**Звіт до Лабораторної роботи №4**

**Тема:** Абстрактні класи та інтерфейси.

**Мета роботи:** Використовуючи теоретичне підґрунтя абстракті класи та інтерфейси у мові Java, виконати дії що будуть вказані в завданні до лабораторної роботи.

**Завдання для виконання**

Створити три класи Triangle, Quadrilateral, Circle, що розширяють абстрактний клас Figure.

Реалізувати конструктор для кожного з класів:

* Для трикутника конструктор у якості параметра приймає три вершини класу Point.
* Для чотирикутника конструктор у якості параметра приймає чотири вершини класу Point.
* Для кола конструктор у якості параметра приймає координати центpу класу Point та радіус типу double.
* Конструктор кожної з фігур має перевіряти, що фігура є не виродженою.

Реалізувати методи підрахунку площі для кожної фігури.

Реалізувати метод toString() для кожної з фігур який виводить текст у вказаному вигляді:

* Трикутник - Triangle[A(x,y) B(x,y) C(x,y)]
* Чотирикутник - Quadrilateral[A(x,y) B(x,y) C(x,y) D(x,y)]
* Коло - Circle[(x,y) Radius]

Реалізувати метод Point centroid(). Повернути центроїд фігури. Іншими словами, це має бути "центроїд площі".

**Хід роботи**

Для виконання цієї роботи мною було створено 7 класів: Point, Segment, Circle, Triangle, Quadro, абстрактний клас Figure, а також клас Main із головною процедурою main.

**Клас Point** описує об’єкти Точка на площині. Об’єкт точка має атрибути – координати х та у,

Метод-Конструктор цього класу: public Point(double x, double y), отримує координати (х, у) точки в якості параметрів та створює об’єкт Точка.

Код класу Point:

public class Point {  
 double x, y;  
  
 public Point(double x, double y) {  
 this.x = x;  
 this.y = y;  
 }  
}

**Клас Segment** описує об’єкт Відрізок на площині. Відрізок має атрибути: firstPoint, secondPoint – початкова і кінцева точки Відрізка.

Метод-Конструктор: public Segment(Point firstPoint, Point secondPoint) – створює об’єкт класу Segment – відрізок із заданими точками firstPoint, secondPoint – тобто його кінцями. Також у цьому конструкторі реалізована перевірка того, чи не збігаються початок та кінець відрізка в одну точку.

Метод public double length() – розраховує довжину Відрізка із заданими точками firstPoint, secondPoint - його кінцями, використовуючи їх координати. Після цього довжина відрізка повертається в головну процедуру main.

Код класу Segment:

import static java.lang.Math.\*;  
  
public class Segment {  
 public Point firstPoint, secondPoint;  
 public Segment(Point firstPoint, Point secondPoint) {  
 if (firstPoint.x == secondPoint.x && firstPoint.y == secondPoint.y) {  
 System.*out*.println("This Segment is NOT exist: start Point and end Point must be different");  
 }  
 this.firstPoint = firstPoint;  
 this.secondPoint = secondPoint;  
 }  
  
 public double length() {  
 return *sqrt*(*pow*(secondPoint.x - firstPoint.x, 2) + *pow*(secondPoint.y - firstPoint.y, 2));  
 }  
}

**Абстрактний клас Figure** – це батьківський клас на основі якого створюються класи Circle, Triangle та Quadro. На початку оголошуються об’єкт точка point1 та дві змінні x та y – координати цієї точки.

Конструктор класу створює об’єкт точка point1 класу Point, із заданими координатами (х, у)

Код класу Figure:

public abstract class Figure {  
 public static double *x*, *y*;  
 public static Point *point1*;  
  
 public Figure(Point point1) {  
 *x* = point1.x;  
 *y* = point1.y;  
 }  
}

**Клас Circle** є розширенням класу Figure та описує об’єкт коло – сircle. На початку оголошуються змінна радіус кола.

Конструктор класу створює об’єкт коло circle із заданими параметрами (точка point1 – центр кола; радіус кола). Далі командою super(point1) створюється точка – центр кола. Далі йде перевірка чи існує таке коло, якщо фігура не існує то програма припиняє свою роботу.

