1. **单项选择（本题共 20 分，每小题 2分）**
   1. **设int A[4][4]; int (\*p)[4]=A; 则下列表达式中，与A[2][3]不等价的是（ B ）**

A. p[2][3]

B．\*（\*p+2）+3

C．（\*（p+2））[3]

D. \*(\*(p+2)+3)

分析：

// \*(\*(p+i)+j)=p[i][j]=A[i][j];

// \*(\*p+2)+3=A[0][2]+3

**1.2 下列关于字符指针初始化的语句中，正确的是（ A ）**

A．char \*s=new char;

B. char s[20]; s=”abcde”;

C. char \*\*s={“abced”,”efgh”};

D. char \*s[10]=”abced”;

分析：

// A这么理解，char\*s,s是指针，new char分配了一个地址给这个指针，相当于给指针赋了初值（即使这个地址并没有存值）

//B 指的是字符数组，而且并不属于初始化

// C,D无效声明

**1.3 设int a=10;int &r=a; 则下列语句正确的是（ C ）**

A．&r=20;

B. int \*p=r;

C. int &ra=r;

D. int &\*p=&a;

分析：

// <类型> & <变量名>=<对象变量名> ;

// r就是a的别名，赋值直接采取 r=20; &r指的是r的地址（实际上是a的地址，引用本身不具有独立的变量地址）

// B,D同一个问题，不存在指向引用的指针

**1.4 已知类ExamClass 的定义，则下列语句中未调用类构造函数的是（ B ）**

A． ExamClass obj1；

B． ExamClass \*obj2；

C． ExamClass obj3[10]；

D． ExamClass \*obj4=new ExamClass()；

//用第二题来理解这道题

// 声明一个指针并没有分配内存，自然也不会调用构造函数

// 至少对指针进行动态分配内存，才会调用构造函数（D选项）

**1.5 下列关于运算符重载的描述中，错误的是（ C ）**

A．类的运算符重载有成员函数和友元函数两种方式

B．可以通过函数调用表达式的方式使用重载运算符

C．任何合法的C++运算符都可以重载，但无法改变其优先性和结合性

D．派生类可以使用基类的公有运算符重载函数

//并不是所有的运算符都可以重载哦（. :: ?: .\* sizeof）

//还有一些只能以类成员而不能以友元身份重载的

//B是对的。比如“-”一般这么调用 a=b-c; 也可以 a=b.operator – ( c );

