

#include <iostream>

using namespace std;

class Fraction{

public:

int num,den;

Fraction(int n=0, int d=0)

{

num=n;

den=d;

}

Fraction operator /(Fraction const &obj){

Fraction res;

res.num=num \* obj.den;

res.den=den \* obj.num;

return res;

}

void display1(){

cout<<num/den;

}

void display2(){

cout<<num<<"/"<<den;

}

void display3(){

cout<<"Error";

}

};

int main()

{

int a,b,c,d;

cin>>a>>b;

cin>>c>>d;

Fraction ob1(a,b), ob2(c,d);

Fraction ob3 = ob1/ob2;

if(ob1.den==0 || ob2.den==0){

cout<<"Error";

return 0;

}

if(ob3.den==1)

ob3.display1();

else{

for(int i=2;i<50;i++)

{

if(ob3.num%i==0 && ob3.den%i==0)

{

ob3.num=ob3.num/i;

ob3.den=ob3.den/i;

}

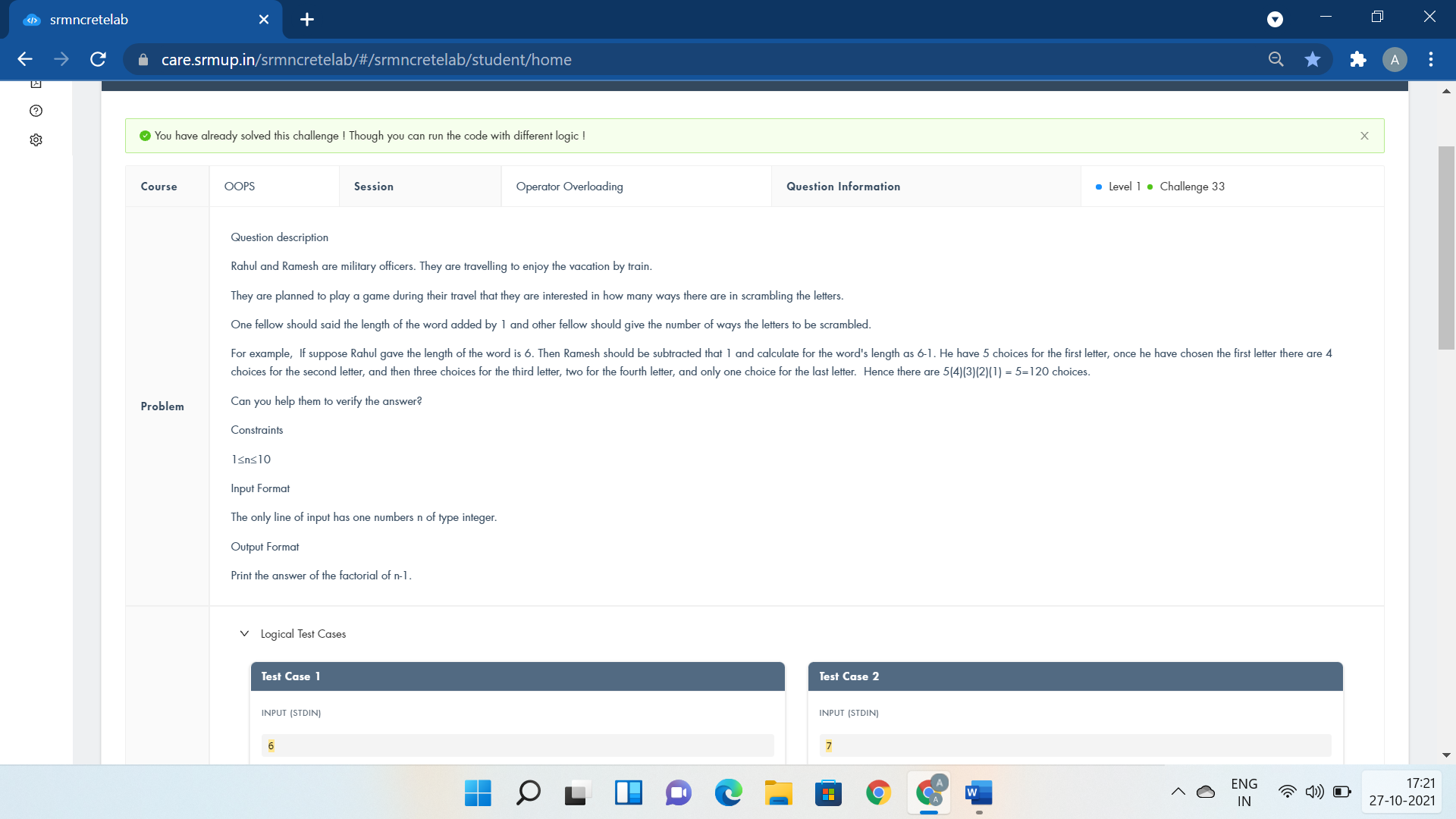
}

ob3.display2();

