



Sample Solution to Assignment 2, Problem 3

COURSE HOME

« [Back to Assignments](#)

Here are the contents of *rational.h*:

SYLLABUS

```
#ifndef _6S096_RATIONAL_H
#define _6S096_RATIONAL_H
```

CALENDAR

```
#include <stdint>
#include <iosfwd>
#include <stdexcept>
```

GETTING STARTED

LECTURE NOTES

```
class Rational {
    intmax_t _num, _den;
public:
    enum sign_type { POSITIVE, NEGATIVE };

```

ASSIGNMENTS



```
Rational() : _num{0}, _den{1} {}
Rational( intmax_t numer ) : _num{numer}, _den{1} {}
Rational( intmax_t numer, intmax_t denom ) : _num{numer}, _den{denom} { normalize(); }
```

RELATED RESOURCES

DOWNLOAD COURSE MATERIALS

```
inline intmax_t num() const { return _num; }
inline intmax_t den() const { return _den; }

void normalize();
float to_float()const;
double to_double()const;
sign_type sign() const;
Rational inverse() const;
};
```

```

std::ostream& operator<<( std::ostream& os, const Rational &ratio );

inline bool operator==( const Rational &lhs, const Rational &rhs ) {
    return lhs.num() * rhs.den() == rhs.num() * lhs.den();
}

inline bool operator<( const Rational &lhs, const Rational &rhs ) {
    if( lhs.sign() == Rational::POSITIVE && rhs.sign() == Rational::POSITIVE ) {
        return lhs.num() * rhs.den() < rhs.num() * lhs.den();
    } else if( lhs.sign() == Rational::NEGATIVE && rhs.sign() == Rational::NEGATIVE ) {
        return lhs.num() * rhs.den() > rhs.num() * lhs.den();
    } else {
        return lhs.sign() == Rational::NEGATIVE;
    }
}

inline Rational operator*( const Rational &a, const Rational &b ) {
    return Rational{ a.num() * b.num(), a.den() * b.den() };
}

inline Rational operator+( const Rational &a, const Rational &b ) {
    return Rational{ a.num() * b.den() + b.num() * a.den(), a.den() * b.den() };
}

inline Rational operator-( const Rational &a, const Rational &b ) {
    return Rational{ a.num() * b.den() - b.num() * a.den(), a.den() * b.den() };
}

inline Rational operator/( const Rational &a, const Rational &b ) {
    return a * b.inverse();
}

class bad_rational : public std::domain_error {
public:
    explicit bad_rational() : std::domain_error("Bad rational: zero denominator" ) {}
};

#endif // _G096_RATIONAL_H

```

Here is the source code file **rational.cpp**:

```

#include "rational.h"
#include "gcd.h"

```

```

#include <stdexcept>
#include <ostream>
#include <iostream>
#include <cmath>

Rational Rational::inverse() const {
    return Rational{ _den, _num };
}

Rational::sign_type Rational::sign()const {
    return _num >= 0 ? POSITIVE : NEGATIVE;
}

std::ostream& operator<<( std::ostream& os, const Rational &ratio ) {
    if( ratio == 0 ) {
        os << "0";
    } else {
        if( ratio.sign() == Rational::NEGATIVE ) {
            os << "-";
        }
        os << std::abs( ratio.num() ) << "/" << std::abs( ratio.den() );
    }
    return os;
}

void Rational::normalize() {
    if( _den == 0 ) {
        throw bad_rational();
    }

    if( _num == 0 ) {
        _den = 1; return;
    }

    auto g = gcd( std::abs( _num ), std::abs( _den ) );
    _num /= g; _den /= g;

    if( _den < 0 ) {
        _num = -_num;
        _den = -_den;
    }
}

```

```
float Rational::to_float() const {
    return static_cast<float>(_num) / static_cast<float>(_den);
}

double Rational::to_double()const {
    return static_cast<double>(_num) / static_cast<double>(_den);
}
```

Below is the output using the test data:

rational:

```
1: OK [0.007 seconds] OK! add
2: OK [0.006 seconds] OK! mult
3: OK [0.009 seconds] OK! add1024
4: OK [0.014 seconds] OK! add1024
5: OK [0.158 seconds] OK! add32768
6: OK [0.007 seconds] OK! op<<
7: OK [0.289 seconds] OK! div65536 in 0.280000 s
8: OK [0.006 seconds] OK! phi, 0.000000e+00
9: OK [0.006 seconds] OK! (Bad rational: zero denominator)
10: OK [0.006 seconds] OK! xyz
11: OK [0.007 seconds] OK! pow2
12: OK [0.006 seconds] OK! x1z
```

« [Back to Assignments](#)

FIND COURSES

- » Find by Topic
- » Find by Course Number
- » Find by Department
- » New Courses
- » Most Visited Courses
- » OCW Scholar Courses
- » Audio/Video Courses
- » Online Textbooks
- » Instructor Insights
- » Supplemental Resources
- » MITx & Related OCW Courses

FOR EDUCATORS

- » Chalk Radio Podcast
- » OCW Educator Portal
- » Instructor Insights by Department
- » Residential Digital Innovations
- » OCW Highlights for High School
- » Additional Resources

GIVE NOW

- » Make a Donation
- » Why Give?
- » Our Supporters
- » Other Ways to Contribute
- » Become a Corporate Sponsor

ABOUT

- » About OpenCourseWare
- » Site Statistics
- » OCW Stories
- » News
- » Press Releases

TOOLS

- » Help & FAQs
- » Contact Us
- » Site Map
- » Privacy & Terms of Use
- » RSS Feeds

OUR CORPORATE SUPPORTERS



» MIT Open Learning Library

» Translated Courses



ABOUT MIT OPENCOURSEWARE

MIT OpenCourseWare makes the materials used in the teaching of almost all of MIT's subjects available on the Web, free of charge. With more than 2,400 courses available, OCW is delivering on the promise of open sharing of knowledge. [Learn more »](#)



© 2001–2020
Massachusetts Institute of Technology



Your use of the MIT OpenCourseWare site and materials is subject to our [Creative Commons License](#) and other [terms of use](#).