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 №5 з дисципліни «Основи програмування 2. Модульне програмування»

«Успадкування та Поліморфізм»

Варіант 27

Виконав студент ІП-11 Савенко Олексій Андрійович

(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська І.І.

(прізвище, ім'я, по батькові)

Лабораторна робота №5

Успадкування та Поліморфізм

Мета роботи – вивчити механізми створення і використання класів та об’єктів

Варіант 27

27. Створити клас TTriangle, який містить координати вершин і методи обчислення його площі та периметру. На основі цього класу створити класи-нащадки, які представляють рівносторонні, прямокутні та рівнобедрені трикутники. Створити певну кількість трикутників кожного виду, щоб їх сумарна кількість дорівнювала n. Для рівносторонніх та прямокутних трикутників обчислити суму їх площ, а для рівнобедрених – суму всіх периметрів.

Постановка завдання

Спочатку створимо клас точки **Point** з атрибутами координат(**X,Y**), створимо клас загального суперкласса трикутника **TTriangle** та застосуємо об’єкти класа точки як його об’єкти для визначення його 3 вершин. Створимо три підкласа трикутників унаслідувавши від батьківського методи знаходження периметру, площі та атрибути, а також виведення об’єктів відповідних трикутників. Перевизначимо у дочірніх класах відповідні методи, після цього створимо масиви об’єктів відповідних підклассів трикутників та обрахуємо суму площ для прямокутних та рівносторонніх трикутників, а також суму периметрів для рівнобедрених трикутників.
Виведемо отримані фінальні дані.

Програмний код на мові C#:

TTriangle.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5OpSharp
{
  internal  class TTriangle //Загальний клас трикутника
  {
    protected Point P1, P2, P3; // Вершини трикутника

    public TTriangle(Point p1, Point p2,Point p3)
```

```
        {
            P1 = p1;
            P2 = p2;
            P3 = p3;

        }
        //Знаходження сторін трикутника
        public  double getA()
        {
            return Math.Sqrt(Math.Pow((P2.X - P1.X), 2) + Math.Pow((P2.Y - P1.Y), 2));
        }

        public double getB()
        {
            return Math.Sqrt(Math.Pow((P3.X-P1.X),2)+ Math.Pow((P3.Y-P1.Y),2));
        }

        public double getC()
        {
            return Math.Sqrt(Math.Pow((P3.X - P2.X), 2) + Math.Pow((P3.Y - P2.Y), 2));
        }
        //Знаходження периметру
        public  double GetPerimetr()
        {
          return getA() + getB() + getC();
        }
        //Віртуальний метод отримання площі трикутника
        public virtual double GetSquare()
        {
            return 0;
        }

        public  override string ToString()
        {
            return $"Triangle with coordinates A:{P1}, B:{P2}, C:{P3}\n";
        }
    }
}
```

RTriangle.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5OpSharp
{
    internal class RTriangle:TTriangle //прямокутний трикутник
    {
        public RTriangle(Point p1, Point p2, Point p3) : base(p1, p2, p3) //Виклик конструктора базового класу
        {
        }
        //Перевизначення функції(віртуальність функції)
        public override double GetSquare()
        {
            return 0.5 *
          (getA() > getB() ? (getA() > getC() ? getB() * getC() : getA() * getB()) : (getB() > getC() ? getC() * getA() : getA() *
getB()));
```

```
        }
        public override string ToString()
        {
            return $"RTriangle with coordinates A:{P1}, B:{P2}, C:{P3}\n";
        }
    }
}
```

ITriangle.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5OpSharp
{
    internal class ITriangle:TTriangle //рівнобедрений трикутник
    {
        public ITriangle(Point p1, Point p2, Point p3) : base(p1, p2, p3)
        {
        }

        public override string ToString()
        {
            return $"ITriangle with coordinates A:{P1}, B:{P2}, C:{P3}\n";
        }


    }
}
```

ETriangle.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5OpSharp
{
    internal class ETriangle:TTriangle // Рівносторонній трикутник
    {
        public ETriangle(Point p1, Point p2, Point p3) : base(p1, p2, p3)
        {
        }
        //Перевизначення методу знаходження площі
        public override double GetSquare()
        {
            return Math.Pow(getA(), 2) * Math.Sqrt(3) / 4;
        }

        public override string ToString()
        {
            return $"ETriangle with coordinates A:{P1}, B:{P2}, C:{P3}\n";
        }
```

```
    }
}
```

