1. 可以建立指针的引用，但不能创建指向引用的指针

int &\*ap=a; （错ap是指向引用的指针） int \*&pr=pi; （对pr是指针的引用）

1. 重载函数必须具有不同的参数个数或不同的参数类型，不以返回值和形参名来区分函数
2. 函数原型中所有取默认值的参数出现右边int add(int i,int k,int j=5);
3. 调用具有默认参数值的函数时，若某个实参默认而省略，则其右面的所有实参皆应省略而采用默认值。int add(int x=7,int y=2,int z=11); 调用add(5,6)后x=5，y=6，z=11
4. 当函数的重载带有默认参数时，要注意避免二义性。例double add(double x,double y=2.2); 和 double add(double x);是错误的，若add(2.5)时，编译器将无法确定调用哪一个函数。
5. 引用参数传递的是实参的地址，因此也是一种地址传递，能够达到与指针同样的效果（The reference parameter passes the address of the argument, so it is also an address passing that can achieve the same effect as the pointer）
6. 当函数返回一个引用时，return语句只能返回一个变量，而不能返回一个表达式
7. 作用域讨论的是标识符的有效范围；可见性是讨论标识符是否可以被使用
8. 函数原型作用域是C++程序中最小的作用域（function prototype）
9. 析构函数没有参数、没有返回值，而且不能重载
10. 静态成员函数没有this指针



1. 引用被创建的同时必须被初始化；不能有NULL引用
2. 对象的赋值只对其中的数据成员赋值，不对成员函数赋值；类的数据成员中不能包括动态分配的数据
3. 若函数的形参为类对象，调用函数时，实参赋值给形参，系统自动调用拷贝构造函数。
4. 当函数的返回值是类对象时，系统自动调用拷贝构造函数
5. 静态数据成员初始化在类体外进行，而且前面不加static
6. 同普通成员函数一样，静态成员函数可以在类内定义，也可以在类外定义。在类外定义时，和普通成员函数的定义格式相同，而不要使用static前缀
7. 友元不是该类的成员函数，但是可以访问该类的私有成员
8. friend void modifyDate(Date& date,int year,int month,int day); //声明类Date的友元函数

void modifyDate(Date& date,int year,int month,int day) { //友元函数定义

date.year=year;

date.month=month;

date.day=day;

}

1. 类的成员函数是友元函数

class Student; //类的提前声明

class Teacher{

public:

void SetScore(Student&,double); //修改指定学生成绩

……

};

class Student ……

friend void Teacher::SetScore(Student &stu,double s); //声明为友元函数

};

1. 若A类声明为B类的友元类，那么，A类中的每一个成员函数都可以访问B类中的任何类型的成员
2. 常量成员函数不能更新类的数据成员的值，也不能调用该类中没有用const修饰的成员函数，只能调用常量成员函数；const关键字可以用于对重载函数的区分
3. 常量数据成员通过初始化列表初始化；常量数据成员不能在函数内赋值
4. 常量指针pointer to constant：int \* const w=&x; 不能被改变指向(地址)，但值可以被改变。
5. 指向常量的指针A pointer to a constant：const int \*w；不能改变值但是可以改变指向
6. int (Point::\*p\_GetX)()=Point::GetX; //使用成员函数指针访问成员函数
7. 对象间赋值(=)是一个（浅）拷贝过程
8. 虚基类L1中的构造函数带有形参，因此从虚基类L1中直接继承（类L2\_1、类L2\_2）或间接继承（类L3）的派生类，其构造函数的成员列表都要列出对虚基类L1构造函数的调用
9. 派生类的对象可以赋值给基类对象。如：Derived d; Base b; b=d;
10. 派生类的对象可以初始化基类的引用。如：Derived d; Base &br=d;
11. 派生类对象的地址可以赋给指向基类的指针。如：Derived d; Base \*pb=&d;
12. c++默认的关联方式是静态关联。
13. 不能对内置类型重载int operator+(int,int);// error
14. 不能重载的运算符只有5个 .(成员访问) .\* (成员指针访问) ∷ (域) sizeof ?: (条件)
15. 运算符重载作为成员函数，可以少写一个函数的参数。但表达式第一个参数(即运算符左侧的操作数)是一个类对象Complex operator+( Complex& b);
16. 重载成友元函数：friend bool operator >(String &string1, String &string2);
17. 必须重载为某个类的成员函数： =、[ ]、（）、->
18. 必须是普通函数或友元函数，不能为成员函数： >>、<<
19. 重载后置自增a++运算符时，多了一个int型的参数Time operator++(int)，增加这个参数只是为了与前置自增运算符重载函数有所区别，此外没有任何作用
20. friend ostream& operator<<(ostream &, 自定义类 &);
21. friend istream& operator>>(istream &, 自定义类 &);
22. 虚函数是重载的另一种表现形式（动态重载——更灵活的多态机制）
23. 只有成员函数才能声明为虚函数；构造函数不可以是虚函数；内联函数不能声明为虚函数
24. 虽然抽象类不能定义对象(或者说抽象类不能实例化)，但是可以定义指向抽象类对象的指针变量。
25. template语句和函数模板定义语句之间不允许有其他的语句
26. 模板类型并不具有隐式的类型转换Template types do not have implicit type conversions
27. 模板函数只能全特化，没有偏特化。而模板类是可以全特化和偏特化的。