}

return 0;

}



#include <iostream>

using namespace std;

class Scrum{

public:

int n;

Scrum(int h)

{

n=h;

}

Scrum operator -- (int){

Scrum T(int h);

--n;

return 1;

}

void display(){

int res=1;

for(int i=1;i<=n;i++){

res=res\*i;

}

cout<<res;

}

};

int main()

{

int n;

cin>>n;

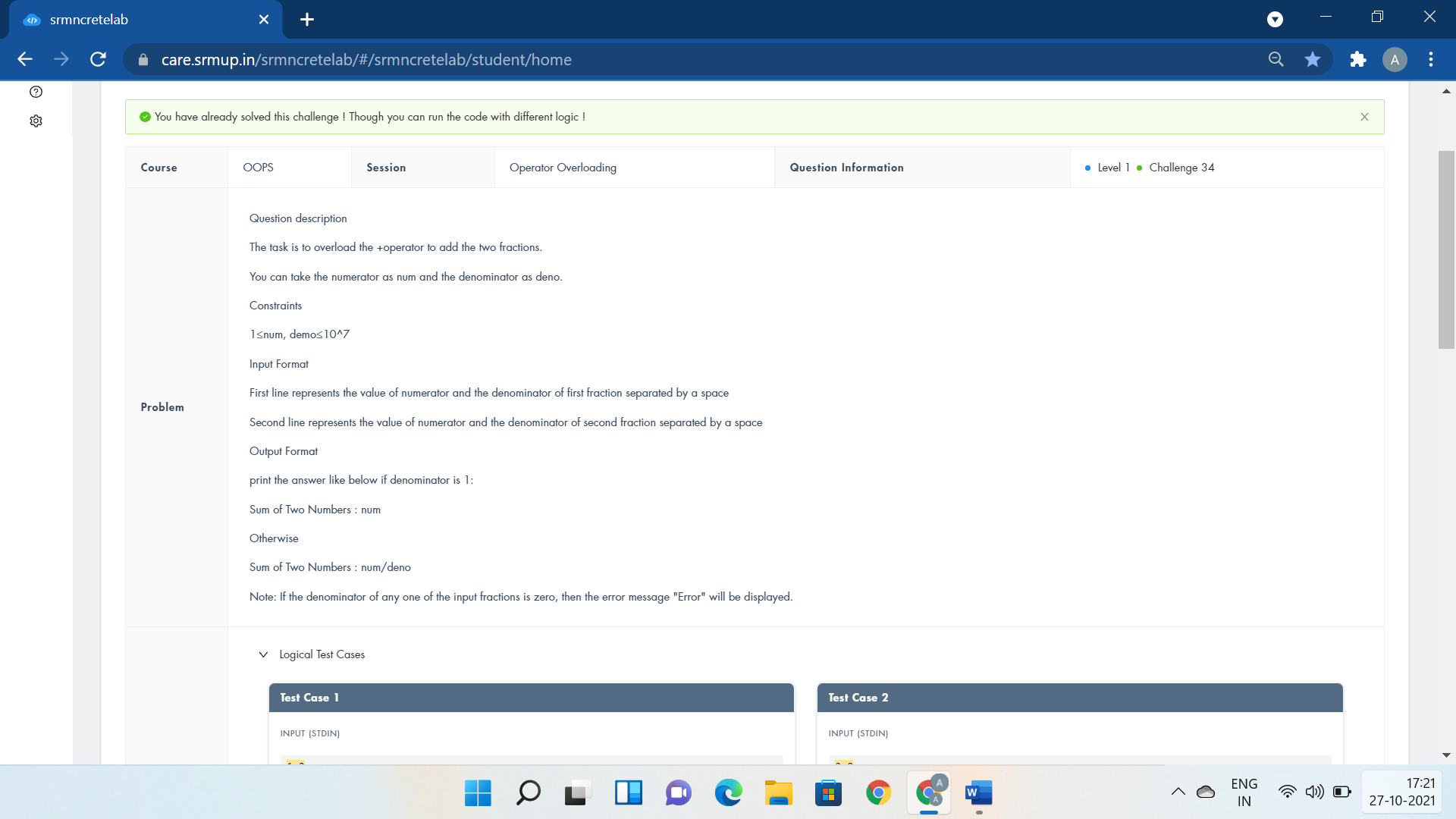
Scrum T(n);

