1、（本题满分 15 分）

请简述 C++程序设计语言的设计理念、演化历程（包括主要的贡献者），并阐明 C 和 C++的关系。

答：设计理念：

"与可能出现的错误相比，更重要的是能做什么好的事情。

Programmer needs to be trusted！"

演化历程：

1、C 语言是贝尔实验室的 Dennis Ritchie 在 B 语言的基础上开发出来的。C 语言作为 UNIX 操作系统的开发语言而广为人知；Dennis Ritchie 和 Ken Thompson 并称为 C 语言之父；

2、为了满足管理程序的复杂性需要，1980 年，贝尔实验室的 Bjarne Stroustrup 开始对 C 语言进行改进和扩充，形成了"C with class"；

3、1983 年 Rick Mascitti 将其正式命名为"C++"；

4、1994 年制定了 ANSI C++标准的草案；

C 与 C++的关系：

1、C++完全包含 C 语言成分（C 的超集），C 是建立 C++的基础，同时 C++还添加了 OOP 的完全支持；

2、C++ 支持 C 所支持的全部编程技巧；

3、任何 C 程序都能被 C++ 用基本相同的方法编写，并具备相同的运行效率和空间

得分

2、（本题满分 10 分）

请简述 C 与 C ++混合编程时要注意的问题。

答：

1、因为 C++是 C 的超集，且 C 是结构化编程语言，而 C++支持面向对象编程语言，所以在混合编程时，不应当出现 class 等面向对象的关键字；

2、若要调用 C 语言库中的函数，要附加关键字"extern "C" "；

3、C 语言不支持函数重载。在 C++中 f (int, int) 与 f(int, double) 是不同的函数，都重载了函数 f()；但是在 C 语言中却被认为是相同的函数。因为在编译时，C 语言给这几个函数的命名为 f_；而 C++命名分别为 f_int_int, f_int_double, f_，以表示区别；所以混合编程时应注意重载函数的问题；

4、表达式的求值顺序与副作用。不同的编译器表达式的求值顺序也不同，由此可能带来副作用问题；

3、（本题满分 10 分）

inline 函数的作用？随意使用所可能导致的问题？并请阐述合理使用的建议。

答：作用：

1、提高程序的可读性；

2、提高程序的运行效率；

3、可以弥补宏定义不能进行类型检查的缺陷。

导致的问题：

1、增大目标代码，因为在调用时，必须在调用该函数的每个文本文件中定义；

2、病态换页；

3、降低指令快取装置的命中率；

建议：

1、仅对使用频率较高的小段代码使用内联，一般行数应不超过 5 行；

2、将内联函数的定义放到头文件中；

3、在内联函数中不能含有负复杂的结构控制语句，如 switch、while 等；

4、递归函数不能用来做内联函数；

得分

4、（本题满分 10 分）

请给出 C++语言中关键字 const 的几种使用方法，并给出示例？

答：const 可以用来修饰数据类型，指针，函数，对象等；

1、当 const 修饰数据类型时，表明该数据类型在程序运行的过程中，值不应当被改变；如 const int a = 3，表示 int 型的数据 a 在程序运行开始到结束，其值一直是 3，不能被改变。并且，在使用 const 修饰数据类型是，声明的时候就必须进行初始化；

2、当 const 修饰指针时，有 3 种不同的用法：“指针常量，常量指针，常量指针常量”

1) 指针常量：const Type *p = (Type&)a;

Ø 指针可修改；

Ø 所指内容不可修改，因为所指的内容是常量；

2) 常量指针：Type const *p = (Type&)a;

Ø 指针不可修改，因为该指针是常量；

Ø 所指内容可以修改；

3) 常量指针常量：const Type const *p = (Type&)a;