Point.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5OpSharp
{
    internal class Point //Клас точки
    {
        public double X { get; set; }
        public double Y { get; set; }



        public Point(double x=0, double y=0)
        {
            X = x;
            Y = y;
        }

        public override string ToString()
        {
            return $"({this.X},{this.Y})";

        }
    }
}
```

Worker.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5OpSharp
{
    internal class Worker
    { //Генерація масивів трикутників
        public ETriangle[] GetEtriangleArray(int n)
        {
            ETriangle[] triangles = new ETriangle[n];
            for(int i = 0; i < n; i++)
            {
                Point[] points = GetEtPointArray();
                triangles[i] = new ETriangle(points[0], points[1], points[2]);
            }
            return triangles;
        }
        public ITriangle[] GetItriangleArray(int n)
        {
            ITriangle[] triangles = new ITriangle[n];
            for (int i = 0; i < n; i++)
            {
```

```
      Point[] points = GetItPointArray();
      triangles[i] = new ITriangle(points[0], points[1], points[2]);
    }
    return triangles;
}

public RTriangle [] GetRtriangleArray(int n)
{
    RTriangle[] triangles = new RTriangle[n];
    for (int i = 0; i < n; i++)
    {
      Point[] points = GetRtPointArray();
      triangles[i] = new RTriangle(points[0], points[1], points[2]);
    }
    return triangles;

}

//Генерація масиву точок для об'єкту класа трикутників
public Point[] GetEtPointArray()
{
    Point[] points = new Point[3];
    Random random = new Random();
    Point p1 = new Point(random.Next(-10, 11), random.Next(-10, 11));
    Point p2 = new Point(random.Next(-10, 11), random.Next(-10, 11));
    Point p3 = new Point();
    p3.X = Math.Round((p2.X - p1.X) * 0.5 - (p2.Y - p1.Y) * Math.Sqrt(3) / 2 + p1.X);
    p3.Y = Math.Round((p2.X - p1.Y) * Math.Sqrt(3) / 2 + (p2.Y - p1.Y) * 0.5 + p1.Y);
    points[0] = p1;
    points[1] = p2;
    points[2] = p3;


    return points;
}
public Point[] GetItPointArray()
{
    Point[] points = new Point[3];
    Random random = new Random();
    Point p1 = new Point (random.Next(-10,11),random.Next(-10,11));
    Point p2 = new Point (random.Next(-10,11),random.Next(-10,11));
    Point p3 = new Point();
    if (p1.X > p2.X) p3.X = (p2.X - Math.Abs(p1.X - p2.X));

    else p3.X = Math.Abs(p1.X - p2.X) + p2.X;
    p3.Y = p1.Y;
    points[0] = p1;
    points[1] = p2;
    points[2] = p3;

    return points;
}



public Point[] GetRtPointArray()
{
    Point[] points = new Point[3];
    Random random = new Random();
    Point p1 = new Point(random.Next(-10,11),random.Next(-10,11));
    Point p2 = new Point(random.Next(-10, 11), random.Next(-10, 11));
    Point p3 = new Point(p1.X, p2.Y);
```

