭连接。当收到对方发送的 FIN 包后（表示对方不再有数据需要发送），相应的返回 ACK 包，然后进入 CLOSE\_WAIT 状态。在该状态下，若己方还有数据未发送，则可以继续向对方进行发送，但不能再读取数据，直到数据发送完毕。

LAST\_ACK ：被动关闭连接的一方在 CLOSE\_WAIT 状态下完成数据的发送后便可向对方发送 FIN 包（表示己方不再有数据需要发送），然后等待对方返回 ACK 包。收到 ACK 包后便回到 CLOSED 状态，释放网络资源。

CLOSING ：比较罕见的例外状态。正常情况下，发送 FIN 包后应该先收到（或同时收到）对方的 ACK 包，再收到对方的 FIN 包，而 CLOSING 状态表示发送 FIN 包后并没有收到对方的 ACK 包，却已收到了对方的 FIN 包。有两种情况可能导致这种状态：其一，如果双方几乎在同时关闭连接，那么就可能出现双方同时发送 FIN 包的情况；其二，如果 ACK 包丢失而对方的 FIN 包很快发出，也会出现 FIN 先于 ACK 到达。

Client 
send SYN 
SYN SENT 
receive SYN+ACK 
ESTABLISHED 
receive FIN 
FIN WAITI 
receive ACK 
FIN WAIT2 
receive FIN 
TIME WAIT 
send ACK 
TimeOut 
CLOSED 
SYN 
SYN+ACK 
ACK 
FIN 
ACK 
FIN 
ACK 
Server 
LISTEN 
receive SYN 
SYN RECV 
SYN + ACK sent 
receive ACK 
ESTABLISHED 
receive FIN 
CLOSE WAIT 
receive ACK 
receive FIN 
LAST_ACK 
receive ACK 
CLOSED 
http : / 'hi. bal du. com/imlidapeng 

28、若一序列进栈顺序为e1,e2,e3,e4,e5,问存在多少种可能的出栈序列（）

正确答案: B 你的答案: B (正确)

A、41

B、42

C、43

D、44

卡特兰数，相关问题还有：已知前序遍历的顺序是xxxx求这棵树有多少种形状

记住公式： C(2n,n)/(n+1)

29、下列 main （）函数执行后的结果是（）

• fun(mt value) 
cnt O; 
while (value) 
cnt+ 
value = value & (value - 
rn cnC 
main (void) 
void 
printfC%d•. fun(65535)): 
1); 

正确答案: C 你的答案: C (正确)

A、17

B、15

C、16

D、1

就是统计65535的二进制有多少个1，这是一种比较好的解法。

30、当参数\*x=1, \*y=1, \*z=1时，下列不可能是函数add的返回值的( )?

int add(int \*x, int \*y, int \*z){

\*x += \*x;

\*y += \*x;

\*z += \*y;

return \*z;

}

正确答案: D 你的答案: D (正确)

A、4

B、5

C、6

D、7

同 二 一 指 
匚 不 
口 同 同 同 自 
N N 
N 匕 应 匕 应 匕 日 N 
一 匚 匚 匚 向 
日 向 一 回 问 耆 

31、设变量已正确定义，以下不能统计出一行中输入字符个数（不包含回车符）的程序段是

正确答案: D   你的答案: C (错误)

A、n=0;while(ch=getchar()!='\n')n++;

B、n=0;while(getchar()!='\n')n++;

C、for(n=0;getchar()!='\n';n++);

D、n=0;for(ch=getchar();ch!='\n';n++);

对于for循环，其初始条件只执行一次，因此ch只从输入流中取一个字符，之后就再不会取字符，因此会死循环

32、下面说法错误的是()

正确答案: A B 你的答案: B D (错误)

A、在组合时,为了保证成员对象被正确清除,在组合类的析构函数中需要显式调用其成员对象的析构函数

B、在类的继承层次中,可以自动进行向上和向下类型转换.而且都是安全的

C、构造函数可以重载,析构函数不能重载

D、C++的派生类如果要覆盖一个继承到的成员函数,在基类中需要将该函数声明为virtual

A选项：在组合类的析构函数中并不需要显式调用其成员对象的析构函数，因为当执行组合类的析构函数时，其数据成员也会被销毁，对于类成员对象来说，成员对象销毁时，程序会自动调用其析构函数；不过对于组合类中new的指向类成员的指针，组合类需要自己去delete该指针；

对于D选项，为什么是错的？因为如果派生类的函数与基类的函数同名，并且参数也相同，但是基类函数没有virtual关键字，此时，基类的函数被隐藏，注意，是隐藏而不是覆盖。覆盖的基本特征之一就是 基类函数必须有virtual关键字。

33、若有以下程序

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | #include <stdio.h>  #define N 2  #define M N+ 1  #define NUM ( M+1 )\*M/2  main()  {      printf( "%d\n" ,NUM );  } |

程序运行后的输出结果是?

