20190114

- 什么是友元？友元的存在形式有？友元有何特点？

> 友元是一个可以能够直接访问一个类中的private和protected成员的函数。
>
> 在类的定义中用friend声明了一个外部函数或其他类的成员函数(public和private均可）后，这个外部函数称为类的友元函数。
>
> ## 友元的存在形式有
>
> **类作为友元**
>
> 友元函数分为友元全局函数和友元成员函数。
>
> 类作为友元需要注意的是友元类和原始类之间的相互依赖关系，如果在友元类中定义的函数使用到了原始类的私有变量，那么就需要在友元类定义的文件中包含原始类定义的头文件。
>
> **类成员函数作为友元函数**
>
> **模板类友元**
>
> 1. 非模板友元
> 2. 约束模板友元：
>
> 即友元的类型取决于类被实例化时的类型。
>
> 3. 非约束模板友元：
>
> 即友元的所有具体化都是类的每一个具体化的友元。
>
> ## 友元特点
>
> **友元不能被继承**
>
> B是A的友元类，C是B的子类，推不出C是A的友元
>
> **友元不具有传递性**
>
> B是A的友元，C是B的友元，推不出C是A的友元
>
> **友元没有相互性只具有单项性**
>
> 类A作为类B的友元时，类A称为友元类。A中的所有成员函数都是B的友元函数，都可以访问B中的所有成员。但是B中的函数不一定是A的友员，要看在类中是否有相应的声明。

- 运算符重载的原则是什么？有哪些规则

> 运算符重载为了使对用户自定义数据类型的数据的操作与内置的数据类型的数据的操作形式一致
>
> **运算符是一种通俗、直观的函数，比如：**

> int x = 2 + 3;
>
> 语句中的“+”操作符
>
> **重载操作符必须具有一个类类型或者是枚举类型的操作数**
>
> 不能是内置类型 eg：
>
> –int operator+(int, int);//不能重载
>
> **优先级和结合性是固定的**
>
> 原来是什么样还是什么样
>
> –操作符的优先级、结合性或操作数个数不能改变
>
> X == Y + Z;
>
> **不再具备短路求值特性**
>
> –重载操作符并不保证操作数的求值顺序 && ||
>
> （短路求值）——>(&& , ||)当（a&&b）a,b必须都为1,否则短路
>
> 不能臆造并重载一个不存在的运算符，
>
> –如@, #，$等。

- 不能重载的运算符有哪几个？

> 成员访问符 .
>
> 成员指针访问运算符 .* （在内部实现是会使用）
>
> 域运算符 ::
>
> 长度运算符 sizeof
>
> 条件运算符号 ?:

- 运算符重载的形式有哪几种？

> 1. 采用普通函数的重载形式
>
> 对于数据成员如果是public的，推荐采用普通函数重载
>
> 2. 采用友元函数的重载形式
>
> 普通函数作为友元
>
> 3. 采用成员函数的重载形式

- 自增自减运算符的区别？其形式是怎样的？返回值类型分别是什么

> **前置**++执行的返回值是引用（效率更高）
>
> **后置**++的返回值为对象：
>
> 后置的参数列表中多一个int ， 该 int 只是为了与前置分开来的标志位

- 实现String类的其它运算符的重载

```
kyle@ubuntu:String$g++ String.cc
kyle@ubuntu:String$./a.out
String (const char *)
s1 is : ykx
s2 is : ykx
String(const String & )
~String()
~String()
s3=s1+s2 : ykxykx
input s4 :
wmk
s4 is : wmk
s3 is not equal s4
s3 is higher than s4
~String()
~String()
~String()
~String()
kyle@ubuntu:String$
```