我们现在了解的数字，包括实数和复数，实数又包括有理数和无理数。本题要求设计一个通用数字类，两个通用数字可以判定是否相等。判定两个复数是否相等时，需要实部虚部均相等。判定一个复数与一个实数是否相等时，如果这个复数虚部为 0，实部与这个实数相等，则可以判定为是相等的。复数之间不可以比大小，但是实数之间可以比大小。根据以上需求，设计并实现以下几个类：

1. 抽象类 Number 通用数字类。 Number 类为以下几个类的基类，提供接口如下：

```
virtual std::string toString() const = 0 ;
virtual const double getRealPart() const = 0;
virtual const double getImaginaryPart() const = 0;
virtual bool operator==(const Number& number) const final;
virtual bool operator!=(const Number& number) const final;
```

其中==和!=运算符重载，实现方法为如果两个 Number 对象 getRealPart()的结果相等且 getImaginaryPart()的结果也相等，即可判定为两个对象相等，否则判定为不等。对于实数而言，getImaginaryPart()永远返回 0。

注意：由于 getRealPart()和 getImaginaryPart()的返回类型是 double，而两个 double x,y 判定是否相等不能简单的 x==y 来判断，而是当 x-y 的绝对值小于一个非常接近 0 的正小数，即可判定为相等。

2. 具体类 Complex 复数类。为 Number 类的派生类。作业 3 中的实现直接拿过来即可。修改类定义为 class Complex : public Number 。

3. 抽象类 Real 实数类。为 Number 类的派生类。类定义为 class Real : public Number 。首先实现 Number 类中的虚函数：

```
const double getImaginaryPart() const override final; //只返回 0
```

getImaginaryPart() 函数只需要返回 0 即可。

提供接口如下：

```
virtual bool operator<(const Real& real) final;
virtual bool operator<=(const Real& real) final;
virtual bool operator>(const Real& real) final;
virtual bool operator>=(const Real& real) final;
```

判定方法为使用 getRealPart() 取得两个 Real 对象的值，比较大小即可。

4. 具体类 Rational 有理数类。为 Real 类的派生类。教材 14 章例题直接拿过来即可。修改类定义为 class Rational : public Real。另外，增加函数实现：

```
const double getRealPart() const override;
```

实现方法为直接返回 doubleValue()的值。

5. 具体类 Irrational 无理数类。为 Real 类的派生类。类定义为 class Irrational : public Real 。有一个 private 的成员变量 double value，保存无理数的有限精度值。主要的 public 成员函数有：

```
Irrational(double value = 0);
const double getRealPart() const override;
std::string toString()   const override ;
```

其中 getRealPart()直接返回 value 的值，toString()将 value 值转换为字符串后返回即可。