Метод getArea() повертає площу кола.

Метод toString() повертає кординати точки центра кола та величину його радіусу і виводить їх на екран.

Код класу Circle:

public class Circle extends Figure {  
 public double radius;  
  
 public Circle(Point point1, double radius) {  
 super(point1);  
 // super(x,y);  
 this.radius = radius;  
 if (radius <=0){  
 System.*out*.println("This Circle not exist: Radius=" + radius);  
 Main.*figureExist* = false;  
 }  
 }  
   
 public double getArea() {  
 return Math.*PI* \* radius \* radius;  
 }  
   
 public String toString(){  
 return "Circle[(" + *x* + "," + *y* + ") " + radius + "]";  
 }  
}

**Клас Triangle** є розширенням класу Figure та описує об’єкт трикутник - triangle. На початку оголошуються три точки – координати вершин трикутника.

Конструктор класу створює об’єкт трикутник із заданими параметрами (точками вершин). Далі командою super(а) створюється точка – одна із вершин трикутника. Далі ініціалізуються всі вершини трикутника та створюються відрізки (ab, ac ,bc), з яких складається трикутник за допомогою класу Segment. Далі виконується перевірка того, чи існує такий трикутник, якщо фігура не існує то програма припиняє свою роботу.

Метод getArea(Point a, Point b, Point c) повертає площу трикутника. Точки a, b, c – вершини трикутника передаються в метод getArea() в якості параметрів. Далі створюються 3 об’єкти Segment (відрізки АВ, ВС, АС) – сторони трикутника, визначається довжина кожної сторони функцією length(). Потім розраховується півпериметр трикутника р, і за формулою Герона розраховується площа трикутника та повертається в головну процедуру main.

Метод getCentroid(Point a, Point b, Point c) повертає точку - центроїд трикутника. Точки a, b, c – вершини трикутника передаються в метод getCentroid() в якості параметрів. Центроїд – це точка перетину медіан трикутника, яка ділить кожну медіану у відношенні 2:1 (довша частина 2 – з боку вершини, коротша 1 – з боку протилежної сторони). Для визначення центроїда використовується медіана АМ, проведена з вершини а до сторони ВС. Спочатку визначаються координати точки М (Мх, Му) – середини відрізка ВС. Потім визначаються координати точки О(Ох, Оу) – центроїда трикутника використовуючи властивіть центроїда, описану вище.

В кінці методу створюється об’єкт точка-центроїд, з координатами (Ох, Оу) і повертається в головну процедуру main.

Метод toString() виводить інформацію про об’єкт Трикутник (координати його вершин) на екран у вигляді строки.

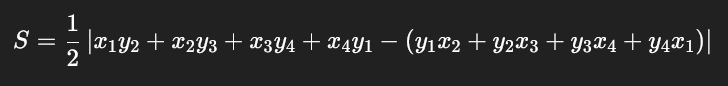
Код класу Triangle:

public class Triangle extends Figure {  
 public Point a, b, c;  
 public Triangle(Point a, Point b, Point c){  
 super(a);  
 this.a = a;  
 this.b = b;  
 this.c = c;  
 Segment ab = new Segment(a,b);  
 Segment bc = new Segment(b,c);  
 Segment ac = new Segment(a,c);  
 if ((ab.length()==0)||(bc.length()==0)||(ac.length()==0)){  
 System.*out*.println("This Triangle NOT Exist");  
 Main.*figureExist* = false;  
 }  
 if ((ab.length() >= (bc.length()+ac.length()))||(bc.length() >= (ab.length()+ac.length()))||(ac.length() >= (ab.length()+bc.length()))){  
 System.*out*.println("This Triangle NOT Exist");  
 Main.*figureExist* = false;  
 }  
 }  
  
 public static double getArea(Point a, Point b, Point c){  
 double AB = new Segment(a,b).length();  
 double BC = new Segment(b,c).length();  
 double AC = new Segment(a,c).length();  
 double p = (AB + BC + AC)/2;  
 return Math.*sqrt*(p\*(p-AB)\*(p-BC)\*(p-AC));  
 }  
  