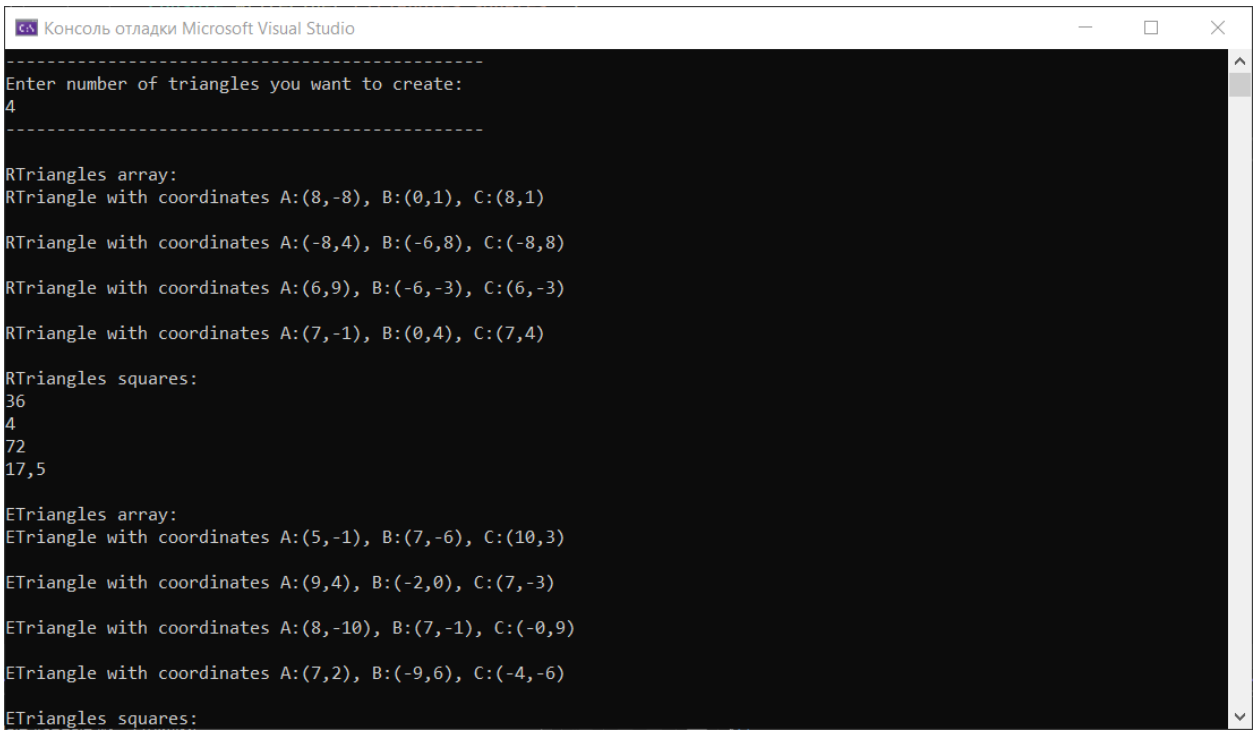
```
            points[0] = p1;
            points[1] = p2;
            points[2] = p3;

            return points;
        }

        //Виведення масиву трикутників
        public void printArray(TTriangle[]triangles)
        {
            foreach (var triangle in triangles)
            {
             Console.WriteLine(triangle);

            }

        }
        //Підрахунок суми площ трикутників з масиву
        public double GetSumSquares(TTriangle[]triangles)
        {
            return triangles.Sum(triangle=>triangle.GetSquare());
        }

        //Підрахунок суми периметрів трикутників з масиву
        public double SumOfPerimeteres(TTriangle[] triangles)
        {
            return triangles.Sum(triangle => triangle.GetPerimetr());

        }

        //Виведення площ трикутників з масиву
        public void printSquare(TTriangle[] triangles)
        {
            foreach(var triangle in triangles)
            {
                Console.WriteLine(Math.Round(triangle.GetSquare(),2));
            }
        }
        //Виведення периметрів трикутників з масиву
        public void printPerimeter(ITriangle[] triangles)
        {
            foreach (var triangle in triangles)
            {
                Console.WriteLine(Math.Round(triangle.GetPerimetr(),2));
            }
        }

    }
}
```