正确答案: B   你的答案: D (错误)

A、4

B、8

C、9

D、6

本题考查宏定义,宏定义只是做个简单的替换,执行NUM=(N+1+1)\*N+1/2=8,选项B正确

34、以下程序输出结果是\_\_\_\_

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | **class** A  {  **public**:      A ():m\_iVal(0){test();}  **virtual void** func() { std::cout<<m\_iVal<<‘ ’;}  **void** test(){func();}  **public**:  **int** m\_iVal;  };  **class** B : **public** A  {  **public**:      B(){test();}  **virtual void** func()      {        ++m\_iVal;        std::cout<<m\_iVal<<‘ ’;        }  };  **int** main(**int** argc ,**char**\* argv[])  {    A\*p = **new** B;    p->test();  **return** 0;  } |

正确答案: C   你的答案: C (正确)

A、1 0

B、0 1

C、0 1 2

D、2 1 0

E、不可预期

F、以上都不对

本问题涉及到两个方面：

1.C++继承体系中构造函数的调用顺序。

2.构造函数中调用虚函数问题。

C++继承体系中，初始化时构造函数的调用顺序如下

    (1)任何虚拟基类的构造函数按照他们被继承的顺序构造

(2)任何非虚拟基类的构造函数按照他们被继承的顺序构造

(3)任何成员对象的函数按照他们声明的顺序构造

(4)类自己的构造函数

据此可知 A\*p = newB;先调用A类的构造函数再调用B类的构造函数。

构造函数中调用虚函数,虚函数表现为该类中虚函数的行为，即在父类构造函数中调用虚函数，虚函数的表现就是父类定义的函数的表现。why？原因如下：

假设构造函数中调用虚函数，表现为普通的虚函数调用行为，即虚函数会表现为相应的子类函数行为，并且假设子类存在一个成员变量int a；子类定义的虚函数的新的行为会操作a变量，在子类初始化时根据构造函数调用顺序会首先调用父类构造函数，那么虚函数回去操作a，而因为a是子类成员变量，这时a尚未初始化，这是一种危险的行为，作为一种明智的选择应该禁止这种行为。所以虚函数会被解释到基类而不是子类。

据此可以得到答案C正确

简短版本：在《C++ Primer》556页中讨论了构造函数和析构函数中调用虚函数的效应，得出的结果是：在构造或析构函数中调用虚函数会执行与之所属类型相对应的虚函数版本。

35、使用 char\* p = new char[100]申请一段内存，然后使用delete p释放，有什么问题？

正确答案: B 你的答案: A (错误)

A、会有内存泄露

B、不会有内存泄露，但不建议用

C、编译就会报错，必须使用delete []p;

D、编译没问题，运行会直接崩溃

当用delete来释放用new int[]申请的内存空间时，由于其为基本数据类型没有析构函数，所以使用delete与delete []相同，两者都会释放申请的内存空间，若是自定义的数据类型，有析构函数时，用new []申请的空间，必须要用delete []来释放，因为要delete []时会逐一调用对象数组的析构函数，然后释放空间

36、下列哪两个是等同的

int b;

1.const int \*a = &b;

2.const \* int a = &b;

3.const int\* const a = &b;

4.int const\* const a = &b;

正确答案: C 你的答案: A D (错误)

A、1，4

B、1，2

C、3，4

D、2，3

第一个 含义是 a 是指针 指向 int，这个int 是静态的 const 修饰的是int

第二个 语法错误

第三个 a 是静态的指针（第二个const 修饰)，指向 int ，这个int是静态的（第一个const 修饰）

第四个 a 是静态的指针（第二个const 修饰)，指向一个静态的变量（第一个const 修饰， 这个变量是int的）

故3、4相等。选C

37、进行数据库提交操作时使用事务（Transaction）是为了?

正确答案: B   你的答案: B (正确)

A、提高效率

B、保证数据一致性

C、网络安全

D、归档数据文件

数据库事务(Database Transaction) ，是指作为单个逻辑工作单元执行的一系列操作。 事务处理可以确保除非事务性单元内的所有操作都成功完成，否则不会永久更新面向数据的资源。通过将一组相关操作组合为一个要么全部成功要么全部失败的单元，可以简化错误恢复并使应用程序更加可靠。一个逻辑工作单元要成为事务，必须满足所谓的ACID(原子性、一致性、隔离性和持久性)属性。

38、假设一段公路上，1小时内有汽车经过的概率为96%，那么，30分钟内有汽车经过的概率为?