T--;

T.display();

return 0;

}



#include<iostream>

using namespace std;

class Fraction

{

public:

int num,den;

Fraction()

{

num=0;

den=0;

}

void getinput()

{

cin>>num>>den;

}

Fraction operator +(Fraction obj)

{

Fraction temp;

temp.num=(num\*obj.den)+(den\*obj.num);

temp.den=den\*+obj.den;

return temp;

}

};

int main()

{

Fraction f1,f2,add;

f1.getinput();

f2.getinput();

add=f1+f2;

if(add.den==0)

cout<<"Error";

else if(add.num%add.den == 0)

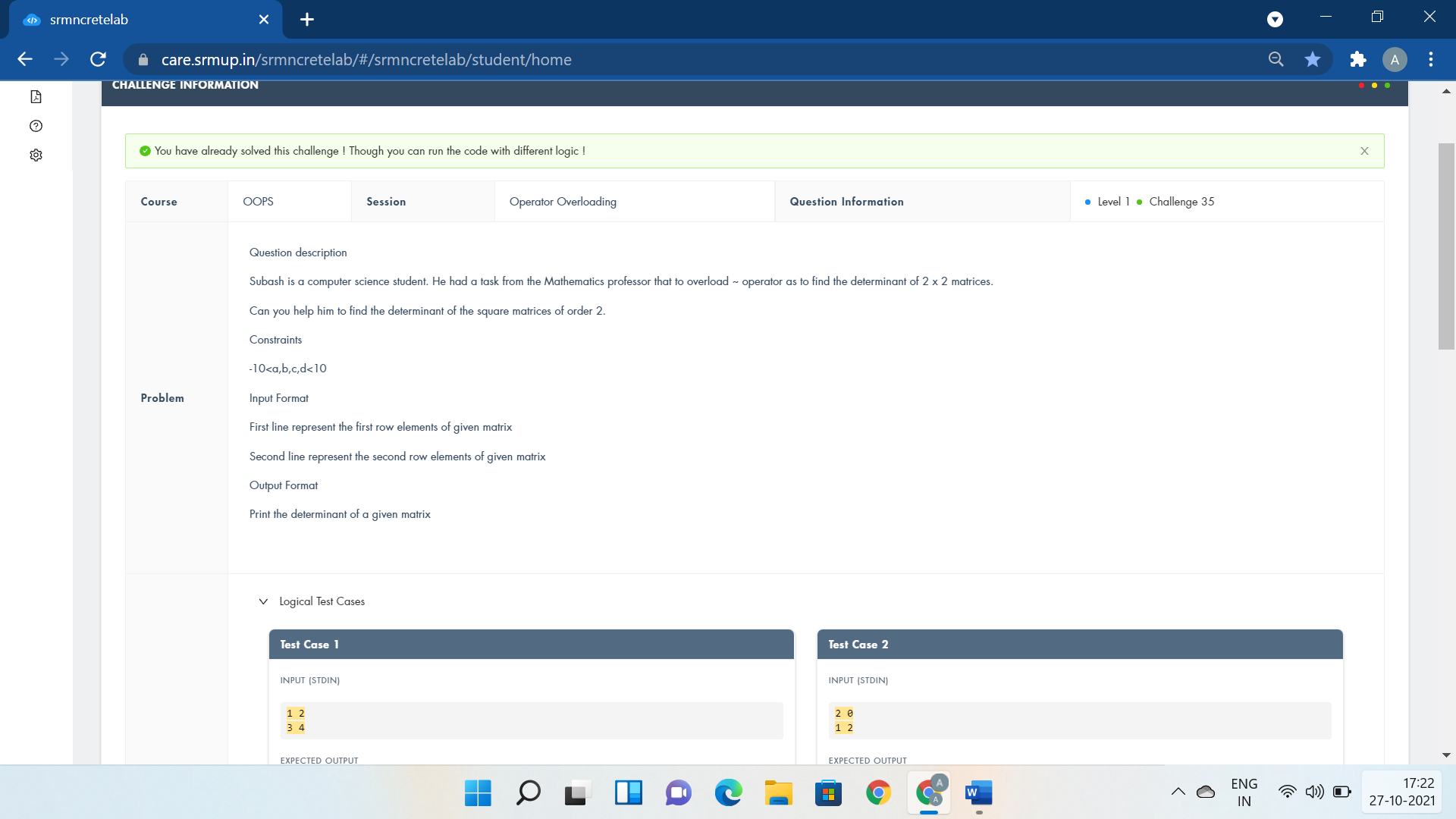
cout<<add.num/add.den;

else

cout<<add.num<<"/"<<add.den;

return 0;

