Assignment_05 Stream

Source Code

DInteger.h

```
#ifndef DINTEGER_H
#define DINTEGER_H
#include<iostream>
using namespace std;

class DInteger{
public:
    DInteger();     // no-arg constructor
    DInteger(int one,int two);
    DInteger(const DInteger& sec_DInteger);     // Copy Constructors

    // Setters and Getters
    int getOne() const;
    void setOne(int one);
    int getTwo() const;
    void setTwo(int two);

    DInteger add(const DInteger& sec_DInteger) const;
    DInteger subtract(const DInteger& sec_DInteger) const;
    DInteger multiply(const DInteger& sec_DInteger) const;
    DInteger divide(const DInteger& sec_DInteger) const;

    DInteger(int one);      // constructor for type conversion

    // augmented assignment operators
    DInteger& operator+=(const DInteger& sec_DInteger);
    DInteger& operator-=(const DInteger& sec_DInteger);

    DInteger& operator++();     // prefix ++ operator
    DInteger operator++(int dummy);     // postfix ++ operator

    friend ostream& operator<<(ostream& out, const DInteger& dInteger); // stream Insertion
operator
    friend istream& operator>>(istream& in, DInteger& dInteger);    // stream Extraction operator

    const DInteger& operator=(const DInteger& dInteger);    // assignment operator
private:
    int one;
    int two;
};

// Define nonmember function operators for arithmetic operators
DInteger operator+(const DInteger& d1, const DInteger& d2);
DInteger operator-(const DInteger& d1, const DInteger& d2);
DInteger operator*(const DInteger& d1, const DInteger& d2);
DInteger operator/(const DInteger& d1, const DInteger& d2);

#endif
```

DInteger.cpp

```cpp
#include"DInteger.h"

DInteger::DInteger()    // no-arg constructor
{
}
DInteger::DInteger(int one,int two)
{
    this->one = one;
    this->two = two;
}
DInteger::DInteger(const DInteger& sec_DInteger)    // Copy Constructors
{
    this->one = sec_DInteger.getOne();
    this->two = sec_DInteger.getTwo();
}

// Setters and Getters
int DInteger::getOne() const
{
    return one;
}
void DInteger::setOne(int one)
{
    this->one = one;
}
int DInteger::getTwo() const
{
    return two;
}
void DInteger::setTwo(int two)
{
    this->two = two;
}

DInteger DInteger::add(const DInteger& sec_DInteger) const
{
    int one = this->one + sec_DInteger.getOne();
    int two = this->two + sec_DInteger.getTwo();
    return DInteger(one,two);
}
DInteger DInteger::subtract(const DInteger& sec_DInteger) const
{
    int one = this->one - sec_DInteger.getOne();
    int two = this->two - sec_DInteger.getTwo();
    return DInteger(one,two);
}
DInteger DInteger::multiply(const DInteger& sec_DInteger) const
{
    int one = this->one * sec_DInteger.getOne();
    int two = this->two * sec_DInteger.getTwo();
    return DInteger(one,two);
}
DInteger DInteger::divide(const DInteger& sec_DInteger) const
{
    int one = this->one / sec_DInteger.getOne();
    int two = this->two / sec_DInteger.getTwo();
    return DInteger(one,two);
}
```

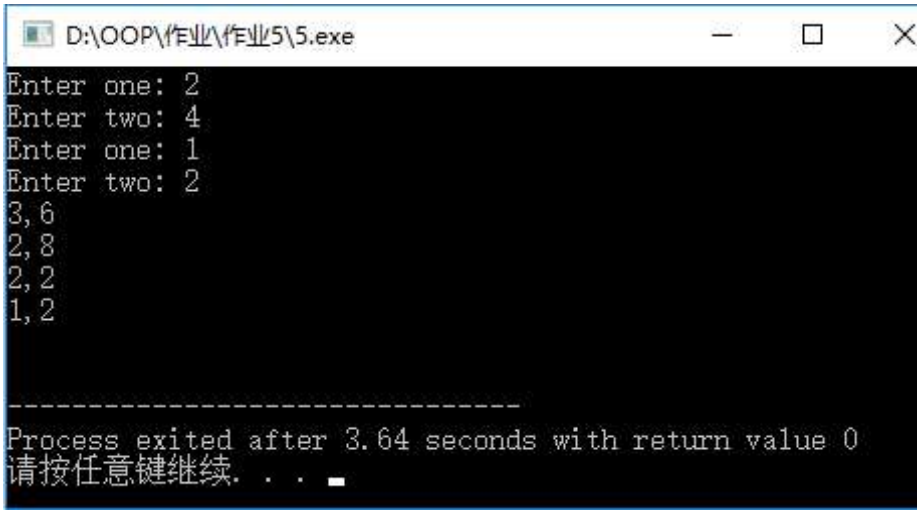
```
// constructor for type conversion
DInteger::DInteger(int one)
{
    this->one = one;
    this->two = 1;
}

// augmented assignment operators
DInteger& DInteger::operator+=(const DInteger& sec_DInteger)
{
    *this = add(sec_DInteger);
    return *this;
}
DInteger& DInteger::operator-=(const DInteger& sec_DInteger)
{
    *this = subtract(sec_DInteger);
    return *this;
}

DInteger& DInteger::operator++()    // prefix ++ operator
{
    ++one;++two;
    return *this;
}
DInteger DInteger::operator++(int dummy)    // postfix ++ operator
{
    DInteger temp(one, two);
    ++one;++two;
    return temp;
}

// stream Insertion operator
ostream& operator<<(ostream& out, const DInteger& dInteger)
{
    out << dInteger.getOne() << "," << dInteger.getTwo();
    return out;
}
// stream Extraction operator
istream& operator>>(istream& in, DInteger& dInteger)
{
    cout << "Enter one: ";
    in >> dInteger.one;
    cout << "Enter two: ";
    in >> dInteger.two;
    return in;
}

// assignment operator
const DInteger& DInteger::operator=(const DInteger& sec_DInteger)
{
    this->one = sec_DInteger.getOne();
    this->two = sec_DInteger.getTwo();

    return *this;
}

// nonmember function operators for arithmetic operators
DInteger operator+(const DInteger& d1, const DInteger& d2)
{
    return d1.add(d2);
```

```
}
DInteger operator-(const DInteger& d1, const DInteger& d2)
{
    return d1.subtract(d2);
}
DInteger operator*(const DInteger& d1, const DInteger& d2)
{
    return d1.multiply(d2);
}
DInteger operator/(const DInteger& d1, const DInteger& d2)
{
    return d1.divide(d2);
}
```

main.cpp

```
#include <stdlib.h>
#include"DInteger.h"

int main(){
    DInteger a, b, sum, mul, div, sub; //DInteger is a class, it is not simple int

    cin>>a;
    cin>>b;

    sum = a + b;
    mul = a * b;
    div = a / b;
    sub = a - b;
    a += 10;
    a -= 10;
    a++;

    cout<<sum<<endl;
    cout<<mul<<endl;
    cout<<div<<endl;
    cout<<sub<<endl;

    cout<<endl;

    return 0;
}
```

Sample Output


D:\OOP\作业\作业5\5.exe
Enter one: 2
Enter two: 4
Enter one: 1
Enter two: 2
3,6
2,8
2,2
1,2

Process exited after 3.64 seconds with return value 0
请按任意键继续. . .