**一、选择：**

**1、** 关于对象概念的描述中，（）是错误的。

**A.** 对象就是C语言中的结构变量;

**B.** 对象代表着正在创建的类的一个实体；

**C.** 对象是一个属性和操作（或方法）的封装体;

**D.** 对象之间的信息传递是通过消息进行的;

**2、** 下列给字符数组进行初始化中，（）是正确的。

**A.** char s1[ ]="12345abcd"; **B.** char s2[3]="xyz";

**C.** char s3[][3]={ 'a', 'x', 'y'}; **D.** char s4[2[3]={"xyz","mnp"};

**3、** 对于int \*pa[5];的描述，（）是正确的。

1. pa是一个指向数组的指针，所指向的数组是5个int型元素；
2. pa是一个指向某个数组中第5个元素的指针，该元素是int型变量；
3. pa[5]表示某个数组的第5个元素的值；
4. pa是一个具有5个元素的指针数组，每个元素是一个int型指针；

**4、** 在int b[][3]={{1},{3,2},{4,5,6},{0}};b[2][2]的值是（）。

**A.** 0; **B.** 2; **C.** 5; **D.** 6

**5、**下列的各类函数中，（ ）不是类的成员函数。

**A.** 构造函数; **B.** 析构函数; **C .**友元函数; **D.** 拷贝初始化构造函数;

**6、** 下述静态数据成员的特征中，（ ）是错误的。

1. 说明静态数据成员时前边要加修饰符static;
2. 静态数据成员要在类体外进行初始化；
3. 引用静态数据成员时，要在静态数据成员名前加<类名>和作用域运算符；
4. 静态数据成员不是所有对象所共用的。

**7、**（ ）是析构函数的特征。

**A.** 一个类中只能定义一个析构函数; **B.** 析构函数与类名不同;

**C.** 析构函数的定义只能在类体内; **D.** 析构函数可以有各个或多个参数。;

**8、** 已知：p是一个指向类A数据成员m的指针，A1是类A的一个对象。如果要给m赋值为5，（ ）是正确的。

**A.** A1.p=5; **B.** A1->p=5;

**C.** A1.\*p=5; **D.** \*A1.p=5;

**9、** 关于new运算符的下列描述中，( )是错的。

1. 它可以用来动态创建对象和对象数组；
2. 使用它创建的对象或对象数组可以使用运算符delete删除;
3. 使用它创建对象时要调用构造函数；
4. 使用它创建对象数组时必须指定初始值。

**10、** 派生类的对象对它的基类成员中（ ）是可以访问的。

1. 公有继承的公有成员;
2. 公有继承的私有成员；
3. 公有继承的保护成员；
4. 私有继承的公有成员。

**二、填空题**

**1、输入正整数num，按从小到大的次序输出所有的因子。**

#include<iostream.h>

#define N 40

void main ( )

{

int num,t,i,j,k,a[N];

cout<<"请输入一个正整数：\n";

cin>>num;

i=2;

k=N-1;

a[0]=1;j=0;

a[N-1]=num;t=num;

while(i<t)

{

**if( )**

{

t=num/i;

a[++j]=i;

if(i<t) **;**

}

**;**

}

for(i=0;i<=j;i++)cout<<a[i]<<'\t';

for(i=k;i<N;i++)cout<<a[i]<<'\t';

cout<<'\n';

}

**2、用1、2、3、4四个数组成四位数，输出所有可能组成的四位数，并统计四位数的个数。**

#include<iostream.h>

void main( )

{

int a,b,c,d,sum=0,x;

for(a=1; ; )

for(b=1; ; )

for(c=1; ; )

for(d=1; ; )

{

x=a\*+b\*+c\*10+d;

cout<<” ”<<x;

sum=+1;

}

cout<<”\nsum=”<<sum;

}

**3、函数compress能为首指针为head的有序链表进行压缩（值域相等的结点只保留一个），并将等值结点的个数存放在count中。**

结点定义如下：

typedef struct node

{

int data, count;

node2 \*next;

}snode, \*ptr;

void compress()

{

ptr p1,p2;

p1=head;

while()

{  **;**

if(p1->data==p2->data)

{

p1->count++;

p1->next= **;**

delete p2;

}

else() ;

}

}

**4、#include<iostream.h>**

class test{

private:

int num;

float f1;

public:

test();

test(,);

int getint(){return num;}

float getfloat(){return f1;}

};

test **::** test()

{

cout<<"默认初始化"<<endl;

=0;

=0.0;

}

test **::** test( n, f)

{

cout<<"初始化"<<endl;

=n;

=f;

}

void main()

{

test a;

test b(2,5.5);

}

**三、读程序题，**

**1、**#include<iostream.h>

int func(int,int);

void main( )