}



#include <iostream>

using namespace std;

class matrix{

public:

int operator ~(){

int a,b,c,d;

cin>>a>>b>>c>>d;

return a\*d-b\*c;

}

};

int main()

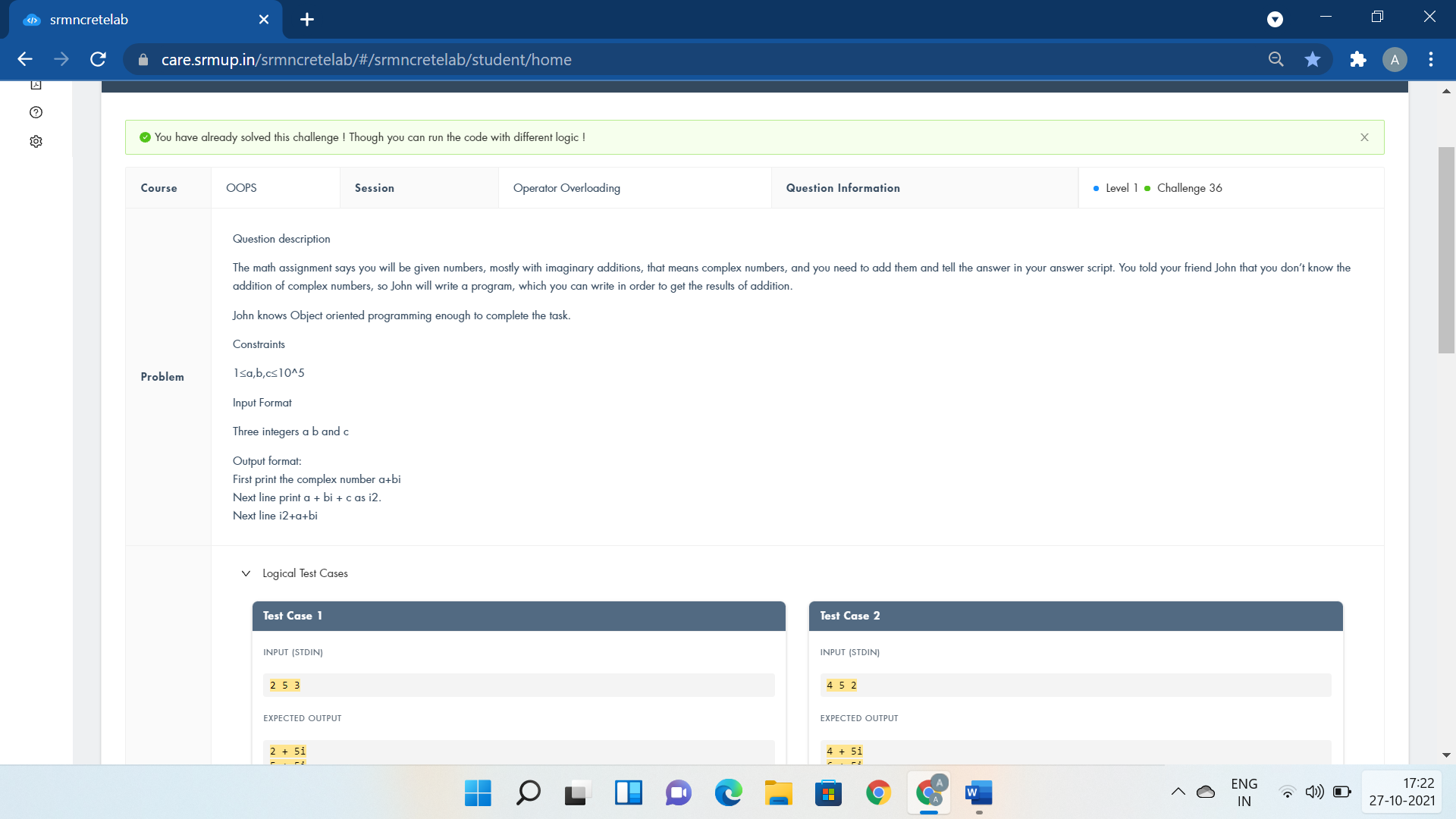
{

matrix t;

cout<<~t;

return 0;

}



#include<iostream>

using namespace std;

class Complex {

private:

int real, imag;

public:

Complex(int r = 0, int i =0) {real = r; imag = i;}

Complex operator+(int a) {

Complex res;

res.real = real + a;

res.imag = imag;

return res;

}

Complex operator+(Complex obj) {

Complex res;

res.real = real + obj.real;

res.imag = imag + obj.imag;

return res;