**1.6 下列关于派生类的描述中，正确的是（ D ）**

A. 派生类可以访问基类的所有成员

B．抽象基类的派生类一定不是抽象基类

C．派生类不能继承基类的静态成员

D．派生类对象可以赋值类基类对象

//A.派生类不能访问基类的私有成员

//B. 如果派生类中没有定义基类的那一纯虚函数，而只是继承了基类的纯虚函数的话，则这个派生类还是一个抽象基类

//C.静态属性可以继承

//D.派生类可以复制基类，只赋值共性部分。基类不能赋值派生类，即<基类对象>=<派生类对象>

**1.7 设函数模板template<class T> void( T a, T b);则下列对该模板错误的调用是（ B ）**

A．int a=1 , b=2 ; f( a , b ) ；

B. int \*a[10]={ 0 } , \*pa ; f( a , pa );

C. double a=1 , b=2 ; f( a , b );

D. char \* a=“abc” , \*b=“def” ; f( a , b );

//模板函数的调用不支持形参的自动转化

//模板函数这么定义，形参类型必须一致

//B错误，a是指针数组名，意义是二重指针；pa是一重指针；

//D 正确，a,b都是一重字符指针

**1.8 下列关于模板的描述中，错误的是（ D ）**

A．类模板可以使用类型参数和普通参数

B．不能创建类模板对象

C．类既可以派生出新类，也可以派生出新的类模板

D．基类模板和派生类模板只能使用相同的类型参数

//派生类模板可以使用不同的类型参数

// B类模板不能直接创建对象，必须先为模板参数指定实参

**1.9下列函数中，不能用于磁盘文件输出的是（ A ）**

A．tellg

B. operator<<

C. put

D. write

//read和write 多用于读写二进制文件

//get 和put 多用于读写文本文件

// 继承来的”<<”和“>>”可以用于读写文件

// tellg和tellp获取读入指针和读出指针的当前值

**1.10 设int a=010;则语句cout<<hex<<a<<””<<dec<<””<<oct<<a<<””<<endl;的输出结果是( C )**

A．A 10 12

B．8 10 8

C．8 8 10

D．10 8 20

//以0开头的为8进制数，则a储存的是8进制数 010；

// hex转化成16进制输出格式，dec转化成10进制输出格式，oct转化成8进制输出格式；

1. **程序改错（本题共16分）**

**2.1（8分）请找出以下程序中的4个语法错误，用横线标出错误所在行，并进行改正或说明错误原因。**

#include<iostream>

using namespace std;

class One

{

int a;

protected:

void output(){ cout<<"class One"<<endl; }

public:

One(int a){ this->a=a; }

int getA(){ return a; }

}**；//类定义后要分号：张莹老师课件上的，不过有的编译器不带貌似没问题**

class Two : public One

{

int b;

public:

void output()

{

cout<<"class Two"<<endl;

}

Two( int b**=2** )**:One(b)**{ this->b=b; }**//Two是One的派生类，One中定义了带形参表的构造函数，Two中就应该有构造函数，并且要构造One中的元素（基类中如果有私有成员变量通常调用基类的构造函数），而且main函数中居然有一个没有初始列表的Two形变量，只能在这里加一个初始化列表了**

};

class Three

{

One one;

Two two;

public:

Three(int a):one(a),two(one**.**getA()){}

void output(){ cout<<"class Three"<<endl; }

};**//数据a是私有成员，派生类不可以调用，只能通过共有函数成员getA（）来调用a的值,或者·····你干脆写个a也是可以的 : )**

int main()

{

Two A ；

Three B(2);

A.output();

B.output();

return **0** ;**//int main()函数必须返回值；当然你也可以把 int改成void**

}

**2.2（8分）请找出以下程序中的4个语法错误，用横线标出错误所在行，并进行改正或说明错误原因。**

#include<iostream>

using namespace std;

class A

{

int a;

public:

virtual void output()**{cout<<”class A”;**}**//main函数中声明了A类的对象，所以不能是个纯虚函数，而且还调用了这个函数，那就得定义一下咯？**

};

class B:**public** A **//只有公有继承的才能将B类对象的值赋给A类对象，main函数中有一个将A类指针赋值给B类指针的，虽然它也错了，不过好像不能删（其实删了我也凑不出四个了），这里就改成公有继承吧**

{

int b;

public:

B(int b=0):b(b){}

void output(){ cout<<"class B"<<endl;}

};

int main(){

A\* pa=new A();

pa->output();

B\*pb=new B();

**pa=pb**;**//只有指向派生类的指针可以赋值给指向基类的指针，反过来不行，不过这一步意义不大啊？**

pb**->**output();**//指针对象调用类成员要用->**

return 0;

}

1. 读程序写结果（本题共24分）

**3.1(6分)**

#include<iostream>

#include<iomanip>

using namespace std;

void main()

{

int m=2,n=3,i,j;

int \*\*b;

b=new int\*[m];

for(int i=0;i<m;i++)

b[i]=new int[n];

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

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

b[i][j]=i\*10+j;

cout.setf(ios::left);

cout.fill('\*');

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

for(int j=0;j<n;j++)

cout<<setw(3)<<b[i][j];

cout<<endl;

}

}

在Visual C++.Net环境下的运行结果为

**0\*\*1\*\*2\*\***

**10\*11\*12\***

**分析：**

**动态分配内存建立二维数组b[2][3],且b[i][j]=i\*10+j;**

**cout.setf(ios::left);设置为左边对齐输出，且在右边的域宽范围内填充对应字符；**

**cout.fill('\*');设置填充字符为“\*”；**

**cout<<setw(3) <<b[i][j]; 设置域宽范围为3；**

**注意到换行符是在i每增加1（不是j）换一次，且域宽为3，填充符为“\*”，应该为**

**0\*\*1\*\*2\*\***

**10\*11\*12\***

**3.2（6分）**

#include<iostream>

#include<string.h>

using namespace std;

char &get\_val(char\* &,int);

int main(void)

