Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 3 з дисципліни
«Основи програмування 2. Модульне програмування»

«Перевантаження операторів»

Варіант 35

Виконав студент ІП-15, Шабанов Метін Шаміль огли
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2022

Лабораторна робота 3

Перевантаження операторів
Варіант 35 (1)

Завдання:

1. Розробити клас "Вектор", який задається полярними координатами (довжина і кут). Реалізувати для нього декілька конструкторів, геттери, метод повороту вектору на будь-який кут. Перевантажити оператори "+" та "/" для знаходження суми векторів і "зменшення" вектору відповідно. Створити три вектори (B1, B2, B3), використовуючи різні конструктори. Повернути вектор B3 на 45°, вектор B2 "зменшити" у 2 рази. Визначити вектор B1 як суму змінених векторів B2 та B3.

Виконання:

C#:

C# Main.cs | C# Vector.cs | C# GetInput.cs

```
using System;

namespace Lab4_OperatorOverloading
{
    class Program
    {
        static void Main(string[] args)
        {
            var B1 = new Vector();

            var B2 :Vector = GetInput.getVector2Data();

            var B3 :Vector = GetInput.getVector3Data();

            Console.WriteLine($"Before change: \n Vector 1: {B1} \n Vector 2: {B2} \n Vector 3: {B3}");

            B3.TurnVector( angle: 45);
            B2 = B2 / 2;
            B1 = B2 + B3;

            Console.WriteLine($"After change: \n Vector 1: {B1} \n Vector 2: {B2} \n Vector 3: {B3}");
        }
    }
}
```

C# Main.cs | C# Vector.cs | C# GetInput.cs

```
using System;

namespace Lab4_OperatorOverloading
{
    16 usages  4 exposing APIs
    public class Vector
    {
        private double _length;
        private double _angle; //in degrees

        1 usage
        public Vector()
        {
            _length = 1;
            _angle = 0;
        }

        1 usage
        public Vector(double length)
        {
            _length = length;
            _angle = 0;
        }

        3 usages
        public Vector(double length, double angle)
        {
            _length = length;
            _angle = angle;
        }

        public double Length
        {
            get { return _length; }
        }

        public double Angle
        {
            get { return _angle; }
        }

        1 usage
        public void TurnVector(int angle)
        {
            _angle = (angle + _angle) % 360;
        }

        public static Vector operator +(Vector vect1, Vector vect2)
        {
            double resultingX = _getXSumInDecart(vect1, vect2);
            double resultingY = _getYSumInDecart(vect1, vect2);

            double newLength = Math.Sqrt(Math.Pow(resultingX, 2) + Math.Pow(resultingY, 2));
            double newAngle = (180 / Math.PI) * Math.Acos(resultingX / newLength);

            return new Vector(newLength, newAngle);
        }
```

```
53
54          public static Vector operator /(Vector vect1, double decreaseTo)
55          {
56              double vectLength = vect1._length / decreaseTo;
57
58              return new Vector(vectLength, vect1._angle);
59          }
60
61          public override string ToString()
62          {
63              return $"({_length}, {_angle}°)";
64          }
65
                1 usage
66          private static double _getXSumInDecart(Vector vect1, Vector vect2)
67          {
68              double decartX1 = vect1._length * Math.Cos(_convertDegToRad(vect1._angle));
69              double decartX2 = vect2._length * Math.Cos(_convertDegToRad(vect2._angle));
70
71              return decartX1 + decartX2;
72          }
73
                1 usage
74          private static double _getYSumInDecart(Vector vect1, Vector vect2)
75          {
76              double decartY1 = vect1._length * Math.Sin(_convertDegToRad(vect1._angle));
77              double decartY2 = vect2._length * Math.Sin(_convertDegToRad(vect2._angle));
78
79              return decartY1 + decartY2;
80          }
81
                4 usages
82          private static double _convertDegToRad(double degrees)
83          {
84              return (Math.PI / 180) * degrees;
85          }
86
87      }
88  }
```

C# Main.cs | C# Vector.cs | C# GetInput.cs

```
using System;
using System.Text.RegularExpressions;

namespace Lab4_OperatorOverloading
{
    2 usages
    public static class GetInput
    {
        1 usage
        public static Vector getVector2Data()
        {
            Console.Write("Enter the length of vector 2: ");
            var vect2Length :double = _getNumbers();
            Console.Write("Enter the angle of vector 2 in degrees: ");
            var vect2Angle :double = _getNumbers();

            return new Vector(vect2Length, vect2Angle);
        }

        1 usage
        public static Vector getVector3Data()
        {
            Console.Write("Enter the length of vector 3: ");
            var vect3Length :double = _getNumbers();

            return new Vector(vect3Length);
        }

        3 usages
        private static double _getNumbers()
        {
            double number = _checkOnNumber( input: Console.ReadLine());
            return number;
        }

        1 usage
        private static double _checkOnNumber(string input)
        {
            while (!Regex.IsMatch(input, pattern: @"^\d+[,]?\d*$"))
            {
                Console.Write("Not a number! Try again: ");
                input = Console.ReadLine();
            }

            return Double.Parse(input);
        }
    }
}
```

Тестування:

C#:

```
Enter the length of vector 2: 8
Enter the angle of vector 2 in degrees: 35,156
Enter the length of vector 3: 15,2
Before change:
 Vector 1: (1, 0°)
 Vector 2: (8, 35,156°)
 Vector 3: (15,2, 0°)
After change:
 Vector 1: (19,15332009350807, 42,95383287153542°)
 Vector 2: (4, 35,156°)
 Vector 3: (15,2, 45°)

Process finished with exit code 0.
```