}

void print() { cout << real << " + " << imag <<"i"<< endl; }

};

int main()

{

int a,b,c;

cin>>a>>b>>c;

Complex i1(a, b);

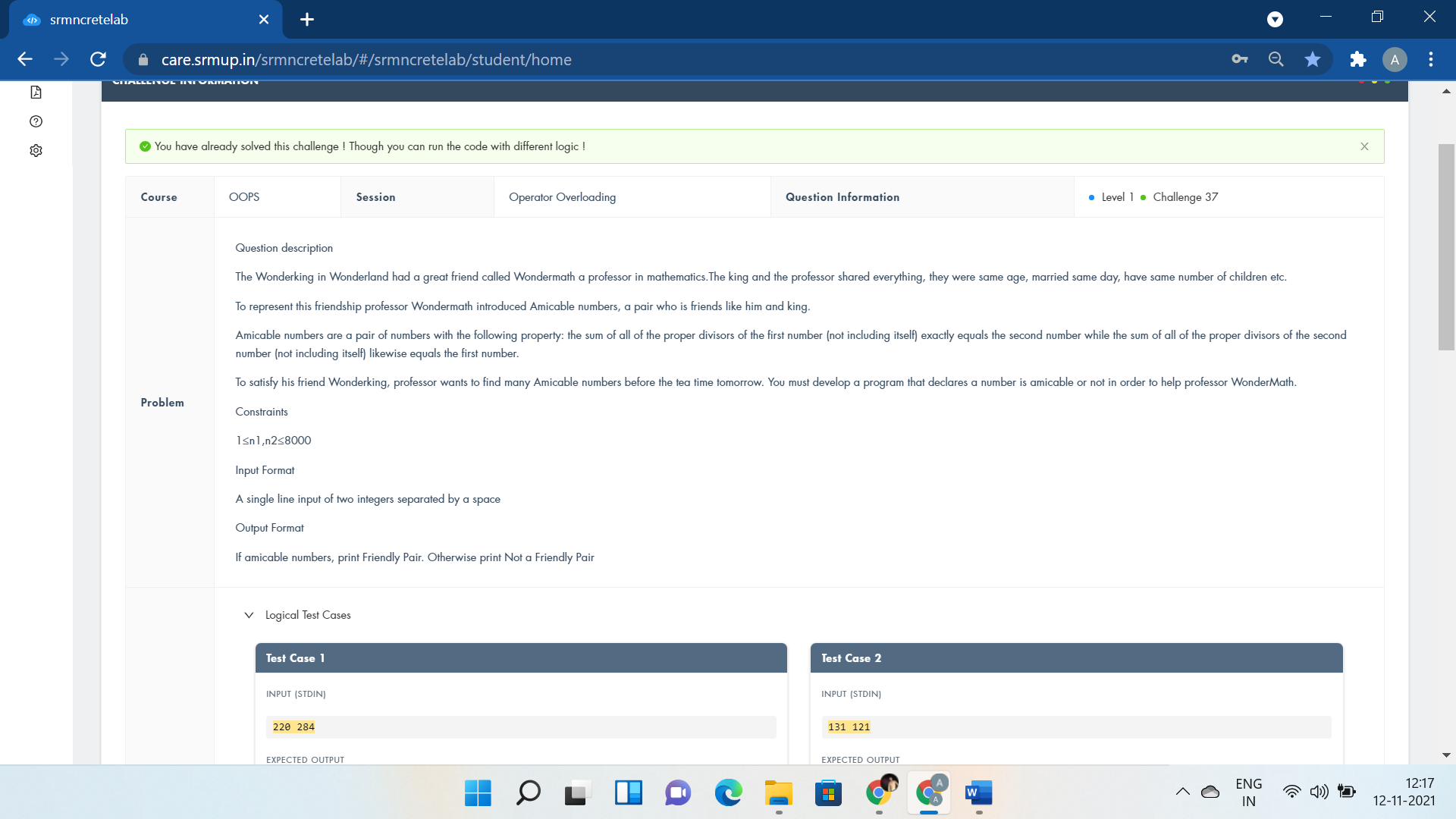
Complex i2 = i1 + c;

i1.print();

i2.print();

(i1+i2).print();

}



#include<iostream>

using namespace std;

class compare{

public:

int first,sum1=0;

compare(int x){

first=x;

}

void f(){

//first1=first;

for(int i=1; i<=first/2 ; i++)

{

//finding and adding divisors of first number

if(first%i==0)

sum1=sum1+i;

}

}

void operator ==(compare t2){

if(first==t2.sum1 && t2.first==sum1)

cout<<"Friendly Pair";

else

cout<<"Not a Friendly Pair";