{

extern int x,y;

cout<<func(x,y)<<endl;  **结果：**

}

int x=15,y=-5;

int func(int a,int b)

{

int s;

s=a+b+x+y;

return s;

}

**2**、#include<iostream.h>

int add(int x=15,int y=10)

{

return x+y;  **结果：**

}

void main( )

{

int a=8;

cout<<add( )<<’\t’;

cout<<add(a)<<’\t’;

cout<<add(a,add(a))<<’\t’;

cout<<add(a,add(a,add( )))<<’\n’;

}

**3、**#include<iostream.h>

void main ( )

{

int a[8]={10,20,30,40,50,60,70,80};  **结果：**

int \*pi;

pi=&a[0];

pi++;

cout<<"\*pi="<<\*pi<<'\n';

pi=&a[0];

cout<<"\*pi++="<<\*pi++<<'\n';

pi=&a[0];

cout<<"\*(pi++)="<<\*(pi++)<<'\n';

pi=&a[0];

cout<<"\*++pi="<<\*++pi<<'\n';

pi=&a[0];

cout<<"(\*pi)++="<<(\*pi)++<<'\n';

pi=&a[0];

cout<<"++\*pi="<<++\*pi<<'\n';

}

**4、**#include<iostream.h>

int number1;

int &set1( )

{ return number1; }  **结果：**

int &set2( )

{ static int number2;

return number2; }

int &max(int &m,int &n)

{ return (m>n?m:n); }

void main ( )

{ set1( )=5;

cout<<set1( )<<'\t'<<number1<<'\n';

number1=7;

cout<<set1( )<<'\t'<<number1<<'\n';

set2( )=6;

int a=set2( );

cout<<a<<'\n';

int b=3,c=4;

cout<<max(b,c)<<'\n';

max(b,c)=2;

cout<<"b="<<b<<"\tc="<<c<<"\tmax(b,c)="<<max(b,c)<<'\n';

max(b,c)=10;

cout<<"b="<<b<<"\tc="<<c<<"\tmax(b,c)="<<max(b,c)<<'\n';

}

**5、**#include<iostream.h>

class A{

private:

int i;

public:

A (int a){

i=a;

cout <<"i="<<i<<'\t'<<"调用了构造函数!\n";

}

~A ( ) {cout<<"i="<<i<<"调用了析构函数!"<<'\n';}

};

A a(0);  **结果：**

void main ( )

{

A a1(10);

A a2=20;

a2=50;

cout<<"main()函数结束！"<<endl;}

**6、**#include<iostream.h>

class A

{ int x;

static int y;

public:

A(int x1,int x2)

{ x=x1; y=y+x2; }

static void show1();

static void show2(A a);

};

void A::show1( )

{ cout<<"Y="<<y<<endl; }

void A::show2(A a)

{cout<<"X="<<a.x<<"\t"<<"Y="<<y<<endl; }

int A::y=6;  **结果：**

void main( )

{

A a1(11,22);

a1.show1();

A::show2(a1);

A a2(33,44);

A::show1();

a2.show2(a2); }

**7、#include <iostream.h>**

class AA

{ public:

AA(int i,int j)

{A=i;B=j;cout<<"Constructor\n";}

AA(AA &obj)

{A=obj.A+1;B=obj.B+2;cout<<"Copy\_Constructor\n";}

~AA()

{cout<<"Destructor\n";}

void print()

{cout<<"A="<<A<<",B="<<B<<endl; }

private:

int A,B;  **结果：**

};

void main()

{ AA a1(2,3);

AA a2(a1);

a2.print();

AA \*pa=new AA(5,6);

pa->print();

delete pa;}

**8、**#include<iostream.h>

class CBase

{

public:

virtual void f1()

{cout<<"调用函数CBase∷f1()!"<<endl;}

virtual void f2()

{cout<<"调用函数CBase∷f2()!"<<endl;}

void f3()

{cout<<"调用函数CBase∷f3()!"<<endl;}

};

class CDerived:public CBase

{

void f1()

{cout<<"调用函数CDerived∷f1()!"<<endl;}

void f3()

{cout<<"调用函数CDerived∷f3()!"<<endl;}

};

void main()

{

CBase obj1,\*P;

CDerived obj2;  **结果：**

P=&obj1;

P->f1();

P->f2();

P->f3();

P=&obj2;

P->f1();

P->f2();

P->f3();

}

**四、编程题**

1、定义一个学生的类：

数据成员包括：姓名、学号、英语、数学、计算机三门成绩。

成员函数包括：设置姓名、学号和三门课的成绩、输出数据、以及求平均成绩。

2、编写一个函数，将参数P中元素颠倒次序。

函数原型：**float \*mirror(float \*p[ ],int n);**

其中：\*mirror：返回类型为指针的函数

float \*p[ ]：为指针数组

**一、填空题**

1、在字长为32位的机器中，**sizeof(char)＝** 1 字节，**sizeof(unsigned int)＝** 4 字节.

