暑期VIP MOOC期末复习

2018-9-25

清华计算机系 潘祖江

考试范围

- C++基础
- 函数
- 类与对象
- 数组及指针
- 类的继承
- 模板/流

C++基础

- C++的关键字和标识符
- C++程序的基本结构
- 基本数据类型及枚举类型
- 变量及常量的定义和使用
- 变量的初始化，以及未经初始化即使用变量的后果
- 运算符的优先级（会查书上的表就行）
- 控制结构（选择、循环、break语句、continue语句）
- 算术运算的运算规则、算术表达式
- 关系运算和逻辑运算的运算规则、关系表达式和逻辑表达式

C++基础

- 在编译指令中，宏定义使用的指令是

A. #include　　***B. #define***　　C. #if　　D. #else

- 头文件扩展名为

A. cpp　　***B．h***　　C．ub　　D．ob

C++基础

• C++中函数中的return指令可以

A．只能有一条

B． 0或多条

C. 至少有一条

D．只能主函数调用

C++基础

• *1.* 设有定义int i = 4; double j＝5;，则10+i+j值的数据类型是（）

A. int

B. double

C. float

D. 不确定

C++基础

• int n=0; while（n=1）n++;
while循环执行次数是___。

C++基础

• 设x和y均为int型变量，则执行下面的循环后，y值为（ ）。

```
for(y=1， x=1; y<=50;y++)
{ if(x>=10) break;
if (x%2==1)
{ x+=5; continue;}
x-=3;
}
```

A.2　B. 4　***C. 6***　D. 8

C++基础

- 假定a和b为int型变量，则执行以下语句后b的值为（ ）。

```
a=1; b=10;
do
{ b-=a; a++; }
while (b--<0);
```

A. 9　B. –2　C. -1　***D. 8***

函数

- 函数定义的语法
- 函数原型声明、函数调用的语法
- 函数嵌套调用时的执行过程
- 函数递归调用时的执行过程
- 函数的默认参数值
- 函数重载（全局函数重载、成员函数重载）
- 函数的参数传递方式（传值、传引用），以及各自不同的效果
- 内联函数的语法

函数

• C++语言中所有在函数中定义的变量，连同形式参数，都属于（）

A. 全局变量
B. 局部变量
C. 静态变量
D. 函数

函数

• 使用地址作为实参传给形参，下列说法正确的是（）

A. 实参是形参的备份
B. 实参与形参无联系
C. 形参是实参的备份
D. 实参与形参是同一对象

函数

• 执行以下程序后的输出结果为（ ）。

```
viod fun (int a，int b，int c)
{ a=4; b=5; c=6;a=b+c;b=c+a;c=a+b;}
main()
{ int x=10， y=20，z=30;
fun (x，y，z);
cout<<x<<’，’<<y<<’，’<<z<<endl;
}
```

A.30，20，10　　***B.10，20，30***

C.11，17，28　　D.4，5，6

函数

• 函数重载必须满足的条件是

A. 函数名相同

B. 参数个数不同

C. 参数类型不同

D. 函数名不相同

函数

- 当需要将一个函数 bool isnumber (char c) ; 声明为内联函数时，则此内联函数的函数原型为（ ）。

A. enum bool isnumber (char c) ;

B. define bool isnumber (char c) ;

C. inline bool isnumber (char c) ;

D. extern bool isnumber (char c) ;

类与对象

- 结构体的基本语法，结构体成员的默认访问权限
- 类的定义
- 类成员函数的实现
- 类成员的访问
- 构造函数、析构函数
- 类的组合（组合类的定义、对内嵌对象的访问、组合类对象构造时构造函数的执行次序）
- 静态数据成员、静态成员函数、常成员函数
- 友元函数、友元类

类与对象

• 设有以下说明语句：

```
struct ex
{int x;float y;char z;} example;
```

• 则下面的叙述不正确的是

A.struct是结构体类定义的关键字

B.example是结构体类型名

C.x,y,z都是结构体成员名

D.struct ex是结构体类型名

类与对象

• 对于结构体中定义的成员，其隐含访问权限为（ ）。

A. public B. protected

C. private D. static

类与对象

• 下列有关类的说法不正确的是

A 类是一种用户自定义的数据类型
B 只有类中的成员函数才能存取类中的私有数据
C 在类中，如果不特别说明，所指的数据均为私有类型
D 在类中，如果不特备说明，所指的成员函数均为公有类型

类与对象

• 下列关于类的构造函数和析构函数的叙述中，不正确的是

A.类的析构函数可以重载
B.类的构造函数可以重载
C.定义一个类时可以不显示定义构造函数
D.定义一个类时可以不显示定义析构函数

类与对象

- 下列关于拷贝构造函数的描述中，错误的是

A.拷贝构造函数是一个成员函数
B.拷贝构造函数的名字与该类的类名相同
C.拷贝构造函数的参数可以1至多个
D.拷贝构造函数没定义时，系统提供一个默认的拷贝构造函数

