# Задание «Комплексное числов»

Реализуйте класс для работы с комплексным числами (ComplexNumber) удовлетворяющий следующим требованиям:

* Класс должен содержать два свойства типа double для хранения действительной и мнимой частей. Называться они должны Real и Imaginary.
* Класс должен иметь два конструктора. Первый принимает значение действительной (real) части (мнимая часть должна равняться 0). Второй принимает значение действительной (real) и мнимой (imaginary) части;
* Класс должен реализовывать операторы сложения (+), вычитания (-) и неявного преобразования типа из int в ComplexNumber;
  + Операторы «+»и «-» должны работать как с комплексными числами так и числами с плавающей точкой. В последнем случае следует выполнять операцию так как будто мнимая часть равна нулю.
  + *Примечание: смотрите слайды посвященные перегрузке бинарных операторов и операторов преобразования типов в презентации lesson-02-oop.pptx*
* Класс должен реализовывать операторы равенства (==) и неравенства (!=),а также переопределять метод Equals();
  + *Примечание: смотрите слайд посвященный перегрузке операторов равенства в презентации lesson-02-oop.pptx*
* Он должен переопределять метод ToString();
  + Вызов ToString() должен возвращать алгебраическую форму числа то есть строку вида «реальная + iмнимая)»
* Он должен реализовывать интерфейс IFormattable:
  + Для формата P или p должна возвращаться строка вида «(реальная, мнимая)»;
  + Для формата A или a должна возвращаться строка вида «реальная + iмнимая)». Обратите внимание что, когда мнимая часть отрицательная, строка должна иметь вид «реальная - iмнимая)». Если мнимая часть равна нулю, то возвращается строка только с реальной частью;
  + Если в качестве формата передано значение null, то используем формат A;
  + Дополнительное задание: при использовании неизвестного формата должно генерироваться исключение FormatException.
  + *Примечание: смотрите слайд посвященный интерфейсу IFormattable в презентации lesson-02-oop.pptx*
* Он должен реализовывать интерфейс IEquatable <ComplexNumber >;

# Unit-тесты

Если вы знаете что такое unit-тесты и умеете с ними работать, то в папке Homework-Tests\ComplexNumber.Tests находится проект с NUnit тестами для класса ComplexNumber. Разместите код вашего класса в файле ComplexNumber.student.cs и запустите тесты. Успешное выполнение всех тестов означает правильность выполнения задания.

# Правила работы с комплексными числами

* Сложение
  + (a+bi) + (c+di) = (a+c) + (b+d)i
* Вычитание
  + (a+bi) - (c+di) = (a-c) + (b-d)i
* Умножение
  + (a+bi) \* (c+di) = (ac-bd) + (bc+ad)i
* Сравнение
  + Два комплексных числа равны между собой тогда и только тогда, когда равны их действительные и мнимые части.

<https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%BB%D0%B5%D0%BA%D1%81%D0%BD%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE>

# Пример использования класса ComplexNumber

Скопируйте нижеприведенный код в функцию Main и убедитесь что он компилируется и правильно работает.

ComplexNumber x = new ComplexNumber(10.5);

ComplexNumber y = null;

if (y == null && x != null)

{

y = new ComplexNumber(7, 8.1);

ComplexNumber z = x + y;

Console.WriteLine(z); // 17,5+i8,1

z -= 20;

Console.WriteLine(z); // -2,5+i8,1

z.Real += 13;

z.Imaginary -= 8.1;

Console.WriteLine(z); // 10,5

if (z == x && z != y)

{

Console.WriteLine("{0:A}", z - y); // 3,5-i8,1

Console.WriteLine("{0:P}", z - y); // (3,5, -8,1)

}

}

x = y = null;

if (x == y) Console.WriteLine("x == y == null");

После запуска на экране должны быть выведены следующие строки:

17,5+i8,1  
-2,5+i8,1  
10,5  
3,5-i8,1  
(3,5, -8,1)  
x == y == null