 public Point getCentroid(Point a, Point b, Point c){  
 double Mx = (b.x + c.x)/2;  
 double My = (b.y + c.y)/2;  
 Point M = new Point(Mx,My);  
 System.*out*.println("Point M (mediana AM): (" + Mx + "," + My + ")");  
 double Ox = a.x + (Mx - a.x) \* 2.0/3.0;  
 double Oy = a.y + (My - a.y) \* 2.0/3.0;  
 return new Point(Ox,Oy);  
 }  
 @Override  
 public String toString(){  
 return "Triangle[A("+ a.x +","+ a.y +") B("+ b.x +","+ b.y +") C("+ c.x +","+ c.y +")]";  
 }  
}

**Клас Quadrangle** є розширенням класу Figure та описує об’єкт чотирикутник - quadrangle. На початку оголошуються чотири точки – координати вершин чотирикутника.

Конструктор класу створює об’єкт чотирикутник із заданими параметрами (точками вершин). Далі командою super(а) створюється точка – одна із вершин трикутника. Далі ініціалізуються всі вершини трикутника та створюються відрізки (ab, bc, cd, ad), з яких складається чотирикутник за допомогою класу Segment. Далі виконується перевірка того, чи існує такий чотирикутник, якщо фігура не існує то програма припиняє свою роботу.

Метод getArea() повертає площу чотирикутника, обраховану за формулою:



Метод getCentroid() повертає точку - центроїд чотирикутника. Точки a, b, c, d – вершини чотирикутника передаються в метод getCentroid() в якості параметрів. В кінці методу створюється об’єкт точка-центроїд, з координатами (Mх, Mу) і повертається в головну процедуру main.

Метод toString() виводить інформацію про об’єкт Чотирикутник (координати його вершин) на екран у вигляді строки.

Код класу Quadrangle:

public class Quadrangle extends Figure{  
 public Point a, b, c, d;  
 public Quadrangle(Point a, Point b, Point c, Point d){  
 super(a);  
 this.a = a;  
 this.b = b;  
 this.c = c;  
 this.d = d;  
 Segment ab = new Segment(a,b);  
 Segment bc = new Segment(b,c);  
 Segment cd = new Segment(c,d);  
 Segment ad = new Segment(a,d);  
 if ((ab.length()==0)||(bc.length()==0)||(cd.length()==0)||(ad.length()==0)) {  
 System.*out*.println("This Quadrangle NOT Exist");  
 Main.*figureExist* = false;  
 }  
 }  
  
 public static double getArea(Point a, Point b, Point c, Point d){  
 double area = 0.5\*((a.x\*b.y + b.x\*c.y + c.x\*d.y + d.x\*a.y) - (a.y\*b.x + b.y\*c.x + c.y\*d.x + d.y\*a.x));  
 return Math.*abs*(area);  
 }  
  
 public Point getCentroid(Point a, Point b, Point c, Point d){  
 double Mx = (a.x+b.x+c.x+d.x)/4;  
 double My = (a.y+b.y+c.y+d.y)/4;  
 return new Point(Mx,My);  
 }  
  
 public String toString(){  
 return "Quadrangle [A("+ a.x +","+ a.y +") B("+ b.x +","+ b.y +") C("+ c.x +","+ c.y +") D("+ d.x + "," + d.y +")]";  
 }  
}

**Клас Main** є головною процедурою яка об’єднує всі інші класи в єдину програму.

Першою командою оголошується логічна змінна figureExist яка необхідна дял визначення чи існує створена фігура.

В головній процедурі main спочатку створюється об’єкт centre класу Point – це точка яка є центром кола. Наступною командою створюється об’єкт коло класу Circle із заданими центром та радіусом. В конструкторі класу Circle виконується перевірка існування кола із заданим центром та радіусом, якщо таке коло не існує (радіус не є додатним числом) – видається відповідне повідомлення і головна процедура main припиняє роботу.

Далі створений об’єкт circle виводиться на екран за допомогою методу toString() класу Circle. Далі викликається метод circle.getArea(), який розраховує площу кола та інформація про площу кола виводиться на екран. Наступною командою на екран виводиться точка – центроїд кола – тобто його центр.