}

};

//main program

int main()

{

int first,second;

//user input

cin>>first;

//user input

cin>>second;

compare t1(first),t2(second);

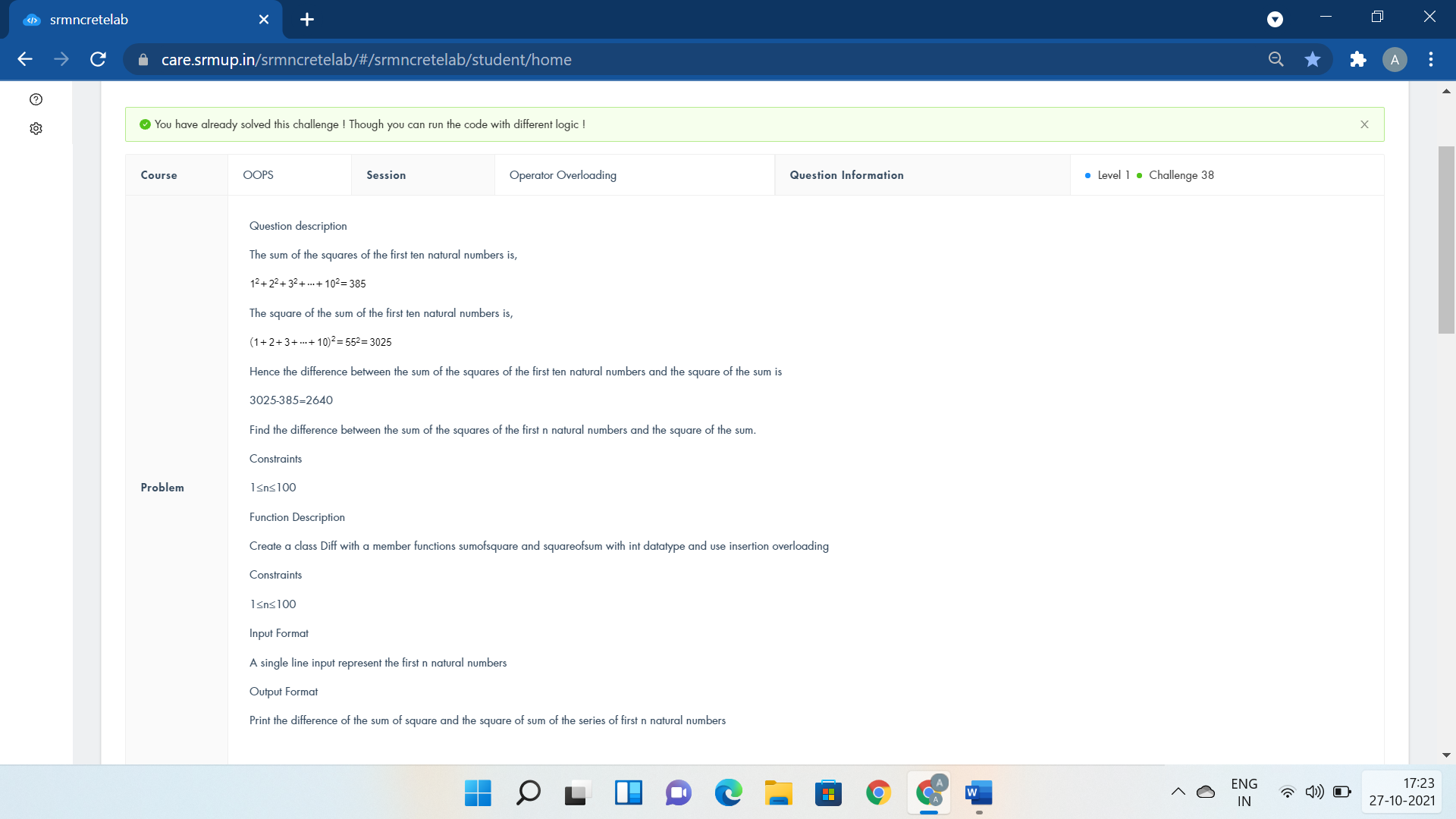
t1.f();

t2.f();

t1==t2;

return 0;

}



#include <iostream>

using namespace std;

class Diff{

public:

int n;

void getdata(){

cin>>n;

}

int sumofsquare();

int sumofnumsq(){

return n\*(n+1)\*(2\*n+1)/6;

}

};

int Diff :: sumofsquare(){

return n\*n\*(n+1)\*(n+1)/4;

}

int main()

{

Diff n;

if(0)

cout<<"friend void operator >> (istream &in, Diff &obj )";

n.getdata();