{

char \*s=new char[10];

strcpy(s,"123456");

cout<<s<<endl;

int i=0;

while(s[i]!=0){

get\_val(s,i)=0x61+i;

i++;

}

cout<<s<<endl;

return 0;

}

char &get\_val(char\* &str ,int ix)

{

return str[ix];

}

在Visual C++.Net环境下的运行结果为

123456

abcdef

**分析:**

**前面的123456很容易理解，中间换行；**

**看一下get\_val这个函数**

**char &get\_val(char\* &str ,int ix)**

**{**

**return str[ix];**

**}**

**函数返回值为引用类型，而且把字符串str作为引用类型的参数，那么这个函数的作用就是返回字符串str的第ix个字符的别名（同样代表这个别名的值），这个别名可以被赋值。**

**对这个别名赋值也就是对这个别名代表的字符赋值。**

**所以这个字符串的6个字符依次被赋值为0x61+i(i=0,1,2,3,4,5);**

**0x61(16进制)代表10进制97，也就代表‘a’，所以被依次赋值为abcedf**

**这个函数就算不怎么理解，估计大家猜都猜得出来**

**3.3**

#include<iostream>

using namespace std;

class A

{

int m\_data;

public:

A(int data=0)

{

m\_data=data;

}

int GetData()

{

return doGetData();

}

virtual int doGetData()

{

return m\_data;

}

};

class B:public A

{

int m\_data;

public:

B(int data=1)

{

m\_data=data;

}

int doGetData()

{

return m\_data;

}

};

class C:public B

{

int m\_data;

public:

C(int data=2)

{

m\_data=data;

}

int doGetData()

{

return m\_data;

}

};

void main()

{

C c;

cout<<c.GetData()<<endl;

cout<<c.A::GetData()<<endl;

cout<<c.B::GetData()<<endl;

cout<<c.C::GetData()<<endl;

cout<<c.doGetData()<<endl;

cout<<c.A::doGetData()<<endl;

cout<<c.B::doGetData()<<endl;

cout<<c.C::doGetData()<<endl;

}

在Visual C++.Net环境下的运行结果为

2

2

2

2

2

0

1

2

分析：

B，C中没有GetData函数，所以c.A(B,C)::GetData()都是从A中继承过来的同一个函数，调用时都是在C中调用同一个函数，再取C中的doGetData（）,所以最后取的都是C中的m\_data值，doGetData()是A中定义的虚函数，在B，C中都有重载，所以限定A,B,C前缀时，调用的分别是是A，B，C中的GetData值，也就是对应到了A,B,C中的data值；

**3.4（6分）**

#include<iostream>

#include<fstream>

using namespace std;

void main()

{

int a[10]={31,34,41,45,49,56,63,74,92,95};

for(int i=0;i<10;i++){

ofstream fout1("l.data",ios::binary|ios::app);

fout1.write((char\*)(a+i),sizeof(int));

fout1.close();

}**//标记1**

int b,i=10;

ifstream fin("l.data",ios::binary);

while((!fin.eof())&&i>0)

{

fin.seekg((i-1)\*sizeof(int));

fin.read((char\*)(&b),sizeof(int));

cout<<b<<" ";

i--;

}**//标记2**

cout<<endl;

fin.close();

}

在Visual C++.Net环境下的运行结果为

95 92 74 63 56 49 45 41 34 31

分析：

标记1处这个循环，打开二进制文件"l.data"进行写入，依次将数组a中的元素写入，再关闭。

标记2处这个循环，eof()判断是否到达文件末尾，到达了返回非0，没到达返回0；!eof()&&i>0,保证了既没有到达文件末尾i也大于0；

fin.seekg((i-1)\*sizeof(int));

fin.read((char\*)(&b),sizeof(int));这个东西张莹老师上课说过的，

seekg是位置重置函数，seekg((i-1)\*sizeof(int)),当i=10时，向后重置了36个字节（9乘以4，我也不知道我为什么要写，我总觉得不放心），再把36-39这个一段的值赋值给b，之后就是向后重置32个28个···，相当于反序输出了。

