# C++ based OOP final exam sample paper

**一、单选题，选择一个最优答案 (1x10 = 10 分)**

1．下列**不属于类的**成员函数的是(　　)

A．构造函数　　B．析构函数

C．友元函数　　D．拷贝构造函数

2．对于任意一个类，如果程序员不写任何构造函数，则编译器**不会添加**(　　)构造函数

A．无参数构造函数　　B．拷贝构造函数

C．赋值运算符函数　　D．输入输出运算符函数

3. 假设 PERSON 是一个类，get_instance_count(　)是该类的公有静态成员函数，Jack 是该类的一个对象，PERSON * p=&Jack，则下面用法**不正确**的是（　　）。

A．PERSON. get_instance_count()　B．PERSON::get_instance_count() C．p->PERSON:: get_instance_count()　D．p-> get_instance_count()

4. 公有派生类的成员函数体内可以访问基类的（　　）

A．公有成员和私有成员　　B．受保护成员和私有成员

C．公有成员和受保护成员　　D．公有成员、受保护成员和私有成员

5. 假定 Tom 为 CStudent 类的一个对象，则执行” CStudent Jack =Tom；”语句时将自动调用该类的(　　)

A．缺省构造函数　　B．转换构造函数

C．拷贝构造函数　　D．赋值运算符函数

6. 下面情况，**不会**调用拷贝构造函数的是(　)

A. 声明一个对象时，使用另一个对象初始化

B.对象作为函数的返回值

C.将对象作为实参，按传值方式进行函数调用

D. 用派生类对象给已经构造好的基类对象赋值

7. 关于多重继承的说法正确的是(　　)

A．多重继承的派生类可以有两个以上基类，可以增大代码的重用性，没什么弊端

B．没有虚基类时，基类构造函数的调用顺序取决于**基类被继承时的顺序**

C．基类构造函数的调用顺序由它们在**初始化列表中**的顺序决定

D．多重继承所产生二义性问题，无法消除，所以多个基类不能有相同的成员。

8．下列说法**不正确**的是(　　)

A．公有派生类的对象可以给基类对象赋值

B．基类类型的指针可以指向派生类的对象.

C．公有派生类的对象可以用作基类的对象使用

D．公有派生类类型的指针可以指向基类的对象。

9． 函数重载与函数模板的说法不正确的是(　　)

A．当函数的函数体不同但功能相同时，可以设计为重载函数

B．当函数体相同，仅仅操作的数据类型不同时，可设计为函数模板

C．重载函数和函数模板都是抽象的，都需实例化

D．函数模板是有待于实例化为模板函数，重载函数可以直接使用

10． 动态绑定与静态绑定的说法不正确的是(　)

A．静态绑定在程序的编译阶段就已确定，动态绑定在程序运行期间确定

B．动态绑定是通过继承、虚函数及指针来实现的，它灵活性大

C．静态类型用于编译程序检查类型的合法性，动态类型用于是运行时动态绑定

D．动态绑定效率高，灵活性大，要尽可能的使用

**二、填空题 (1x10 = 10 分)**

1. 在 C++中，使用_____分配的内存, 使用 delete 进行释放.

2. 在类模板中, template 关键字后的尖括号内的类型参数都要冠以保留字_____或_____。

3. 当用 private 方式派生一个类时，基类的 public 成员成为派生类的_____成员，protected 成员成为派生类的_____成员，派生类的成员函数不能访问基类中_____成员。

4. 每个类都有一个隐含的_____指针, 该指针指向当前对象的首地址。

5．class CComplex 重载前缀运算符++的函数原型是_____

6. 重载函数是靠_____来区分的，函数的_____和参数传递方式是不能区分重载函数的。

7．编译时的多态性通过_____实现，称作静态多态。

8．为了实现动态多态，派生类需要重新定义基类的_____

9．如果一个类中含有_____，则该类称为抽象类，这样的类不能直接创建实例，只能做其他类的基类。

10．C++中，缺省是复制继承，采用_____可以实现共享继承.