正确答案: C   你的答案: C (正确)

A、48%

B、52%

C、80%

D、96%

1个小时内有车通过的概率是0.96，也就是说这个一个小时内没有车通过的概率是0.04。

题目，故意给出1个小时的单位来迷惑我们。其实，我们可以把单位分解为30分钟。

我是这么分解的：如果一个小时都没出现任何车辆，就相当于在连续的两个30分钟里面都没有出现任何车辆，

这里每半个小时是否有车出现绝对是独立，那么我们可以设30分钟内没有车出现的概率为p，可以列出如下公式：

x² ＝ 0.04 解得 x＝0.2 就表示30分钟内出现车辆和不出现车辆是对立事件，那么出现车辆的概率就是0.8.

39、在网络应用测试中，网络延迟是一个重要指标。以下关于网络延迟的理解，正确的是？

正确答案: D   你的答案: B (错误)

A、指响应时间

B、指报文从客户端发出到客户端接收到服务器响应的间隔时间

C、指报文在网络上的传输时间

D、指从报文开始进入网络到它开始离开网络之间的时间

B选项应该是往返时延。RTT(Round-Trip Time): 往返时延：在计算机网络中它是一个重要的性能指标，表示从发送端发送数据开始，到发送端收到来自接收端的确认（接收端收到数据后便立即发送确认），总共经历的时延。

网络延迟是指各式各样的数据在 网络介质 中通过 网络协议 (如TCP/IP)进行传输， 在 传输介质 中传输所用的时间，即从 报文 开始进入网络到它开始离开网络之间的时间。

40、已知一个线性表（38，25，74，63，52，48），假定采用散列函数h（key) = key%7 计算散列地址，并散列存储在散列表A【0....6】中，若采用线性探测方法解决冲突，则在该散列表上进行等概率成功查找的平均查找长度为

正确答案: C   你的答案: B (错误)

A、1.5

B、1.7

C、2.0

D、2.3

**答案C：**

**平均查找长度=总的查找次数/元素数**

**总的查找次数：** 38%7=3 （第1次出现3，无冲突，放在位置3，**查找次数为1**）

25%7=4（第1次出现4，无冲突，放在位置4，**查找次数为1**）

74%7=4（第2次出现4，有冲突，放在位置5，**查找次数为2**）

63%7=0（第1次出现0，无冲突，放在位置0，**查找次数为1**）

52%7=3（第2次出现3，有冲突，发现冲突3,4,5，故只能放到6，**查找次数为4**）

48%7=6 （第1次出现6，有冲突，发现冲突6,0，故只能放到1，**查找次数为3**）

**1+1+2+1+4+3=12**

**元素数=6**

**所以：平均查找长度=12/6=2**

41、由多个源文件组成的C程序，经过编辑、预处理、编译，链接等阶段会生成最终的可执行程序。下面哪个阶段可以发现被调用的函数未定义？

正确答案: C   你的答案: B (错误)

A、预处理

B、编译

C、链接

D、执行

**A：** **预处理**是 C 语言程序从源代码变成可执行程序的第一步，主要是 C 语言编译器**对各种预处理命令进行处理**，**包括头文件的包含、宏定义的扩展、条件编译的选择等**。

**B：** 编译之前，C 语言编译器会进行词法分析、语法分析 (-fsyntax-only) ，接着会把源代码翻译成中间语言，即汇编语言 。 **编译程序工作时，先分析，后综合，从而得到目标程序。所谓分析，是指词法分析和语法分析；所谓综合是指代码优化，存储分配和代码生成。** 值得一提的是，大多数的编译程序直接产生机器语言的目标代码，形成可执行的目标文件，但也有的编译程序则先产生汇编语言一级的符号代码文件，然后再调用汇编程序进行翻译加工处理，最后产生可执行的机器语言目标文件。

**C：** **链接**是处理可重定位文件，**把它们的各种符号引用和符号定义转换为可执行文件中的合适信息**( 一般是虚拟内存地址 ) 的过程。

编译时只是把函数的符号地址记录下来，链接时该函数符号有定义才会变成具体的地址。如果链接过程中所有符号都有定义，链接成功，生成可执行文件；否则链接失败。

42、某次买可乐集瓶盖活动中有5种不同的瓶盖以等概率出现，每买一瓶汽水可得到一个瓶盖，集齐所有瓶盖所买汽水瓶数的期望，与以下哪个结果最为接近？

正确答案: B   你的答案: D (错误)

A、9

B、11

C、13

D、15

选11。 取到一种不同瓶盖的期望次数为1； 在已经取到一种瓶盖的情况下，再取到一种不同的瓶盖的期望次数是1/（4/5）=5/4； 在已经取到两种瓶盖的情况下，再取到一种不同的瓶盖的期望次数是1/（3/5）=5/3； 。。。 因此，取到五种瓶盖的期望次数为1+5/4+5/3+5/2+5/1=11+5/12。

43、JAVA语言的下面几种数组复制方法中，哪个效率最高？

正确答案: B   你的答案: C (错误)

A、for循环逐一复制

B、System.arraycopy

D、Arrays.copyof

D、使用clone方法

 A、for循环的话，很灵活，但是代码不够简洁.