Ø 指针和所指内容均不可修改，因为二者都是常量，并且在声明时就必须初始化；

3、当 const 修饰函数时，有 2 种不同的用法：“Type f() const{}，Type f(const T & t){}”；

1) 常成员函数 Type f() const{}：表明程序员告诉编译器不会修改常量值；

2) 拷贝构造函数中的常量参数 Type f(const T & t){}：该用法长出现在拷贝构造函数中，其中 const 可有可无，主要目的是希望不要改变对象 T 的实例 t 中的任何内容；

4、当 const 修饰对象时，主要是修饰对象中成员变量：如：

Class T{ … … const int a; … …};

因为在类的成员变量声明时，不应该初始化变量的值，而常量在运行时刻就已经拥有数值且不能被改变。所以，应当在构造函数中使用成员初始化表来对常量成员变量进行初始化。

得分

5、（本题满分 15 分）

什么是纯虚函数、虚函数和非虚函数？合理定义三种成员函数所应遵循的基本原则？请给出你认为合理定义的一个实例，并说明。

答：三者的定义：

非虚函数：即一般的成员函数；

虚函数：即在成员函数前加关键字 virtual 来标志；

纯虚函数：是指被表明为不具体实现的虚成员函数，形式为在虚函数后加“=0”。、

基本原则：

使用虚函数的限制

类的成员函数才可以是虚函数

静态成员函数不能是虚函数

内联成员函数不能是虚函数

构造函数不能是虚函数

析构函数可以（往往）是虚函数

注意事项：

不要定义与继承而来的非虚成员函数同名的成员函数

绝对不要重新定义继承而来的缺省参数值

三者的关系为：

纯虚函数

Ø 只有函数接口会被继承

Ø 子类必须继承函数接口

Ø （必须）提供实现代码

一般虚函数

Ø 函数的接口及缺省实现代码都会被继承

Ø 子类必须继承函数接口

Ø 可以继承缺省实现代码

非虚函数

Ø 函数的接口和其实现代码都会被继承

Ø 必须同时继承接口和实现代码

```
class Shape{
public:
Shape(){}
virtual ~Shape() = 0;                    //虚函数
virtual double area(){  return 0.0;  }          //纯虚函数
double getWeight(){  return  weight;  }      //非虚函数
private:
double area;
```

```
double weight;
};
```

得分

6、（本题满分 20 分）

在 C++程序设计中，可以利用析构函数防止资源泄露，请给出模板 auto_ptr 的基本定义、实现，以及应用实例。

- 答： Solution
  - Smart pointers
    - Template <class T>

```
class auto_ptr
{   public:
        auto_ptr(T *p=0):ptr(p) {}
      ~auto_ptr() { delete ptr; }
        T* operator->()  const { return ptr;}
         T& operator *()  const { return *ptr; }
    private:
        T*   ptr;
};
void processAdoptions(iostream& dataSource)
{   while (dataSource)
    {   auto_ptr<ALA>   pa(readALA(dataSource));
        pa->processAdoption();
    }
```

}

- 【小动物收养保护中心】
  - 收养中心每天产生一个文件，包含当天的收养个案信息
  - 读取这个文件，为每个个案做适当的处理

7、（本题满分 20 分）

请阐述在面向对象程序设计语言中引入构造函数机制的原因，并请给出控制一个类创建实例个数的手段（举例说明）。

答：原因：

1、类的封装性，体现在一部分数据是不能让外界访问的，所以初始化类的成员变量应该有类的成员函数来完成；

2、在构造对象实例时，希望由系统自动调用，而不是人为的去调用，同时根据类的唯一性和对象的多样性，则使用类名作为构造函数的名字；

示例：

利用静态成员可以被所有该类的对象所共享的性质来控制该类创建实例的个数。

```
class singleton{
protected:
singleton(){ }
    singleton(const singleton &);
public:
static singleton * instance(){
     return m_num == 0? m_instance = new singleton: m_instance;
}
    static void destroy() { delete m_instance; m_instance = NULL; }
private:
static singleton * m_instance;
Static int m_num;
};
singleton * singleton ::m_instance= NULL;
```

源地址：http://blog.renren.com/GetEntry.do?id=705798623&owner=316565492