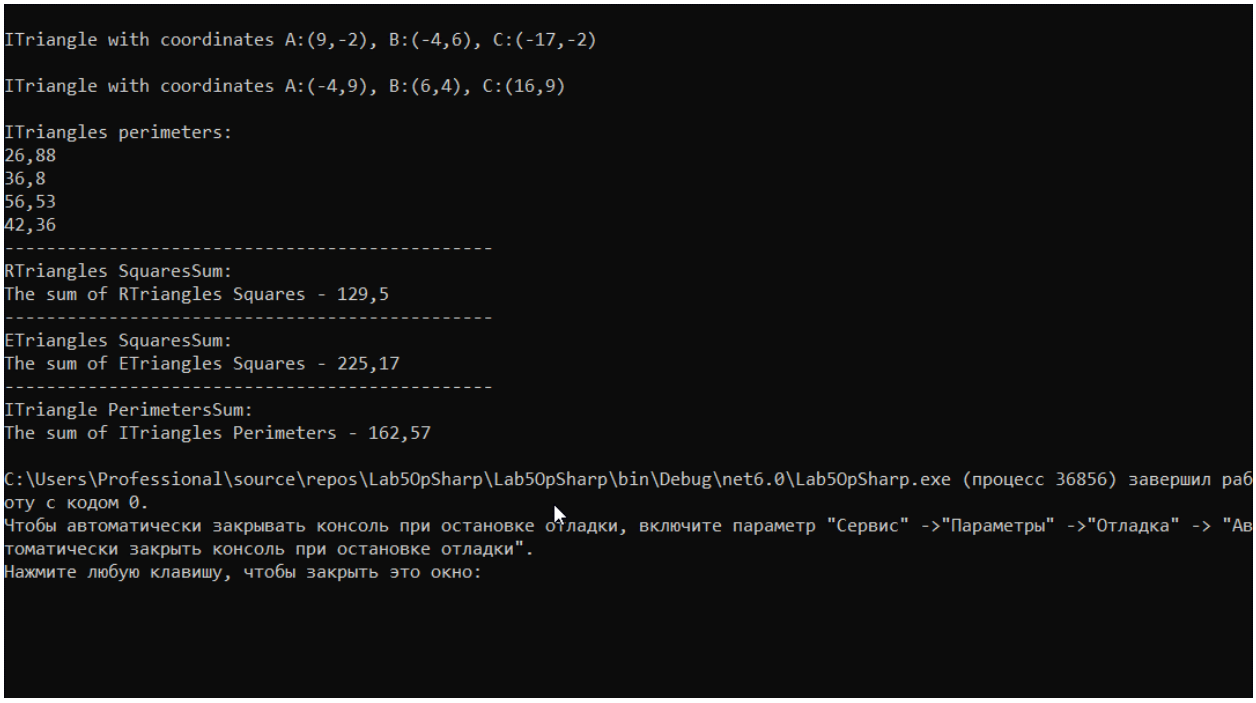
Program.cs:

```
namespace Lab5OpSharp
{
    internal class Program
    {
        static void Main(string[] args)
        {   Worker worker = new Worker();
            Console.WriteLine("-----------------------------------------------");
            Console.WriteLine("Enter number of triangles you want to create: ");

            int n = Convert.ToInt32(Console.ReadLine());
```

```
            RTriangle[] rTriangles = worker.GetRtriangleArray(n);

            ITriangle[] iTriangles = worker.GetItriangleArray(n);

            ETriangle[] eTriangles = worker.GetEtriangleArray(n);
            Console.WriteLine("-----------------------------------------------");
            Console.WriteLine("\nRTriangles array:");
            worker.printArray(rTriangles);
            Console.WriteLine("RTriangles squares:");
            worker.printSquare(rTriangles);

            Console.WriteLine("\nETriangles array:");
            worker.printArray(eTriangles);
            Console.WriteLine("ETriangles squares:");
            worker.printSquare(eTriangles);

            Console.WriteLine("\nITriangles array:");
            worker.printArray(iTriangles);
            Console.WriteLine("ITriangles perimeters:");
            worker.printPerimeter(iTriangles);

            Console.WriteLine("-----------------------------------------------");
            Console.WriteLine("RTriangles SquaresSum:");
            double sumSquaresRt = worker.GetSumSquares(rTriangles);
            Console.WriteLine($"The sum of RTriangles Squares - {Math.Round(sumSquaresRt,2)}");
            Console.WriteLine("-----------------------------------------------");
            Console.WriteLine("ETriangles SquaresSum:");
            double sumSquaresEt = worker.GetSumSquares(eTriangles);
            Console.WriteLine($"The sum of ETriangles Squares - {Math.Round(sumSquaresEt,2)}");
            Console.WriteLine("-----------------------------------------------");
            Console.WriteLine("ITriangle PerimetersSum:");
            double sumPerimetersIt = worker.SumOfPerimeteres(iTriangles);
            Console.WriteLine($"The sum of ITriangles Perimeters - {Math.Round(sumPerimetersIt,2)}");



        }
    }
}
```

Виконання програми на мові C#:





Програмний код на мові Python:

main.py:

```
import func


if __name__ == '__main__':
    n = int(input("How many triangles you want to create?\n"))
```

```
    pArr = [] # список об'єктів класу PTriangle
    eArr = [] # ...ETriangle
    iArr = [] # ...ITriangle
    sumP = 0 # сума площ прямокутних трикутників
    sumE = 0 # сума площ рівносторонніх трикутників
    sumI = 0 # сума площ рівнобедренних трикутників
    for i in range(0, n):
        # генеруємо вершини прямокутного трикутника
        pointRT = func.GetRtPoint()
        temp = func.PTriangle(pointRT[0], pointRT[1], pointRT[2])
        pArr.append(temp)

        #генеруємо вершини рівнобедренного трикутника
        pointIT = func.GetItPoint()
        temp = func.ITriangle(pointIT[0], pointIT[1], pointIT[2])
        iArr.append(temp)

        # генеруємо вершини рівностороннього трикутника
        pointET = func.GetEtPoint()
        temp = func.ETriangle(pointET[0], pointET[1], pointET[2])
        eArr.append(temp)

    for i in range(0, n):
        pArr[i].print()
    print("--------------------")
    for i in range(0, n):
        eArr[i].print()
    print("--------------------")
    for i in range(0, n):
        iArr[i].print()
        sumP = sumP + pArr[i].getSquare()
        sumI = sumI + eArr[i].getSquare()
        sumE = sumE + iArr[i].getPerimeter()
    print("--------------------")

    print("Sum of squares of right triangles: " + str(round(sumP, 2)) +
";\nSum of squares of equilateral triangles: " + str(round(sumE, 2)) +
";\nSum of perimeters of isosceles triangles: " + str(round(sumI, 2)) + ".")
```