 B、System.arraycopy()源码。可以看到是native方法：native关键字说明其修饰的方法是一**个原生态方法**，方法对应的实现不是在当前文件，而是在用其他语言（**如C和C++**）实现的文件中。 可以将native方法比作Java程序同Ｃ程序的接口。

public static native void arraycopy(Object src, int srcPos,  
 Object dest, int destPos,int length);

C、选项有误，copyOf不是System的方法，而是Arrays的方法，下面是源码，可以**看到本质上是调用的arraycopy方法。**，那么其效率必然是比不上 arraycopy的

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **public static int**[] copyOf(**int**[] original, **int** newLength) {  **int**[] copy = **new int**[newLength];     System.arraycopy(original, 0, copy, 0,     Math.min(original.length, newLength));  **return** copy;  } |

 D。clone的话，**返回的是Object【】，需要强制转换**。 一般用clone效率是最差的，

44、一个不透明的箱子里共有红，黄，蓝，绿，白五种颜色的小球，各个小球的数量非常多而且接近相等，每种颜色的小球大小相同，质量相等，每个人从篮子里抽出两个小球，请问至少需要多少个人抽，才能保证有两个人抽到的小球颜色全部相同？

正确答案: D   你的答案: C (错误)

A、6

B、11

C、13

D、16

E、21

F、26

应该是16吧

一共可以

红红 红黄 之类的 有15种方案

但是到第16种无论如何都会和之前的重复

45、在以下协议中，那个协议与其他协议是不属于同一类的()

正确答案: B   你的答案: E (错误)

A、FTP

B、ICMP

C、TELNET

D、SMTP

E、DNS

**网络层**：IP协议、ICMP协议、ARP协议、RARP协议。

**传输层**：UDP协议、TCP协议。

**应用层**：FTP（文件传送协议）、Telenet（远程登录协议）、DNS（域名解析协议）、SMTP（邮件传送协议），POP3协议（邮局协议），HTTP协议， SNMP协议， TFTP。

47、在C++STL中常用的容器和类型，下面哪些支持下标"[]"运算？

正确答案: A C D F I   你的答案: A I (错误)

A、vector

B、list

C、deque

D、map

E、set

F、unordered\_map

G、unordered\_set

H、stack

I、string

vector:随机访问迭代器,复杂度O（1）

deque：同上，O（1）

map：双向迭代器，不过由于是关联容器，需要通过key访问Value的方法，O（h），h为树的高度

unordered\_map：前向迭代器，同上，平摊复杂度O（1），最差O（n），也与散列函数的好坏有关。

string：同vector

48、在OSI参考模型中，物理层的作用是？

正确答案: B   你的答案: C (错误)

A、建立和释放连接

B、透明地传输比特流

C、在物理实体之间传送数据帧

D、发送和接受用户数据

**物理层：**通过媒介传输比特,确定机械及电气规范（比特Bit）

**数据链路层**：将比特组装成帧和点到点的传递（帧Frame）

**网络层**：负责数据包从源到宿的传递和网际互连（包PackeT）

**传输层**：提供端到端的可靠报文传递和错误恢复（段Segment）

**会话层**：建立、管理和终止会话（会话协议数据单元SPDU）

**表示层**：对数据进行翻译、加密和压缩（表示协议数据单元PPDU）

**应用层**：允许访问OSI环境的手段（应用协议数据单元APDU）

49、给出以下定义, 则对两个数组的strlen正确的叙述为

|  |  |
| --- | --- |
| 1  2 | **char** x[]="abcdefg";  **char** y[]={'a','b','c','d','e','f','g'}; |

正确答案: D   你的答案: B (错误)

A、数组X和数组Y等价

B、数组X和数组Y长度相同

C、数组X的长度大于等于数组Y的长度

D、数组X的长度小于等于数组Y的长度

这个题有歧义

如果判断数组的长度肯定用strlen()函数，而这个函数以'\0'为结尾判断符，char x[]="abcdefg";定义x数组默认追加结尾符'\0'，char y[]={'a','b','c','d','e','f','g'};没有结尾符，strlen(y)的结果是遇到第一个'\0'，是一个未定值，因此答案是数组x的长度小于等于y的长度

如果判断数组的大小用sizeof()函数，sizeof()计入数组中包括'\0'在内的所有元素，因此数组x的大小大于数组y的大小，是大小而不是长度

50、以下程序的输出结果为

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | #include "stdio.h"  **int** func(**int** x, **int** y)  {  **return** (x + y);  }  **int** main()  {  **int** a = 1, b = 2, c = 3, d = 4, e = 5;      printf(" %d\n", func((a + b, b + c, c + a), (d, e)));    **return** 0;  } |

正确答案: C   你的答案: D (错误)