这个题，其实稍微听了点课一定可以猜出来的 : ) （笑脸警告）

1. **程序填空（本题共20分，每空4分）**

**4.1以下程序的功能是要输出name：Jeff classid：1001.请完善该程序**

#include<iostream>

#include<string.h>

using namespace std;

class mClass

{

int classid;

int score;

public:

mClass(int c,int s)

{

classid=c;

score=s;

}

int getScore()

{

return score;

}

int getClassid()

{

return classid;

}

};

class Student

{

char \*name;

mClass \*studentClass;

public:

**void virtual print()=0//main 函数中Student类型指针调用了print函数，无论如何这里要有一个print函数，即使你之后使用的将是它的派生类中的函数哦也**

};

class Graduate:public Student

{

char \*name;

mClass \*studentClass;

public:

Graduate(const char \*n,int c,int s=0)

{

name=new char[strlen(n)+1];

strcpy(name,n);

studentClass=new mClass(c,s);

}

void print()

{

**cout<<"name:"<<name<<" ";**

**cout<<"classid:"<<studentClass->getClassid()<<"。";//需要输出啥你就给啥就行了，顺便说一下classid必须用这个函数获取，Classid是私有成员不能直接从派生类中调用**

}

};

int main()

{

Student \*gStudent=**new Graduate(“Jeff”,1001,0)//这里必须用基类指针指向派生类了，貌似这个基类里面并没有关于Classid的一切**

gStudent->print();

return 0;

}

**4.2以下程序的功能是输出n=f，请完善该程序。**

#include<iostream>

using namespace std;

template <class T>

class TemplateTest**//你这个名字敢不敢再真实一点**

{

T n;

public:

TemplateTest(**T i**):n(i){}**//这里赋值，很简单**

void operator++();

void disp()

{

cout<<"n="<<n<<endl;

}

};

template<class T>

**void TemplateTest<T>::operator++()//这里就看你的记忆力了，记住对着写就行了**

{

n+=1;

}

void main()

{

**TemplateTest<char>s('e');//记住怎么声明的，而且下一行要加一，所以要初始化成f前面那个----e；**

++s;

**s.disp();//这种灯笼打着题目都找不到 ：）**

}

**4.3以下程序的功能是读取文件data.txt中所有字符串，判断其中长度大于5的字符串是否是回文串，如果是，则将其输出到屏幕上。请完善该程序**

**//回文串是正反读都一样的字符串**

#include<iostream>

#include<fstream>

#include<string.h>

using namespace std;

int main()

{

char \*str=new char[81];

ifstream file;

**file.open(“data.tex”);//由于是先声明再使用的，这里要用到open**

while(file>>str);

{

if(strlen(str)>5)

{

int flag=0;

for(int i=0;i<strlen(str)/2;i++)

{

if(**str[i]!=str[strlen(str)-i-1]**)**//依次判断对应位置的字符是否一样，不一样GG**

{

flag=1;

break;

}

}

if(flag==0)

{

**cout<<str;//这个总得猜出来吧**

}

}

}

file.close();

return 0;

}

**5.1（10分）**

**叫号机是银行的等公共场所常见的设备。一般情况下，可以用一个递增链表类来记录叫号机的状态。链表中的每一个节点用两个数据成员来记录信息，Number表示号值（从整数1开始逐个递增），Status表示当前号状态，号有四种状态：已被使用（过号也算被使用）、正在被服务、正在等待、还未使用。**

**（1）设计一个成员函数，AddElement，该函数首先判断当前链表是否为空，如果空，依次生成100个节点（号码从1到100，状态都还是未使用）。如果非空，判断是否还有状态还未使用的节点，如果没有，依次生成100个节点（号码从当前最后一个号码+1到+100，状态都还是未使用）；如果有，什么都不做。**