类与对象

- 已知类Sample中的一个成员函数说明如下：void Set(Sample&a);
  其中Sample& a的含义是

A 类Sample的指针
B 将a的地址值赋给变量Set
C a是类Sample的对象引用，用来做函数Set()的形参
D 变量Sample与a按位相与作为函数Set()的参数

类与对象

• 假定一个类的构造函数为“A(int i=4, int j=0) {a=i;b=j;}”, 则执行“A x (1);”语句后，x.a和x.b的值分别为（）

A. 1和0

B. 1和4

C. 4和0

D. 4和1

类与对象

• 类的构造函数被自动调用执行的情况是在定义该类的（）

A. 成员函数时

B. 数据成员时

C. 对象时

D. 友元函数时

类与对象

- 假定AB为一个类，则该类的拷贝构造函数的调用语句为：（ ）。

A.AB x,y(x);

B. AB x,y;

C. AB x,y(AB &);

D. AB x,y(AB &x);

类与对象

- 假定AB为一个类，则执行“AB a(4),b[3],*p[2]；”语句时，自动调用该类构造函数的次数为（ ）。

A.3

B.4

C.6

D.9

类与对象

- 已知类A是类B的友元，类B是类C的友元，则（）

A. 类A一定是类C的友元

B. 类C一定是类A的友元

C. 类C的成员函数可以访问类B的对象的任何成员

D. 类A的成员函数可以访问类B的对象的任何成员

类与对象

- 假定一个类的构造函数为A(int aa,int bb) {a=aa--;b=a*bb;},则执行A x(4,5)；语句后，x.a和x.b的值分别为（）

A. 3和15

B. 5和4

C. 4和20

D. 20和5

类与对象

• 以下程序执行的结果是

A.11 ***B.21*** C.30 D.31

```
#include<iostream>
using namespace std;
class Sample {
    int n;
public:
    Sample(){}
    Sample(int i){n =i;}
    Sample & operator=(Sample);
    void disp(){cout<<"n="<<endl;}
}
Sample &Sample::oprator=(Sample s)
{ n=s.n+1;
    return *this;
}
void main() {
Sample s1(10),s2,s3(20);
(s2=s1)=s3;
s2.disp();
}
```

类与对象

在下面横线处填上适当的语句,使类型该程序执行结果为10。

```
#include <iostream>
using namespace std;
class MyClass
{
public:
    ____________________
    MyClass(int a)
    {
    ____________________
    }
private:
        int x;
};
void main()
{
    MyClass my(10);
    cout<<my.GetNum()<<endl;
 }
```

类与对象

```
#include <iostream>
using namespace std;
class MyClass
{
public:
    int GetNum(){return x;}
    MyClass(int a)
    {
        x=a;
    }
private:
        int x;
};
void main()
{
    MyClass my(10);
    cout<<my.GetNum()<<endl;
 }
```

数组及指针

- 数组的定义以及初始化
- 用数组名做函数的参数/用数组元素做函数的参数
- 用指针做函数的参数
- 用字符数组存储字符串（C风格的字符串），以及C风格的字符串处理
- 指针的定义，通过指针访问指向数据
- 指针的运算
- 用指针访问数组元素
- 指针数组
- this指针
- 动态内存分配与释放：动态构造基本类型数据、对象、数组、对象数组，以及释放这些对象

数组及指针

- 已知：int I; int* pi = &I; int* &p = pi;则p是

A．指向变量I的变量
B.指向变量pi的变量
C. 变量pi的引用
D.语法是错误的

数组及指针

- 若有:int a[6]={4，5，6，9，5，7}，*p=a， *q=p; 则对数组元素的错误引用是（ ）。

A. a[4]　B. *(p+4)　***C. *a++***　D. *q++

数组及指针

- 下列关于指针的说法，错误的是

A.两个指针在一定条件下，可以进行相等或者不等的运算
B.一个指针可以加上两个整数之差
C.两个指针在一定条件下可以相加
D.可以将一个空指针赋给某个指针

数组及指针

- 对于int *pa［5］;的描述，正确的是（）

A. pa是一个指向数组的指针，所指向的数组是5个int型元素
B. 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
C. pa［5］表示某个数组的第5个元素的值
***D.* 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**

数组及指针

- 要禁止修改指针p本身，又要禁止修改p所指向的数据，这样的指针应定义为（）

A. const char *p=“ABCD”；

B. char *const p=“ABCD”；

C. char const *p=“ABCD”；

D. const char * const p=“ABCD”；

数组及指针

- 设有语句“int fun(char*,int&); char str[100]; int k;”则对函数fun的正确调用形式是

A. fun(str,&k);

B. fun(str,k);

C. fun(str[100],k);

D. fun(str[100],&k);

数组及指针

• 设有如下定义:

int arr[]={6，7，8，9，10};

int * ptr;