A、15

B、5

C、9

D、出错

主要知识点是C++中的逗号操作符！ C++ Primer第4版中145页关于逗号操作符论述如下：逗号表达式的结果是其最右边表达式的值

51、有一个类A，其数据成员如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | **class** A {  ...  **private**:  **int** a;  **public**:  **const int** b;  **float**\* &c;  **static const char**\* d;  **static double**\* e;  }; |

则构造函数中，成员变量一定要通过初始化列表来初始化的是：\_\_\_\_\_\_。

正确答案: B   你的答案: 空 (错误)

A、a b c

B、b c

C、b c d e

D、b c d

E、b

F、c

构造函数初始化时必须采用初始化列表一共有三种情况，

1.需要初始化的数据成员是对象(继承时调用基类构造函数)

2.需要初始化const修饰的类成员

3.需要初始化引用成员数据

因为static属于类并不属于具体的对象，所以 static成员是不允许在类内初始化的，那么static const 成员是不是在初始化列表中呢？

答案是NO

一是static属于类，它在未实例化的时候就已经存在了，而构造函数的初始化列表，只有在实例化的时候才执行。

二是static成员不属于对象。我们在调用构造函数自然是创建对象，一个跟对象没直接关系的成员要它做什么呢

52、面不是面向对象的基本原则的是？

正确答案: C   你的答案: D (错误)

A、单一职责原则（Single-Resposibility Principle）

B、开放封闭原则（Open-Closed principle）

C、抽象类原则（Abstract-Class principle）

D、依赖倒置原则（Dependecy-Inversion Principle）

E、接口隔离原则（Interface-Segregation Principle）

面向对象的五大基本原则：

单一职责原则（SRP）

开放封闭原则（OCP）

里氏替换原则（LSP）

依赖倒置原则（DIP）

接口隔离原则（ISP）

53、以下叙述中不正确的是（）。

正确答案: B   你的答案: D (错误)

A、一个好的程序应该有详尽的注释

B、在 C 程序中，赋值运算符的优先级最低

C、在 C 程序中，j++;是一条赋值语句

D、C程序中的#include和#define均不是C语句

D是对的，用分号;结尾的才看成是C语句  
#include和#define并不参加编译，只是由预处理器来处理  
其中，#include <file> 由预处理器负责将file引入的内容引入当前文件中，只是做简单的展开。  
#define A B 只是将文件中出现A的地方简单的用B来替换。

54、以下对C语言的”指针“描述不正确的是：

正确答案: D   你的答案: C (错误)

A、32位系统下任何类型指针的长度都是4个字节

B、指针的数据类型声明的是指针实际指向内容的数据类型

C、野指针是指向未分配或者已释放的内存地址

D、当使用free释放掉一个指针内容后，指针变量的值被置为NULL

free指的是释放该指针所指向的内存资源，指针值没有变化，为了防止指针成为野指针，需要在free后将指针置为NULL。

55、对于纯虚函数，下列表述正确的是？（    ）

正确答案: B   你的答案: A (错误)

A、派生类必须实现基类的纯虚函数

B、纯虚函数的声明总是以“=0”结束

C、纯虚函数是给出实现版本（即无函数体定义）的虚函数

D、含有纯虚函数的类一定是派生类

如果基类有纯虚函数virtual void fun()=0;，派生类没有实现这个函数，对于这个派生类，会自动添加一个virtual void fun{}=0;即默认派生类的这个纯虚函数没有实现，也为纯虚函数。此时这个派生类也是纯虚类不能实例化。

56、下列 main （）函数执行后的结果是（）

• fun(mt value) 
cnt O; 
while (value) 
cnt+ 
value = value & (value - 
rn cnC 
main (void) 
void 
printfC%d•. fun(65535)): 
1); 

正确答案: C   你的答案: C (正确)

A、17

B、15

C、16

D、1

要点：

x|(x+1)统计x二进制数中，0的个数

x&（x-1）统计x二进制数中，1的个数

57、在下面的类定义中，横线上应填入的内容是（）

class 
(public: 
void 
setData(doubLe 
static int count: 
p rivate: 
double data;); 

正确答案: B   你的答案: A (错误)

A、static int Fred：：

B、int Fred：：

C、int

D、static int

静态变量在类外初始化要加上：：，同时不能再加static关键字了

58、以下意图用来描述FACTORY METHOD(工厂方法)的是（）

正确答案: C   你的答案: B (错误)

A、表示作用于某对象结构中的各元素的操作，它使你可以在不改变各元素的类的前提下定义作用于这些元素的新操作

B、提供一个创建一系列相关或相互依赖对象的接口，而无需指定他们具体的类

C、定义一个用于创建对象的接口，让子类决定实例化哪一个类，该模式使一个类的实例化延迟到其子类

D、定义一系列的算法，把它们一个个封装起来，并且使它们可相互替换，本模式使得算法可独立于使用它的客户而变化

A: Visitor访问者模式

B: 抽象工厂模式

C: 工厂方法模式

D: 策略模式

59、大整数845678992357836701转话为16进制的表示，最后两位字符是（）

正确答案: B   你的答案: B (正确)