2、C++语法规定，任何变量在使用前必须先 dingyi ；变量有两个值，一个是变量本身的值，另一个是变量的 dizhizhi 。

3、C++中任何一个数组的名字是一个 changliang 指针，该指针的值是该数组 shouyuanshu 的地址。

4、函数调用时的参数传递主要分为单向传递和 shuangxiang 传递，前者的特点是 bugaibu 。

5、函数重载时要求同名函数的参数 或 不同，否则无法确定是哪个函数。

6、静态数据成员是类的所有对象中gongxiang 的成员，静态数据成员初始化与一般数据成员初始化 butong

**二、判断题**

1、在说明语句 int a(15),&b=a,\*p=&a;中，b的值的\*P的是相等的。（　　）

2、在不同类型操作数组成的表达式中，其表达式的类型一定是最高类型double型。（　　）

3、break语句也可以用于if体内，它可退出if语句。（　　）

4、在一个被调用函数中，一个return语句可返回多个值给调用函数。（　　）

5、在传值调用中，要求实参和形参的类型完全一致并且个数相等。（　　）

6、for循环中，循环变量的作用域是该循环的循环体内。（　　）

7、类中所提供的成员函数可以直接访问私有成员。（　　）

8、私有继承中，基类中所有成员对派生类的对象都是不可见的。（　　）

**三、读程序题，写出运行结果**

1、#include <iostream.h>

void main()

{

int a,b,c,d=5;

c=2,c+=10; 结果：

a =b=10;

a \* = 2;

b / = 2;

c % = 2;

cout<<a<<","<<b<<","<<c<<endl;

}

2、#include <iostream.h>

void main()

{

int a = 50 , b=0 ; 结果：

b = ++a;

cout<<a<<","<<b<<endl;

b = a++;

cout<<a<<","<<b<<endl;}

3、#include <iostream.h>

void main()

{

int f = 2002 , x;

if( f != 3)

x = 2003 ; 结果：

else

x = 20 ;

cout<<f<<","<<x<<endl;

}

4、#include <iostream.h>

void main()

{

int i=1,sum=0; 结果：

while(i<=10)

sum+ = ++i;

cout << "sum=" << sum << ", i=" << i<<endl;

}

5、#include <iostream.h>

void main()

{

int i;

for(i=4 ; i<=10 ; i++ )

{ if (i%3= =0) continue; 结果：

cout<<i; }

}

6、#include <iostream.h>

void main()

{

char flag='c' ;

switch(flag)

{

case 'a' :cout<<"1"<<endl ;

case 'b' :cout<<"2"<<endl ; break; 结果：

case 'c' : cout <<"3"<<endl ;

default : cout <<"4"<<endl ; break;

}

}

7、#include <iostream.h>

void main()

{

static int b[][3] = { {1,2,3},{4},{5,6} };

b[0][2]=12,b[1][2]=18;

cout << \*\*b<<"\t"<<\*\*(b+1) <<"\t"<<\*\*b+1<<"\t"<<\*(\*(b+1)+2) <<"\n";

cout << b[0][2]+b[1][2]+b[2][2]<< endl; 结果：

}

8、#include <iostream.h>

void Swap( int &a, int & b);

void main()

{

int x=10, y=7;

cout<<"x="<<x<<" y="<<y<<endl;

Swap( x , y );

cout<<"x="<<x<<" y="<<y<<endl; 结果：

}

void Swap(int & a, int & b)

{ int temp; temp = a ; a=b ; b=temp ; }

9、#include <iostream.h>

int add(int a, int b);

void main()

{

extern int x, y;

cout<<add(x,y)<<endl;

x/=y;

cout<<add(x,y)<<endl; 结果：

}

int x=50,y=100;

int add(int a, int b)

{ int s=a+b ; return s; }

10、#include <iostream.h>

class A

{

public:

A();

A(int i,int j);

~A(){cout<<"Donstructor.\n";}

void print();

private:

int a,b;

};

A::A()

{ a=b=10;cout<<"Default constructor.\n";} 结果：

A::A(int i,int j)

{ a=i,b=j;cout<<"Constructor.\n";}

void A::print()

{cout<<"a="<<a<<",b="<<b<<endl;}

void main()

{

A m,n(15,18);

m.print();

n.print();}

**四、编程题**

从键盘上输入数组s1[2][3] 的各元素，并将数组s1转置后存入数组s2中，最后输出数组s1和数组s2的各元素。