# 1

Zbudować klasę *Wektor* reprezentującą wektor stereometryczny (trójwymiarowy), złożoną z następujących elementów:

* Pola *double* **x**, **y**, **z** - wartości na poszczególnych współrzędnych
* Konstruktora wymagającego wszystkich współrzędnych
* Własność tylko do odczytu *double* **Długość** {**get**;} - można skorzystać z IloczynSkalarny(this, this)
* Statyczne metody:
  + *double* **IloczynSkalarny**(*Wektor* **V**, *Wektor* **W**)
  + *Wektor* **Suma**(*Wektor* **V**, *Wektor* **W**)

## B

Zdefiniować:

* operator +(Wektor,Wektor) zwracający nową instancję, będącą sumą wektorów
* operator -(Wektor,Wektor) zwracający nową instancję, będącą różnicą wektorów
* operatory \*(Wektor,double) i \*(double,Wektor), zwracające nową instancję, będącą wektorem wymnożonym przez skalar
* operatory /(Wektor,double), zwracające nową instancję, będącą wektorem podzielonym przez skalar

# 2

## A

Zbudować klasę *Wektor* reprezentującą wektor o dowolnym wymiarze

* Pola *double[]* **współrzędne** - wartości na poszczególnych współrzędnych
* Konstruktora/ów *Wektor*(byte **wymiar**) i/lub *Wektor*(params *double[]* **współrzędne**)
* Własność tylko do odczytu *double* **Długość** {**get**;} - można skorzystać z IloczynSkalarny(this, this)
* Własność tylko do odczytu *uint* **Wymiar** {get;} - albo osobne pole, albo po prostu długość tablicy współrzędnych
* Indeksatora *this[byte]*
* Statyczne metody:
  + *double(?)* **IloczynSkalarny**(*Wektor* **V**, *Wektor* **W**) - należy skontrolować wymiar obu wektorów i jeśli jest różny to zwracamy double.NaN (albo null ?) lub wyjątek
  + *Wektor* **Suma**(*Wektor* **V**, *Wektor* **W**) - tak jak wyżej skontrolować wymiar…

## B

Do klasy wektora z punktu A dodać

* statyczne słownik <*char/string*, *byte*>, np [‘x’] = 0, [‘y’] = 1, [‘z’] = 2, [‘t’] = 3
* Indeksator *this[char]/this[string]*
* Metodę string WypiszWektorPoWspółrzędnych(), która:
  + jeśli wymiar wektora jest mniejszy lub równy ilości nazwanych osi, to zwróci:  
    “x = [wartość], y = [wartość]...”
  + w przeciwnym wypadku zwróci:  
    “x1 = [wartość], x2 = [wartość]...”

## C

Zdefiniować operatory analogiczne do tych z 1B