A、AB

B、9D

C、8B

D、EF

大数÷2=\*\*\*\*\*\*3350 …… 1

3350÷2=1675……0

1675÷2=837……1

837÷2=418……1

转换成二进制的话，最后4位就是1101，转换成16进制，最后一位就是D，选B

60、一个局域网中某台主机的ip地址为176.68.160.12，使用22位作为网络地址，最多可以连接的主机数为（）

正确答案: D   你的答案: A (错误)

A、254

B、1024

C、512

D、1022

22位网络地址，按二进制计算：11111111.11111111.11111100.00000000，十进制：255.255.252.0，主机位有10位（10个0），因此主机数为2的10次方减去2(网络地址[IP](https://www.baidu.com/s?wd=IP&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YknjI-mhRLuWFbrj7-rycs0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1n3PHDsnjck)和广播[IP](https://www.baidu.com/s?wd=IP&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YknjI-mhRLuWFbrj7-rycs0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1n3PHDsnjck)）：2^10-2=1022，最多可以连接1022台主机，若再出去网关[IP](https://www.baidu.com/s?wd=IP&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YknjI-mhRLuWFbrj7-rycs0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1n3PHDsnjck)，则最大可以接1021台主机。

61、重新启动Linux系统的同时把内存中的信息写入硬盘，应使用（）命令实现

正确答案: A   你的答案: A (正确)

A、#shutdown -r now

B、#halt

C、#reboot

D、#init3

在linux命令中reboot是重新启动，**shutdown -r now是立即停止然后重新启动，都说他们两个是一样的，其实是有一定的区别的**。

**shutdown**命令可以**安全地关闭**或重启Linux系统，它在系统关闭之前给系统上的所有登录用户提示一条警告信息。该命令还允许用户指定一个时间参数，可以是一个精确的时间，也可以是从现在开始的一个时间段。

精确时间的格式是hh:mm，表示小时和分钟，时间段由+ 和分钟数表示。系统执行该命令后会自动进行数据同步的工作。

该命令的一般格式: shutdown [选项] [时间] [警告信息]

命令中各选项的含义为:

- k 并不真正关机而只是发出警告信息给所有用户

- r 关机后立即重新启动

- h 关机后不重新启动

- f 快速关机重启动时跳过fsck

- n 快速关机不经过init 程序

- c 取消一个已经运行的shutdown

需要特别说明的是该命令只能由超级用户使用。

例1，系统在十分钟后关机并且马上重新启动: # shutdown –r +10

例2，系统马上关机并且不重新启动:# shutdown –h now

**halt**是最简单的关机命令，其**实际上是调用shutdown -h命令**。halt执行时，杀死应用进程，文件系统写操作完成后就会停止内核。

halt命令的部分参数如下：

[-f] 没有调用shutdown而强制关机或重启

[-i] 关机或重新启动之前，关掉所有的网络接口

[-p] 关机时调用poweroff，此选项为缺省选项

**reboot**的工作过程与halt类似，其作用是重新启动，而halt是关机。其参数也与halt类似。reboot命令重启动系统时是删除所有的进程，而不是平稳地终止它们。因此，**使用reboot命令可以快速地关闭系统，但如果还有其它用户在该系统上工作时，就会引起数据的丢失。所以使用reboot命令的场合主要是在单用户模式**。

**init**是所有进程的祖先，其进程号始终为1。init用于切换系统的运行级别，切换的工作是立即完成的。**init 0命令用于立即将系统运行级别切换为0，即关机；init 6命令用于将系统运行级别切换为6，即重新启动**

62、系统当前已经加载的所有文件系统在\_\_\_\_\_\_\_\_\_文件中得到反映。

正确答案: C   你的答案: D (错误)

A、/usr/sbin/cfdisk

B、/sbin/fdisk

C、/etc/mtab

D、/etc/fstab

**/etc/mtab文件的作用：**记载的是现在系统已经装载的文件系统，包括操作系统建立的虚拟文件等；而/etc/fstab是系统准备装载的

**etc/fstab文件的作用 ：**记录了计算机上硬盘分区的相关信息，启动 Linux 的时候，检查分区的 fsck 命令，和挂载分区的 mount 命令，都需要 fstab 中的信息，来正确的检查和挂载硬盘。

63、Linux文件系统的文件都按其作用分门别类地放在相关的目录中，对于外部设备文件，一般应将其放在()目录中

正确答案: C   你的答案: C (正确)

A、/bin

B、/etc

C、/dev

D、/lib

/bin  二进制执行文件，也就是命令文件

/etc 下存放的是配置文件

/dev 存放是时外部设备文件，硬盘，usb等

/lib 存放的是库文件

64、如果某测试用例集实现了某软件的路径覆盖，那么它一定同时实现了该软件的

正确答案: D   你的答案: A (错误)

