Cel

Celem zajęć jest zapoznanie się z przeciążaniem operatorów dla własnych klas lub struktur. W tym celu utworzymy strukturę **liczbaZespolona**, a następnie przeciążymy operatory dodawania, odejmowania, dzielenia i mnożenia. Dodamy również operator rzutowania na liczbę rzeczywistą i odwrotnie. Rzutowanie na liczbę rzeczywistą jest niebezpieczne i przyjmiemy, że wartość liczby rzeczywistej będzie równa co do wartości modułu liczby zespolonej. Dodamy również operator równości oraz operator (będący drugim elementem pary logicznej), który sprawdza, czy obie liczby zespolone są różne.

Wymagania

- Struktury
- Właściwości
- Przeciążanie operatorów - składnia
- Własna konwersja (bezpieczna i niebezpieczna)
- Pary logiczne
- Typy nullable
- Indeksery

Zadanie 1

Utwórz nowy projekt aplikacji konsolowej. Dodaj nową strukturę **liczbaZespolona**. Do struktury dodaj prywatne atrybuty typu double _re i _im określające wartości części rzeczywistej oraz zespolone.

Zadanie 2

Dodaj właściwość typu **double**, która pozwoli na odczytanie modułu liczby zespolonej. Nazwij ją **modul**. Poniżej przedstawiono wzór na moduł liczby zespolonej (**a**, **b**i).

$$|\vec{z}| = \sqrt{a^2 + b^2}$$

Zadanie 3

Dodaj właściwość typu **liczbaZespolona**, która zwraca liczbę zespoloną. Nazwij ją **liczbaSprzezona**.

$$\overline{z} = a - bi$$

Zadanie 4

Dodaj statyczne, publiczne metody dodające, odejmujące, dzielące i mnożące liczby zespolone:

- dodawanie i odejmowanie: $(a + bi) \pm (c + di) = (a \pm c) + (b \pm d)i$,
- mnożenie:
  $(a + bi)(c + di) = ac + (bc + ad)i + bdi^2 = (ac - bd) + (bc + ad)i$,

- dzielenie: $$\frac{a+bi}{c+di} = \frac{(a+bi)(c-di)}{(c+di)(c-di)} = \frac{(ac+bd)+(bc-ad)i}{c^2+d^2}$$.

By ułatwić całość zadania, utwórz najpierw publiczny konstruktor dla struktury **liczbaZespolona**

Zadanie 5

Przesłoń metodę ToString()

Zadanie 6

Przeciąż operatory dodawania, odejmowania, mnożenia i dzielenia. W tym celu wykorzystaj napisane w zadaniu 4 metody. Argumentami tych operatorów są liczby zespolone.

Zadanie 7

Napisz w funkcji main kod, który wykorzystuje wszystkie napisane operatory i metody.

Zadanie 8

Przeciąż operator konwersji liczby zespolonej na zmienną typu double.

Zadanie 9

Przeciąż operator porównania == oraz odpowiadający jego parze logicznej operator !=.

Zadanie 10

Przeciąż operator rzutowania liczby rzeczywistej (double) na liczbę zespoloną (liczbaZespolona).

Podpowiedź

```
    {
        LiczbaZespolona a = new LiczbaZespolona(1, 1);
        LiczbaZespolona b = new LiczbaZespolona(2, 3);

        System.Console.WriteLine("{0} + {1} = {2}", a, b, a + b);
        System.Console.WriteLine("{0} - {1} = {2}", a, b, a - b);
        System.Console.WriteLine("{0} * {1} = {2}", a, b, a * b);
        System.Console.WriteLine("{0} / {1} = {2}", a, b, a / b);

        System.Console.WriteLine("{0} + {1} = {2}", a, 1, a + 1);

        System.Console.ReadLine();
    }
}

public struct LiczbaZespolona
{
    private double _re; // część rzeczywista
    private double _im; // część zespolona

    public LiczbaZespolona(double re, double im)...

    public double re { get { return _re; } }
    public double im { get { return _im; } }
    public double modul { get { return Math.Sqrt(_--   _-- . _----   _---); } }
    public LiczbaZespolona sprzezona { get { return new LiczbaZespolona(_--,  _---); } }

    static public LiczbaZespolona dodaj(LiczbaZespolona lho, LiczbaZespolona rho)...
    static public LiczbaZespolona odejmij(LiczbaZespolona lho, LiczbaZespolona rho)...
    static public LiczbaZespolona pomnoz(LiczbaZespolona lho, LiczbaZespolona rho)...
    static public LiczbaZespolona podziel(LiczbaZespolona lho, LiczbaZespolona rho)...

    public static LiczbaZespolona operator +(LiczbaZespolona lho, LiczbaZespolona rho)...
    public static LiczbaZespolona operator -(LiczbaZespolona lho, LiczbaZespolona rho)...
    public static LiczbaZespolona operator *(LiczbaZespolona lho, LiczbaZespolona rho)...
    public static LiczbaZespolona operator /(LiczbaZespolona lho, LiczbaZespolona rho)...

    public static bool operator ==(LiczbaZespolona lho, LiczbaZespolona rho)...
    public static bool operator !=(LiczbaZespolona lho, LiczbaZespolona rho)...

    public static explicit operator double(LiczbaZespolona rho)...
    public static implicit operator LiczbaZespolona(double rho)...

    public override bool Equals(object obj)...
    public override int GetHashCode()...
    public override string ToString()...
}
```