//copyright by Carl Kevin Gasal (Winter2023)

//301242419 COMP123(Sec.012) Programming 2

using System.Runtime.CompilerServices;

class Complex

{

//Properties

public int Real {

get; private set;

}

public int Imaginary { get; private set; }

public double Modulus {

get => Math.Sqrt(Math.Pow(Real, 2) + Math.Pow(Imaginary, 2));

}

public double Argument {

get => 1 / Math.Tan(Real / Imaginary);

}

//Zero – this static property returns a new complex object with

//both the real and the imaginary parts equal to 0.

//The getter is public and of course there is no setter.

//Constructor

public Complex(int real, int imaginary)

=> (Real, Imaginary) = (real, imaginary);

//Methods

public override string ToString()

{

return $"({Real}, {Imaginary})";

}

public static Complex operator +(Complex lhs, Complex rhs) //plus operator or Addition

{

int real = lhs.Real + rhs.Real;

int imaginary = lhs.Imaginary + rhs.Imaginary;

return new Complex(real, imaginary);

}

public static Complex operator -(Complex lhs, Complex rhs) //minus operator or Substraction

{

int real = lhs.Real - rhs.Real;

int imaginary = lhs.Imaginary - rhs.Imaginary;

return new Complex(real, imaginary);

}

//additional task 1

public static Complex operator \*(Complex lhs, Complex rhs) //multiplication operator

{

//<a, b> \* <c, d> = <ac-bd, ad+bc>.

int real = (lhs.Real \* lhs.Imaginary) - (rhs.Real \* rhs.Imaginary);

int imaginary = (lhs.Real \* rhs.Imaginary) + (rhs.Real \* lhs.Real);

return new Complex(real, imaginary);

}

//additional task UNARY operator

public static Complex operator -(Complex c)

=> new Complex(-c.Real, -c.Imaginary);

public static Boolean operator ==(Complex lhs, Complex rhs)

=> (lhs.Real == rhs.Real && lhs.Imaginary == rhs.Imaginary);

public static Boolean operator !=(Complex lhs, Complex rhs)

=> (lhs.Real != rhs.Real && lhs.Imaginary != rhs.Imaginary);

}

class Program

{

static void Main(string[] args)

{

Complex c0 = new Complex(-2, 3);

Complex c1 = new Complex(-2, 3);

Complex c2 = new Complex(1, -2);

Console.WriteLine($"{c0}");

Console.WriteLine(c1);

Console.WriteLine(c2);

Console.WriteLine($"+ operator ~ {c1} + {c2} = {c1 + c2}");

Console.WriteLine($"- operator ~ {c1} - {c2} = {c1 - c2}");

Console.WriteLine($"\* operator ~ {c1} \* {c2} = {c1 \* c2}");

Complex c3 = c1 + c2;

Complex c4 = c3 - c2;

Console.WriteLine($"unary operator ~ -{c4} = {-c4}");

Console.WriteLine($"{c0} {((c0 == c1)? "equal" : "not equal" )} {c1}");

Console.WriteLine($"{c0} {((c0 == c2) ? "equal" : "not equal")} {c2}");

Console.WriteLine($"\n{c3} in polar form is {c3.Modulus:f2}cis({c3.Argument:f2})");

}

}