func.py:

```
import math
import random

class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def getX(self):
        return self.x

    def getY(self):
        return self.y

    def setX(self, x):
        self.x = x

    def setY(self, y):
        self.y = y
```

```
    def __str__(self): # перетворюємо інформацію про точку в рядок
        return f"({self.x},{self.y})"


class TTriangle:
    def __init__(self, P1, P2, P3): # використовуємо об'єкти класу точки як
атрибути класу трикутника (вершини)
        self.P1 = P1
        self.P2 = P2
        self.P3 = P3

    def getA(self): # знаходимо сторони трикутника
        return round(math.sqrt(pow(self.P2.getX() - self.P1.getX(), 2) +
pow(self.P2.getY() - self.P1.getY(), 2)), 2)

    def getB(self):
        return round(math.sqrt(pow(self.P3.getX() - self.P1.getX(), 2) +
pow(self.P3.getY() - self.P1.getY(), 2)), 2)

    def getC(self):
        return round(math.sqrt(pow(self.P3.getX() - self.P2.getX(), 2) +
pow(self.P3.getY() - self.P2.getY(), 2)), 2)

    def getPerimeter(self):  # знаходимо периметр трикутника
        return self.getA() + self.getB() + self.getC()
    def getSquare(self):
        pass

    def print(self):
        pass

class PTriangle(TTriangle): # прямокутний трикутник
    def __init__(self, P1, P2, P3): # наслідуємо від класу TTriangle атрибути
(об'єкти класу точки)
        super().__init__(P1, P2, P3)

    def print(self):
        print(f"PTriangle:\n A:{self.P1}, B:{self.P2}, C:{self.P3}\nLength of
AB:{self.getA()}, BC:{self.getB()}, AC:{self.getC()}" )

    def getSquare(self): # знаходимо площу трикутника
        if self.getA() > self.getB():
            if self.getA() > self.getC():
                return 0.5 * self.getB() * self.getC()
            elif self.getB() > self.getC():
                return 0.5 * self.getC() * self.getA()
            else:
                return 0.5 * self.getA() * self.getB()


class ETriangle(TTriangle): # рівносторонній трикутник
    def __init__(self, P1, P2, P3): # наслідуємо від класу TTriangle атрибути
(об'єкти класу точки)
        super().__init__(P1, P2, P3)

    def print(self):
        print(f"ETriangle:\nA:{self.P1}, B:{self.P2}, C:{self.P3}\nLength of
AB:{self.getA()}, BC:{self.getB()}, AC:{self.getC()}")

    def getSquare(self): # знаходимо площу трикутника
        return self.getA()*self.getA()*math.sqrt(3)/4


class ITriangle(TTriangle): # рівнобедренний трикутник
```

```
    def __init__(self, P1, P2, P3): # наслідуємо від класу TTriangle атрибути
(об'єкти класу точки)
        super().__init__(P1, P2, P3)

    def print(self):
        print(f"ITriangle:\nA:{self.P1}, B:{self.P2}, C:{self.P3}\nLength of
AB:{self.getA()}, BC:{self.getB()}, AC:{self.getC()}")


def GetEtPoint():
    p1 = Point(round(random.uniform(-10, 10), 2), round(random.uniform(-10,
10), 2))
    p2 = Point(round(random.uniform(-10, 10), 2), round(random.uniform(-10,
10), 2))
    p3 = Point(0, 0)
    p3.setX(round((p2.getX() - p1.getX()) * 0.5 - (p2.getY() - p1.getY()) *
math.sqrt(3) / 2 + p1.getX(), 2))
    p3.setY(round((p2.getX() - p1.getX()) * math.sqrt(3) / 2 + (p2.getY() -
p1.getY()) * 0.5 + p1.getY(), 2))
    return [p1, p2, p3]


def GetItPoint():
    p1 = Point(round(random.uniform(-10, 10), 2), round(random.uniform(-10,
10), 2))
    p2 = Point(round(random.uniform(-10, 10), 2), round(random.uniform(-10,
10), 2))
    p3 = Point(0, 0)
    if p1.getX() > p2.getX():
        p3.setX(round(p2.getX() - abs(p1.getX() - p2.getX()), 2))
    else:
        p3.setX(round(abs(p1.getX() - p2.getX()) + p2.getX(), 2))
    p3.setY(p1.getY())
    return [p1, p2, p3]

def GetRtPoint():
    p1 = Point(round(random.uniform(-10, 10), 2), round(random.uniform(-10,
10), 2))
    p2 = Point(round(random.uniform(-10, 10), 2), round(random.uniform(-10,
10), 2))
    p3 = Point(p1.getX(), p2.getY())
    return [p1, p2, p3]
```