**三、改错题，指出错误，并改正之(5x3 =15)**

1．
```
class CComplex{
        int m_real(0),     m_image=0;
public：
        CComplex (int real = 0,int image) {m_real = real; m_image = image;}
}
CComplex   x(2,3),   y(4);
```

2.
```
#include <iostream>
using namespace std;

class CPerson {
```

```
public:
    void set_age(int x)  {
        age= x;
    }
    int get_age()   {
        return age;
    }
protected:
    int age;
};
class CWorker: public CPerson{
public:
    void set_salery(int s)     {
        selary = s;
        }
protected:
    int salery;
private:
    void set_age(int x) { }
};

int main(){
     CWorker obj;
     obj.set_age(15);
     obj.set_selary(1000);
     cout << obj.get_age() << "\n";
     return 0;
}
```

3. `#include <iostream>`

```
using namespace std;

float abs(float x){
    return (x >= 0 ? x : -x);
}
double abs(double x){
    return (x >= 0 ? x : -x);
}
int main(){
    cout << abs (3.14) << "\n";
    cout << abs (-5) << "\n";
    return 0;
}
```

**四、程序填空题(6x3=18)**

1. #include <iostream.h>
class CTiger{
_____
_____
};
_____
int main()
{
cout<<"Grow"<<endl;
return 0;
}
在横线处填上合适的语句,不改变其他语句，使程序运行时输出：
Born
Grow
Dead

2．template _____
class Array{
protected:
    T * m_pData;

Array( int n=10) {
if( n<=0 )
    m_pData = NULL;
else
    m_pData = new T [n];
}
~Array() {_____}
};
在横线处填上代码，完成此模板类。

3．#include <iostream.h>
class CPserson {
public:
_____work (  ) =0;
};
class CTeacher :public CPerson
{
public:
void work(){_____}
};

```
class CStudent: :public CPerson
{
public:
void work(){_____}
};
int main(){
CPerson * pPerson = NULL;
CTeacher TeacherWang；
CStudent John；
pPerson = & TeacherWang;
pPerson->work();

pPerson = & John;
pPerson->work();
}
```

在横线处填上适当代码，使程序运行结果如下(不包括引号)：

“教学科研”

“上课学习”

**五、理解题，写出下面程序的输出结果(6x3=18 分)**

1.

```
#include<iostream>
using namespace std;

class A {
     public:
       void print_classname( ) { cout << "class A\n ";    }
       virtual ~A(){cout<<”~A();\n”;}
};
class B: public A {
   public:
          B(){ cout<< “B()”<<end;}
          void print_classname( ) { cout << “class B \n”;    }
          virtual ~B(){cout<<”~B( );\n”;}
};
void main( )
{
   A *pA =new    B( );
   pA->print_calssname( );
  delete pA;
}
```

2.

```
#include <iostream>
```

```
using namespace std;

class BASE{
public:
    BASE(){ cout<<”BASE constructor is active\n”;}
    ~BASE(){cout<<”Base destructor is active\n”;}
}

class DEMO_CLASS:public BASE {
public:
    DEMO_CLASS(int i);
    ~DEMO_CLASS();
};

DEMO_CLASS::DEMO_CLASS(int i){
    cout << "Initial value is " << i << "\n";        return;
}

DEMO_CLASS::~DEMO_CLASS(){
    cout << "Goodbye!\n";          return;
}

int main(){
    DEMO_CLASS obj(30);
    cout << "This is the end of main().\n";
    return 0;
}
```

**六、简答题：**（5*4=20 分）

1．为什么要引入引用概念，使用引用有何好处？引用经常被用在什么场合？

2．举例说明继承和组合的概念及使用场合。

3．什么情况需要编写拷贝构造函数和赋值运算符，两个主要区别是什么？

**七、写程序题(9 分)**

1. 写一个完整的 Complex 类，实部和虚部均为 double 类型，该类支持带参数构造或者无参数构造，无参数构造， 实部和虚部均为 0.0；要求 (a)写出头文件及该类的声明和实现. 包括构造函数，及在主函数中的使用示例 (b) 成员变量为 private (c) 该类支持加法和减法运算. (d) 重载输出运算符”<<”，Complex obj; cout <<obj; 输出个格式为“(0, 0)”.