**Shubham Garg**

**9919103057**

**Batch : F2**

**Lab Assignment**

**(4 May, 2020- 9 May, 2020)**

**Operator Overloading**

Question:

**Define a class ratio with numerator and denominator as member variables.**

**Write the constructors, destructors and copy constructor. Overload the**

**following operators for the same:**

**• Arithmetic operators: +, -, \*, /**

**• Relational operators: <=, >=, ==, !=**

**• Assignment operator: =**

**• Arithmetic assignment operator: +=, -=**

**• Pre and post increment operators: ++**

**• Stream input and output operators: << and >>**

#include<iostream>

using namespace std;

class Ratio

{

public:

int numerator,denominator;

Ratio(int num=1,int den=1)

{

numerator=num;

denominator=den;

}

~Ratio()

{

}

void print(){

cout<<numerator<<"/"<<denominator<<endl;

}

Ratio(Ratio &test)

{

numerator=test.numerator;

denominator=test.denominator;

}

Ratio operator\* (Ratio const & test)

{

Ratio fraction;

fraction.numerator = numerator \* test.numerator;

fraction.denominator = denominator \* test.denominator;

return fraction;

}

Ratio operator/ (Ratio const& test)

{

Ratio fraction;

fraction.numerator = numerator \* test.denominator;

fraction.denominator = denominator \* test.numerator;

return fraction;

}

Ratio operator + (Ratio const& test)

{

Ratio fraction;

int x,y;

x = numerator \* test.denominator;

y = denominator \* test.numerator;

fraction.numerator=x+y;

fraction.denominator=denominator\*test.denominator;

return fraction;

}

Ratio operator- (Ratio const& test)

{

Ratio fraction;

int x,y;

x = numerator \* test.denominator;

y = denominator \* test.numerator;

fraction.numerator=x-y;

fraction.denominator=denominator\*test.denominator;

return fraction;

}

int operator==(Ratio const& test)

{

int x=numerator/denominator;

int y=test.numerator/test.denominator;

if(x==y)

return 0;

else

return 1;

}

int operator<=(Ratio const& test)

{

int x=numerator/denominator;

int y=test.numerator/test.denominator;

if(x<=y)

return 0;

else

return 1;

}

int operator>=(Ratio const& test)

{

int x=numerator/denominator;

int y=test.numerator/test.denominator;

if(x>=y)

return 0;

else

return 1;

}

int operator!=(Ratio const& test)

{

int x=numerator/denominator;

int y=test.numerator/test.denominator;

if(x!=y)

return 0;

else

return 1;

}

int operator=(Ratio const& test)

{

numerator=test.numerator;

denominator=test.denominator;

}

Ratio operator += (Ratio const& test)

{

int x,y;

x = numerator \* test.denominator;

y = denominator \* test.numerator;

numerator=x+y;

denominator=denominator\*test.denominator;

}

Ratio operator -= (Ratio const& test)

{

int x,y;

x = numerator \* test.denominator;

y = denominator \* test.numerator;

numerator=x+y;

denominator=denominator\*test.denominator;

}

Ratio operator++()

{

numerator=numerator+denominator;

}

Ratio operator ++ (int)

{

numerator=numerator+denominator;

}

friend ostream & operator << (ostream &out, const Ratio &obj);

friend istream & operator >> (istream &in, Ratio &obj);

}; // End of Class

ostream & operator << (ostream &out, const Ratio &test)

{

out<<test.numerator<<"/"<<test.denominator;

return out;

}

istream & operator >> (istream &in, Ratio &test)

{

cout << "Enter numerator ";

in >> test.numerator;

cout << "Enter denominator ";

in >> test.denominator;

return in;

}