# Lab6 实验文档

## 回答问题

**操作符重载和类方法有什么区别?**

- 运算符重载是将已经定义的、有一定功能的操作符进行重新定义，来完成更为细致具体的运算功能，类方法是定义全新的函数来完成运算功能。
- 运算符重载函数可以是类的成员函数，也可以是类的友元函数，还可以是既非类的成员函数也不是友元函数的普通函数。而类方法只能是类的成员函数。
- 重载运算符的目的是在最终实现中代码体现中更简洁易懂，但只能重载原本有的运算符。

**举例说明两者各自的应用场景**

例：分别用操作符重载和类方法实现自定义类的加法运算：

```
class Number{
    int val;
public:
    Number(int v)
    {
        val=v;
    }
    //操作符重载
    Number operator+(const Number &n)
    {
        Number temp;
        temp.val = this->val + n.val;
        return temp;
    }
    //类方法
    Number numberAdd(const Number &n)
    {
        Number temp;
        temp.val = this->val + n.val;
        return temp;
    }
};
```

调用

```
int main()
{
    Number a(10);
    Number b(20);
    Number c = a + b;           //操作符重载
    Number d = a.numberAdd(b);  //类方法
```

```
    return 0;
}
```

# 实现思路

## BigNum类

众所周知，int，long，long long 这些基本数据类型是有长度限制的。而字符串的上限非常大，用字符串存储数字，可以解决大数存储和计算的问题。 BigNum类中的私有成员变量str把大数以字符串的形式存储。构造函数只需把传参的字符串赋值给BigNum内的str即可。在这里每一位数字都是字符。

```
//class BigNum was designed to solve the problem of large number calculation
class BigNum
{
    //use string to store numbers
    string str;
    //class Operate was designed to perform a specified operation with BigNum on
the input file
    friend class Operate;
public:
    //no parameter, but it would not happen in current test
    BigNum();
    //use string to store numbers
    BigNum(string Str);
    //override four operators
    friend istream &operator>>(istream &input, BigNum &c);
    friend ostream &operator<<(std::ostream &output, BigNum &c);
    BigNum operator+(const BigNum &c);
    BigNum operator-(const BigNum &c);
};
```

**构造函数**

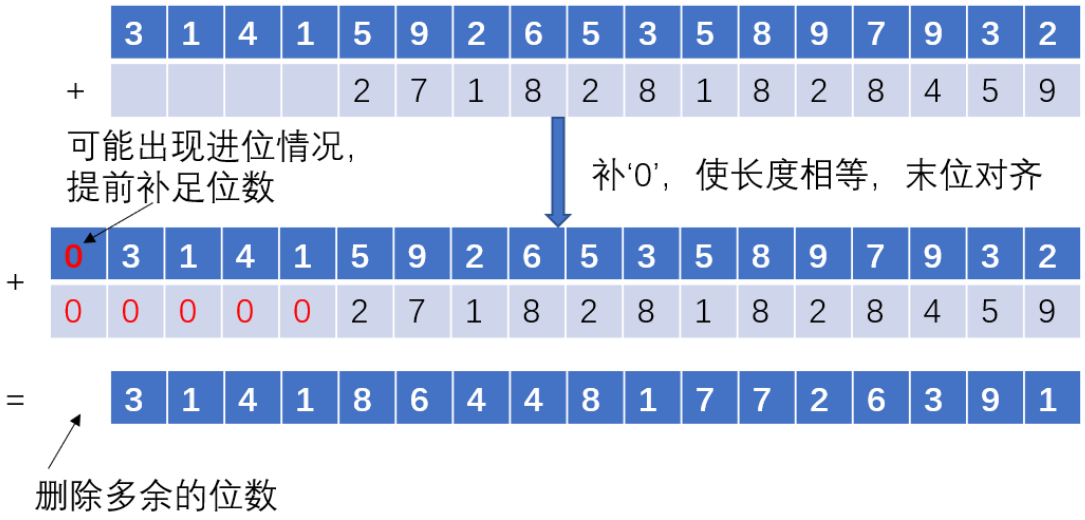
```
BigNum::BigNum()
{
    str = "0";
}
BigNum::BigNum(string Str)
{
    str = Str;
}
```

**加减法运算符重载：**

在这里每一位数字都是字符，不能像整型那样直接加减。因此我在这里采用逐位对齐相加减，满十进一的竖式计算思想。 需要注意的是，逐位进行加减的实际上用的是数字对应的ASCii码，相加后要减去一个'0'才能使计算结果为设想的数字；同理，相减后要加上一个'0'才能使计算结果为设想的数字。例如，'4' + '3' - '0' == '7' , '9' -

'5' + '0' == '4'.

例如：　31415926535897932384626433 + 2718281828459045




例如：　31415926535897932384626433 – 2718281828459045（大减小）


例如：　2718281828459045 – 31415926535897932384626433（小减大）


**输入、输出流运算符重载：**

输入：将数据读入到BigNum的str中。返回istream引用。

```
//override ">>" operator
istream &operator>>(istream &input, BigNum &c)
{
    input >> c.str;
    return input;
}
```

输出：将BigNum的str输出。返回ostream引用。

```
//override "<<" operator
ostream &operator<<(std::ostream &output, BigNum &c)
{
    output << c.str;
    return output;
}
```

## Operate类

实例化BigNum对象，根据读入文件中的运算符进行相应的运算，并输出到指定程序中。

```
//class Operate was designed to perform a specified operation with BigNum on the
input file
class Operate
{
public:
    //perform four kinds of specified operation
    void in_stream(ofstream &output);
    void out_stream(string &s, ofstream &output);
    void add_bn(string &a, string &b, ofstream &output);
    void substract_bn(string &a, string &b, ofstream &output);
};
```