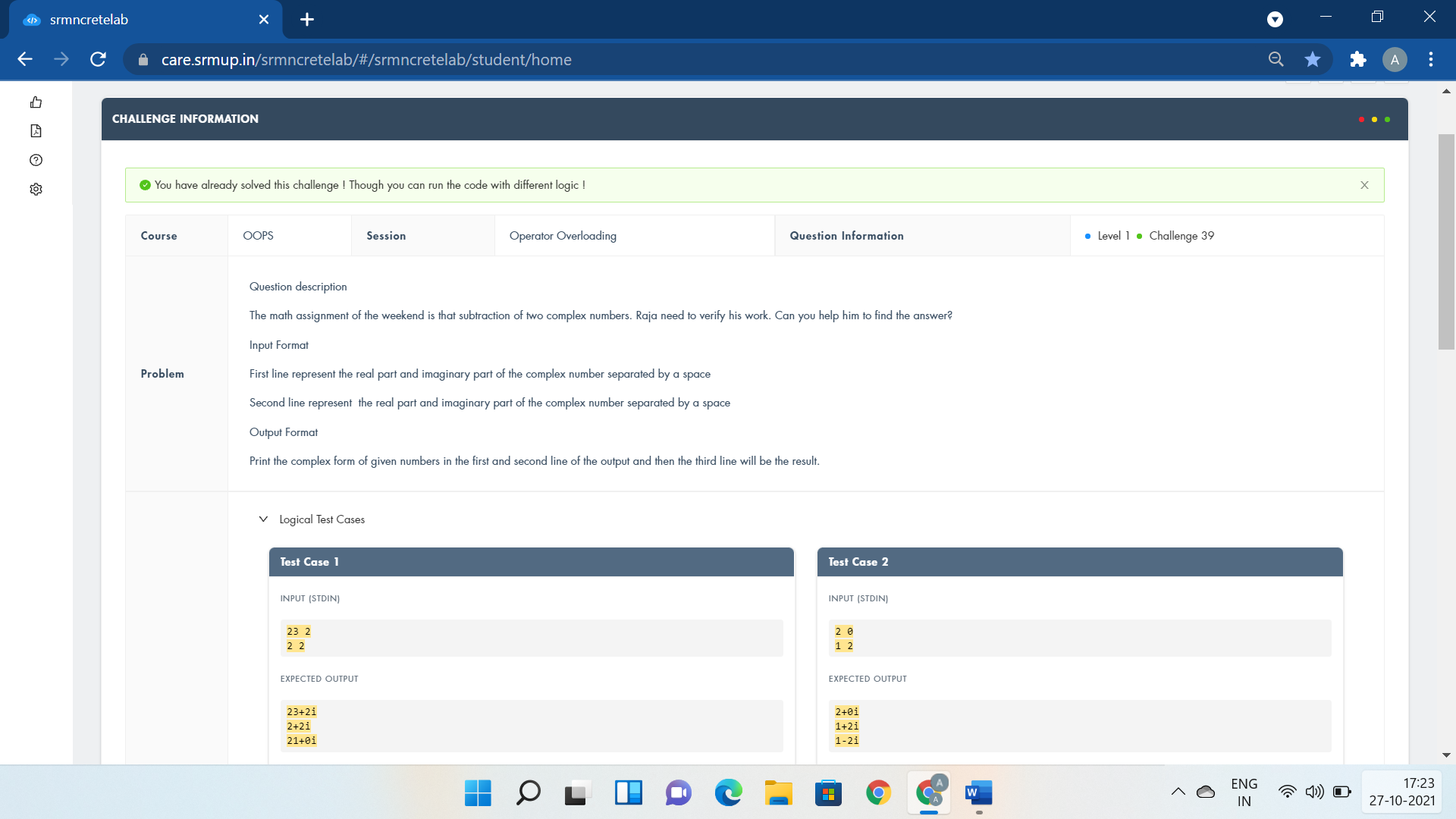
//int sq=n\*n\*(n+1)\*(n+1)/4;

//int sq2=n\*(n+1)\*(2\*n+1)/6;

cout<<n.sumofsquare()-n.sumofnumsq();

return 0;

}



#include <iostream>

using namespace std;

class complex

{

private:

float real;

float imag;

public:

complex() {cin>>real>>imag;}

complex operator-(complex ob)

{

complex t;

t.real = real - ob.real;

t.imag = imag - ob.imag;

return t;

}

void output()

{

if(imag < 0)

cout<< real << imag << "i"<<endl;

else

cout<< real << "+" << imag << "i"<<endl;

}

};

int main()

{

complex c1, c2;

