**Practice "Non-static methods"**

You suddenly realized that it is not very convenient to write the name of the Geometry class when performing any operation on vectors and segments. However, you cannot refuse this class, because in the few minutes while you submitted the previous task, thousands of people downloaded your library and began to use it in their projects.

Therefore, you decided to keep this class, but add the methods Vector.GetLength (), Segment.GetLength (), Vector.Add (Vector), Vector.Belongs (Segment) and Segment.Contains (Vector) not instead, but together with the corresponding methods the Geometry class.

Do it! Each of these methods must call an existing method on the Geometry class to avoid duplicating code.

All the functionality of the previous step should remain!

// Paste the final content of the VectorTask.cs file here

**Code:**

**Contents of the Program.cs file:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using GeometryTasks;

namespace GeometryTasks

{

class Program

{

public static Vector

v1 = new Vector { X = 3.2, Y = 8.5 },

v2 = new Vector { X = 5.1, Y = 3.4 },

VSum = new Vector { X = 0, Y = 0 };

public static Segment

sgm = new Segment

{

Begin = new Vector() { X = 8.1, Y = 11.36 },

End = new Vector() { X = 15.74, Y = 19.42 }

};

public static void Main(string[] args)

{

Console.WriteLine("Длина вектора v1={0}", Geometry.GetLength(v1));

Geometry.Add(v1, v2);

Console.WriteLine("Суммирующий вектор х={0} y={1}", Geometry.VSum.X, Geometry.VSum.Y);

Console.WriteLine();

Console.WriteLine("Длина сегмента sgm={0}", Geometry.GetLength(sgm));

Console.WriteLine(Geometry.IsVectorInSegment(v1, sgm));

Console.WriteLine();

Console.WriteLine(v1.Belongs(sgm));

Console.WriteLine();

Console.WriteLine(sgm.Contains(v1));

Console.ReadKey();

}

}

}

**Contents of the VectorTask.cs file:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using GeometryTasks;

namespace GeometryTasks

{

public class Vector

{

public double X;

public double Y;

public double GetLength()

{

return Geometry.GetLength(this);

}

public Vector Add(Vector v)

{

return Geometry.Add(this, v);

}

public bool Belongs(Segment s)

{

return Geometry.IsVectorInSegment(this, s);

}

}

public class Segment

{

public Vector Begin;

public Vector End;

public double GetLength()

{

return Geometry.GetLength(this);

}

public bool Contains(Vector v)

{

return Geometry.IsVectorInSegment(v, this);

}

}

public static class Geometry

{

public static Vector

VSum = new Vector { X = 0, Y = 0 };

public static Segment

Sg = new Segment

{

Begin = new Vector { X = 0, Y = 0 },

End = new Vector { X = 1, Y = 1 }

};

public static double DlinaV, S, S1;

public static bool B = false;

public static double GetLength(Vector v1)

{

S = v1.X \* v1.X + v1.Y \* v1.Y;

if (S > 0)

{

return DlinaV = Math.Sqrt(S);

}

else return 0;

}

public static Vector Add(Vector v1, Vector v2)

{

VSum.X = v1.X + v2.X;

VSum.Y = v1.Y + v2.Y;

return VSum;

}

public static double GetLength(Segment sgm)

{

S1 = (sgm.End.X - sgm.Begin.X) \* (sgm.End.X - sgm.Begin.X) +

(sgm.End.Y - sgm.Begin.Y) \* (sgm.End.Y - sgm.Begin.Y);

return Math.Sqrt(S1);

}

public static bool IsVectorInSegment(Vector v, Segment sg)

{

B = ((v.X - sg.Begin.X) \* (v.X - sg.End.X) <= 0) && ((v.Y - sg.Begin.Y) \* (v.Y - sg.End.Y) < 0);

if (((v.X == sg.Begin.X) || (v.X == sg.End.X)) && ((v.Y == sg.End.Y) || (v.Y == sg.Begin.Y)))

return true;

else

return B;

}

}

}