Виконання програми на мові Python:

```
How many triangles you want to create?
4
PTriangle:
 A:(0.99,-9.4), B:(1.75,-6.84), C:(0.99,-6.84)
Length of AB:2.67, BC:2.56, AC:0.76
PTriangle:
 A:(-2.69,0.51), B:(4.16,0.37), C:(-2.69,0.37)
Length of AB:6.85, BC:0.14, AC:6.85
PTriangle:
 A:(7.9,1.76), B:(7.51,-7.32), C:(7.9,-7.32)
Length of AB:9.09, BC:9.08, AC:0.39
PTriangle:
 A:(1.79,-3.89), B:(7.82,-3.47), C:(1.79,-3.47)
Length of AB:6.04, BC:0.42, AC:6.03
--------------------
ETriangle:
A:(-3.8,4.4), B:(5.42,-6.72), C:(10.44,6.82)
Length of AB:14.45, BC:14.44, AC:14.44
ETriangle:
A:(-1.21,-6.65), B:(7.23,-6.11), C:(2.54,0.93)
Length of AB:8.46, BC:8.46, AC:8.46
ETriangle:
A:(-9.8,-6.39), B:(-9.83,0.27), C:(-15.58,-3.09)
Length of AB:6.66, BC:6.66, AC:6.66
ETriangle:
A:(-8.75,3.95), B:(-1.28,6.5), C:(-7.22,11.69)
Length of AB:7.89, BC:7.89, AC:7.89
--------------------
ITriangle:
```

```
Length of AB:14.45, BC:14.44, AC:14.44
ETriangle:
A:(-1.21,-6.65), B:(7.23,-6.11), C:(2.54,0.93)
Length of AB:8.46, BC:8.46, AC:8.46
ETriangle:
A:(-9.8,-6.39), B:(-9.83,0.27), C:(-15.58,-3.09)
Length of AB:6.66, BC:6.66, AC:6.66
ETriangle:
A:(-8.75,3.95), B:(-1.28,6.5), C:(-7.22,11.69)
Length of AB:7.89, BC:7.89, AC:7.89
--------------------
ITriangle:
A:(5.48,-9.38), B:(-2.43,-4.97), C:(-10.34,-9.38)
Length of AB:9.06, BC:15.82, AC:9.06
ITriangle:
A:(-3.95,3.01), B:(-9.4,8.84), C:(-14.85,3.01)
Length of AB:7.98, BC:10.9, AC:7.98
ITriangle:
A:(-1.57,2.83), B:(-9.95,6.09), C:(-18.33,2.83)
Length of AB:8.99, BC:16.76, AC:8.99
ITriangle:
A:(6.28,4.6), B:(-6.91,-8.28), C:(-20.1,4.6)
Length of AB:18.44, BC:26.38, AC:18.44
--------------------
Sum of squares of right triangles: 4.49;
Sum of squares of equilateral triangles: 158.8;
Sum of perimeters of isosceles triangles: 167.57.

Process finished with exit code 0
```

Висновок

Під час виконання лабораторної роботи були застосовані підпрограми як програмні фрагменти, код реалізований відповідно до модульного підходу, створено окремі файли: мейну, з прототипами класів та з їх описом. Особливістю виконання роботи є використання класу, а саме його атрибутів (об’єктів класу) та абстрактних методів для наслідування похідними класами. Для обробки декількох об’єктів класу були створені відповідні масиви об’єктів. Для коректного задання вершин трикутника, вони задаються автоматично. Для обчислення площ (периметрів) використовуються відповідні формули для кожного трикутника.