A、条件覆盖

B、判定/条件覆盖

C、组合覆盖

D、判定覆盖

白盒测试分为：

1.语句覆盖：可执行语句至少被执行一次；

2.判断覆盖：每个判断的取真分支和取假分支至少经历一次；

3.条件覆盖：每个条件的取值至少满足一次；

4.判断条件覆盖：判断和条件都满足；

5.条件组合覆盖：每个条件的所有可能都至少出现一次，并且判定结果至少出现一次 ；

他与条件覆盖的区别：他不是简单要求每个条件出现“真”和“假”两种结果，而是要求这些结果所有可能至少出现一次；

6.路径测试：执行所有可能的执行路径；

7.基本路径测试：

路径测试执行了每个路径，每个判定的结果肯定经历过一次

65、下面描述正确的是

|  |  |
| --- | --- |
| 1  2 | **int** \*p1 = **new int**[10];  **int** \*p2 = **new int**[10](); |

正确答案: C   你的答案: C (正确)

A、p1和p2申请的空间里面的值都是随机值

B、p1和p2申请的空间里的值都已经初始化

C、p1申请的空间里的值是随机值，p2申请的空间里的值已经初始化

D、p1申请的空间里的值已经初始化，p2申请的空间里的值是随机值

66、下prim函数的功能是分解质因数。括号内的内容应该为？

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | **void** prim(**int** m, **int** n)   {  **if** (m >= n)       {  **while** (        ) n++;           (     );           prim(m, n);           cout << n << endl;       }   } |

正确答案: D   你的答案: B (错误)

A、m/n m/=n

B、m/n m%=n

C、m%n m%=n

D、m%n m/=n

n从2开始

第一处为m%n，代表取余。当余数是0的时候表示除尽，跳出while循环，即找出一个质因数。此时一个质因数即为n

然后 m/=n 即让m除去这个质因数，然后再进入求新m质因数的递归。

举例：m=6，n=2

m>n;

m%n=0,跳出while，n没有加1。此时m=6，n=2

m/=n，此时m=3，n=2   （2为一个质因数）

    递归prim(m, n)，即prim(3, 2);

     m>n;

    m%n=1,n++,此时m=3，n=3，继续while循环

    m%n=0,跳出while循环，此时 m=3，n=3  （3为另一个质因数）

    m/=n,此时m=1，n=3

        递归 prim(m, n)，即 prim(1, 3);

       不满足条件（m > n），返回上层

   输出质因数 n=3

输出质因数 n = 2

67、在32位机器中，如下代码：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **void** example(**char** acWelcome[]){      printf("%d",sizeof(acWelcome));  **return**;  }  **void** main(){  **char** acWelcome[]="Welcome to Huawei Test";      example(acWelcome);  **return**;  } |

的输出是?

正确答案: B   你的答案: C (错误)

A、0

B、4

C、23

D、24

这题的正确答案应该是32位系统是4 64位系统是8，数组作为函数的参数是会退化为函数指针的，想想看，数组作为函数参数的时候经常是需要传递数组大小的

68、下面程序运行后的结果为：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **char** str[] = "glad to test something";  **char** \*p = str;  p++;  **int** \*p1 = reinterpret\_cast<**int** \*>(p);  p1++;  p = reinterpret\_cast<**char** \*>(p1);  printf("result is %s\n", p); |

正确答案: D   你的答案: C (错误)

A、result is glad to test something

B、result is ad to test something

C、result is test something

D、result is to test something

该题的关键是要认清楚强制类型转换后指针的类型。

p的类型为char \*，p++后p指向str数组的第2个元素即字母“l”的位置。

p1的类型为int \*，p1++后p1指向的位置增加4个字节，指向str数组中的第6个元素即字母“t”的位置。

因此最后p的内容为“to test something”。

选D

69、设已经有A,B,C,D4个类的定义，程序中A,B,C,D析构函数调用顺序为？

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | C c;  **void** main()  {      A\*pa=**new** A();      B b;  **static** D d;      delete pa;  } |

正确答案: B   你的答案: B (正确)