**（2）设计一个成员函数，GetNumber，完成取号功能：从排队链表中去除最前面还未使用的号；**

**（3）设计一个成员函数，CallNumber,完成叫号功能；在处理完当前服务的号后，从等待的号中取出最靠前的号，开始服务。**

#include<iostream>

using namespace std;

struct delt{

int number;

int status;

delt \*next;

};

class list {

public:

delt\*head;

int number;

list() { number = 0; }

void AddElement();

int Getnumber();

int Callnumber();

};

void list:: AddElement() {

if (number == 0) {

int i = 0; delt \*temp,\*tail;

head = new delt; head->number = 1; head->status = 0; head->next = NULL;

tail = head;

while (i <= 99) {

temp = new delt; temp->number = i+1; temp->status = 0; temp->next = NULL;

tail->next = temp; temp = tail; i++;

}}

else {

delt \*q; q = head;

while (q->next!= NULL) { if (q->status == 0)return; q = q->next; }

if (q ->status==0) { return; }

int i = 1; delt \*temp, \*tail;

tail = q; i = 0;

while (i <= 100) {

temp = new delt; temp->number = 100+i; temp->status = 0;

tail->next = temp; temp = tail; i++;

}

}

}

int list::Getnumber() {

delt\*q; q = head;

while (q != NULL) {

if (q->status == 0)

q->status = 1; return q->number;

}return 0;

}

int list::Callnumber() {

delt\*q; q = head;

while (q != NULL) {

if (q->next->status == 1) {

q->status = 3; q->next->status=2;

return q->next->number;

}

}return 0;

}

int main() {

cout << "hello world!";

};

**5.2（10分）**

**微信是目前中国最火的社交软件之一。微信的设计理念以用户体验为先，最大限度减少对用户的干扰。因此，指导2015年1月25日，微信才推出了朋友圈广告。而且微信广告都是根据用户情况定制推送的，不同人看到的广告是不同的。以首批微信朋友圈广告为例，广告只有三个，宝马汽车，VIVO手机和可口可乐。按照业界传闻，首批广告就价值1000万，宝马汽车目标客户是“中年富裕阶层”；VIVO手机目标客户是“青年中产阶层”；可口可乐目标客户是“青少年阶层”。微信根据性别、年龄、爱好、地理位置、所加入朋友圈、所连接的微信好友的等一些用户标签以及其他信息，对用户进行精准匹配。**

**请根据以上描述，回答如下问题：**

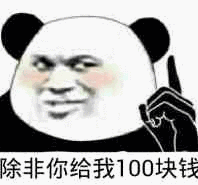
1. **猜测微信如何定位用户所属的类型，简述算法思路并给出相应简单代码。**

**根据对题目数量的分析，大部分人做到这里时，时间已经所剩无几。我建议，先简述算法思路，之后转而去做第二题，~~这样的话，这道题就跟c++程序无关了~~。**

**算法思路：1.确定重要性，如此多的筛选条件，哪一个更重要是首要问题。比如，确切的年龄比地理位置等更能分清阶级。在确定一个权重后加和（或是其他计算方式），可以得到最终结果。**

**2.划定界限，青少年与青年，青年与中年之间的界限是模糊的，应该明确的指出这一界限所在。（可以在这里，运用类的思想，每个判断标准都可以分为一个类，这个地方还可以写很多代码）**

**3.对于每一个判断标准，你都可以单独拿出来阐述。比如，所连接的微信好友，好友大部分是同龄人，所以可以通过好友来判断。~~这里就可以写4，5，6，7，8…….~~**

**简单代码：**

1. **如果你属于“青少年阶层”，应该如何将自己伪装成一个“中年富裕阶层”的用户呢？简述算法思路 ~~并给出相应简单代码（这也要代码？？）~~。**
2. **性别改为男**
3. **年龄改为40以上**
4. **微信名称改为XX富豪**
5. **个性签名：我真的是中年富裕阶层，真的，我想买宝马**