则下列程序段的输出结果为（ ）。

ptr=arr;

* (ptr+2)+=2;

cout<<*ptr<<*(ptr+2)<<endl;

A.8，10　B.6，8　C.7，9　***D.6，10***

数组及指针

• 设有定义 int (*ptr)(); 则以下叙述中正确的是（ ）。

A. ptr是指向一维数组的指针变量

B. ptr是指向int型数据的指针变量

C. ptr是指向函数的指针，该函数返回一个int型数据

D.ptr是一个函数名，该函数的返回值是指向int型数据的指针

数组及指针

• 关于this指针的描述中，错误的是

A. this指针是指向对象的指针
B. this指针是在使用对象引用成员函数时系统自动生成的
C. this指针是指向成员函数的指针
D. this指针可以在程序中显示使用

数组及指针

• 关于this指针使用说法正确的是（）

A. 保证每个对象拥有自己的数据成员，但共享处理这些数据的代码
B. 保证基类私有成员在子类中可以被访问。
C. 保证基类保护成员在子类中可以被访问。
D. 保证基类公有成员在子类中可以被访问。

类的继承

- 类的继承，以及公有、私有、保护继承的基本语法
- 有继承关系时，构造和析构对象时构造函数与析构函数的执行次序
- 运算符重载的基本概念和语法
- 各种运算符重载为成员函数和非成员函数，注意参数、返回值等。
- 派生类与基类之间的类型转换规则，派生类与基类对象、指针、引用初始化、赋值、类型转换等。
- 虚函数与多态性、纯虚函数与抽象类、虚析构函数
- 用基类指针和引用访问派生类对象时的多态现象

类的继承

- 在公有继承的情况下，基类的公有或保护成员在派生类中的访问权限（）

A. 受限制

B. 保持不变

C. 受保护

D. 不受保护

类的继承

• 所谓多态性是指（）

A. 不同的对象调用不同名称的函数

B. 不同的对象调用相同名称的函数

C. 一个对象调用不同名称的函数

D. 一个对象调用不同名称的对象

类的继承

• 假定要对类AB定义加号操作符重载成员函数，实现两个AB类对象的加法，并返回相加结果，则该成员函数的声明语句为（ ）。

A. AB operator+(AB &a ， AB &b);

B. AB operator+(AB &a);

C. operator+(AB a);

D. AB & operator+( );

类的继承

• 派生类的构造函数的成员初始化列表中，不能包含

A.基类的构造函数

B.派生类中子对象的初始化

C.基类的子对象初始化

D.派生类中一般数据成员的初始化

类的继承

• 以下关于运算符说法正确的是

A. 所有的运算符都可以重载

B. C++中利用运算符重载办法可以创造新运算符

C. 根据需要在重载时可以提高重载运算符的优先级

D. 不能改变重载运算符的运算顺序

类的继承

• 下列函数中，纯虚函数是

A. void fun(int)=0;

B. virtual void fun(int);

C. virtual void fun(int){};

D. virtual void fun(int)=0;

类的继承

• 关于虚函数的描述，正确的是

A.虚函数是一个static类型的成员函数

B.虚函数是一个非成员函数

C.基类中说明了虚函数后，派生类中其对应的函数可不必说明为虚函数

D.派生类的虚函数与基类的虚函数具有不同的参数个数和类型

动态分配

- 若一个程序中使用如下语句申请了一个对象数组：point*prt=new point[2]；则在需要释放pn指向的动态数组对象时，所使用的语句是____delete [] prt___.

模板与流

- 函数模板（包括根据算法描述编写算法函数模板）
- 类模板
- 常用的文件流类及其成员函数的用法

模板与流

- 函数模板template<typename T>void Func(T， T)不能具有哪种实例化形式?

A．void Func(int,int)

B．void Func(bool，bool)

C．void Func(double,int)

D．void Func(char，char)

模板与流

- 下面说法正确的是

A. cin是一个类对象

B. cin是一个类的成员函数

C. cin是一个函数

D. cin对应的设备是鼠标

模板与流

- 语句ofstream f(“TEMP.DAT”,ios::app | ios::binary)；的功能是建立流对象f，试图打开文件TEMP.DAT 并与之连接，并且（）

***A.** 若文件存在，将文件写指针定位于文件尾；若文件不存在，建立一个新文件*

B. 若文件存在，将其置为空文件；若文件不存在，打开失败

C. 若文件存在，将文件写指针定位于文件首；若文件不存在，建立一个新文件

D. 若文件存在，打开失败；若文件不存在，建立一个新文件

期末寄语

这只是一个开始