A、A B C D

B、A B D C

C、A C D B

D、A C B D

这道题主要考察的知识点是 ：全局变量，静态局部变量，局部变量空间的堆分配和栈分配

其中全局变量和静态局部变量时从 静态存储区中划分的空间，

二者的区别在于作用域的不同，全局变量作用域大于静态局部变量（只用于声明它的函数中），

而之所以是先释放 D 在释放 C的原因是， 程序中首先调用的是 C的构造函数，然后调用的是 D 的构造函数，析构函数的调用与构造函数的调用顺序刚好相反。

局部变量A 是通过 new 从系统的堆空间中分配的，程序运行结束之后，系统是不会自动回收分配给它的空间的，需要程序员手动调用 delete 来释放。

局部变量 B 对象的空间来自于系统的栈空间，在该方法执行结束就会由系统自动通过调用析构方法将其空间释放。

之所以是 先 A  后 B 是因为，B 是在函数执行到 结尾 "}" 的时候才调用析构函数， 而语句 delete a ; 位于函数结尾 "}" 之前。

70、

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | #include<iostream>  using namespace std;  **class** MyClass  {  **public**:      MyClass(**int** i = 0)      {          cout << i;      }      MyClass(**const** MyClass &x)      {          cout << 2;      }      MyClass &operator=(**const** MyClass &x)      {          cout << 3;  **return** \***this**;      }      ~MyClass()      {          cout << 4;      }  };  **int** main()  {      MyClass obj1(1), obj2(2);      MyClass obj3 = obj1;  **return** 0;  } |

运行时的输出结果是（）

正确答案: C   你的答案: C (正确)

A、11214444

B、11314444

C、122444

D、123444

C D 辨析：

关键是区分 浅/深拷贝操作 和 赋值操作:

没有重载=之前：

A a ;

A b;

a = b;

这里是赋值操作。

A a;

A b = a;

这里是浅拷贝操作。

重载 = 之后：

A a ;

A b;

a = b;

这里是深拷贝操作（当然这道题直接返回了，通常我们重载赋值运算符进行深拷贝操作）。

A a;

A b = a;

这里还是浅拷贝操作。

所以 MyClass obj3 = obj1; 调用的是拷贝构造函数。

如果写成 MyClass obj3; obj3 = obj1; 输出的结果就是 1203444

71、下列哪些容器可以使用数组，但不能使用链表来实现?

正确答案: D   你的答案: C (错误)

A、队列

B、栈

C、优先级队列

D、Map或者Dict

Map或者Dict是可以按key索引值，这个只有数组能实现，链表不能

72

using namespace std; 
class 
public: 
void FunctionAO { cout < < •FunctionA• endl;} 
virtual void FunctionBO { cout "FunctionB• << endl;} 
static void FunctionCO { cout •FunctionC" endl;} 
dass B . pub'ic 
public: 
void FunctionBO { cout •FunctionB• << end';} 
FunctionDO { cout •FunctionD• << end';} 
mainO 
B = NULL; 
b->FunctionAO; 
b->FunctionBO; 
b->FunctionCO; 
b->FunctionDO; 
return O, 

以上程序中，下列哪个函数调用会有问题（）

正确答案: B   你的答案: D (错误)

A、b->FunctionD();

B、b->FunctionB();

C、b->FunctionA();

D、b->FunctionC();

因为虚函数本质上是通过类对象的虚表进行访问，而且类的成员函数除了虚函数，其他都不存储在类当中，因此类对象不存在的情况下，无法使用虚函数，其他函数都可以正常访问（前提是这些函数都没有存取类对象的成员变量）

总结：访问虚函数，必须有对象的存在。原因：对象中存放虚函数表指针

73

下面的代码输出是（）

mainO 
Data 
union 
struct int int 
d.x = 
2, 
d.s.x = d.x * d.x; 
d.s.y = d.y + d.y; 
printfC%d d.sx d.s.y); 

正确答案: D   你的答案: A (错误)

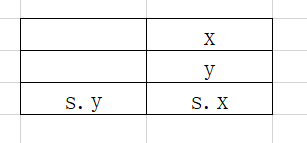
A、1 4

B、4 4

C、1 2

D、4 8

union中的所有成员相对于基地址的偏移量都为零。d.x,d.y和d.s.x的起始地址都相同,共享内存空间，给任意一个变量赋值，其他两个变量也会赋相同的值。



d. 
1 
x 
d. x*d. x 
d. y+d. 

注：由于d是局部变量， d.s.y未初始化之前输出应该是个随机值，而不是0.

74、请问实际运行结果是哪个？（）

*include < iostream> 
class 
public: 
const char* p_szNarne): ( < < m_szName < < • . ) 
virtual —Base() < < • 
— • < < m_szName < < • 
private: 
const char* m_szName: 
class 
public: 
: public 
Denved(const char* p_szName): 
, m_objA(( 
const char') • R) 
, m_ObjB(( 
const 
std::cout < < m_szName • 
virtual —Derived() 
< < m_szName < < • 

正确答案: B   你的答案: D (错误)

A、B,A,C,D,~D,~C,~A,~B

B、C,B,A,D,~D,~A,~B,~C

C、C,A,B,D,~D,~B,~A,~C

D、A,B,C,D,~D,~C,~B,~A

两个考点：

1.构造函数的调用顺序是：基类》对象》派生类，析构顺序相反；

2.构造函数里初始化列表初始化顺序由成员变量的声明顺序决定。

所以选B