Наступний блок команд створює 3 точки (a, b, c)– вершини трикутника, потім створюється сам трикутник класу Triangle і в його конструкторі виконується перевірка на існування трикутника із заданими координатами вершин. Якщо такий трикутник не існує – головна процедура main припиняє свою роботу.

Потім трикутник виводиться на екран за допомогою методу toString() класу Triangle. Далі методом getArea() класу Triangle розраховується площа трикутника, і потім виводиться на екран. Наступною командою створюється об’єкт класу Point – точка центроїд трикутника, координати якої розраховуються методом getCentroid() класу Triangle. Накінець координати центроїда трикутника виводяться на екран.

Наступний блок команд створює 4 точки (a, b, c, d)– вершини чотирикутника, потім створюється сам чотирикутник класу Quadrangle і в його конструкторі виконується перевірка на існування чотирикутника із заданими координатами вершин. Якщо такий чотирикутник не існує – головна процедура main припиняє свою роботу.

Потім чотирикутник виводиться на екран за допомогою методу toString() класу Quadrangle. Далі методом getArea() класу Quadrangle розраховується площа чотирикутника, і потім виводиться на екран. Наступною командою створюється об’єкт класу Point – точка центроїд чотирикутника, координати якої розраховуються методом getCentroid() класу Quadrangle. Накінець координати центроїда чотирикутника виводяться на екран.

Код класу Main:

public class Main {  
 public static boolean *figureExist* = true;  
 public static void main(String[] args) {  
   
// Створення обєкта Circle по заданій точці-центру кола і радіусу:  
 Point centre = new Point(1, 2);  
 Circle circle = new Circle(centre, 3);  
 if (!*figureExist*){  
 System.*exit*(0);  
 }  
 System.*out*.println(circle);  
 System.*out*.println("Area of Circle: " + circle.getArea());  
 System.*out*.println("Centroid of Circle is (" + centre.x + ", " + centre.y + ")");  
  
// Створення обєкта Triangle по 3-х заданих точках:  
 Point a = new Point(0,0);  
 Point b = new Point(1,2);  
 Point c = new Point(2,0);  
 Triangle triangle = new Triangle(a, b, c);  
 if (!*figureExist*){  
 System.*exit*(0);  
 }  
 System.*out*.println(triangle);  
 System.*out*.println("Area of Triangle: " + Triangle.*getArea*(a, b, c));  
 Point centroid1 = triangle.getCentroid(a, b, c);  
 System.*out*.println("Centroid of Triangle: " + "(" + centroid1.x + ", "+ centroid1.y + ")");  
  
// Створення обєкта Quadrangle по 4-х заданих точках:  
 Point a1 = new Point(0,0);  
 Point b1 = new Point(1,2);  
 Point c1 = new Point(2,0);  
 Point d1 = new Point(1,-2);  
 Quadrangle quadrangle = new Quadrangle(a1, b1, c1, d1);  
 if (!*figureExist*){  
 System.*exit*(0);  
 }  
 System.*out*.println(quadrangle);  
 System.*out*.println("Area of Quadrangle is: " + Quadrangle.*getArea*(a1, b1, c1, d1));  
 Point centroid2 = quadrangle.getCentroid(a1, b1, c1, d1);  
 System.*out*.println("Centroid of Quadrangle is: (" + centroid2.x + ", " + centroid2.y + ")");  
 }  
}

**Висновки.** Під час виконання цієї лабораторної роботи я навчився основам абстрактного програмування в Java. Я навчився використовувати абстрактні класи та методи для створення загального шаблону поведінки, що реалізується в підкласах; ознайомився з поняттям інтерфейсів; навчився моделювати найпростіші геометричні фігури (трикутник, чотирикутник, коло) за допомогою створення класів, побудова конструкторів із перевірками, реалізація методів обчислення площі та центроїду фігур та перевизначення методів.

Посилання на GitHub репозиторій:

<https://github.com/diachenkom/OOP-KB-231-Diachenko-Maksym>