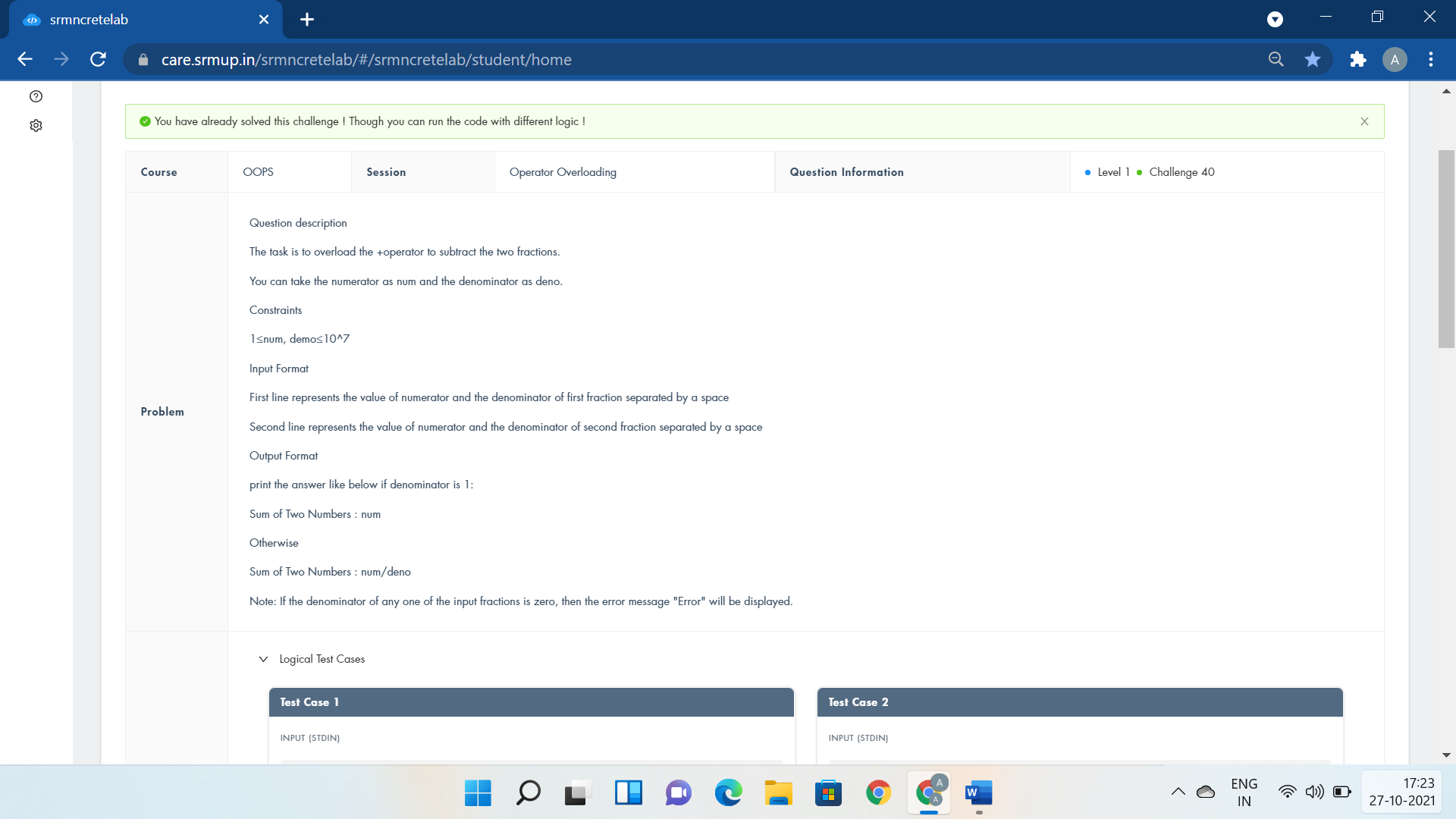
c1.output();

c2.output();

(c1 - c2).output();

return 0;

}



#include<iostream>

using namespace std;

class Fraction

{

public:

int num,den;

Fraction()

{

num=0;

den=0;

}

void getinput()

{

cin>>num>>den;

}

Fraction operator -(Fraction obj)

{

Fraction temp;

temp.num=(num\*obj.den)-(den\*obj.num);

temp.den=den\*+obj.den;

return temp;

}

};

int main()

{

Fraction f1,f2,add;

f1.getinput();

f2.getinput();

add=f1-f2;

if(add.den==0)

cout<<"Error";

else if(add.num%add.den == 0)

cout<<add.num/add.den;

else

cout<<add.num<<"/"<<add.den;

return 0;

}