Programozás technológia

1.beadandó feladat: 7. feladat

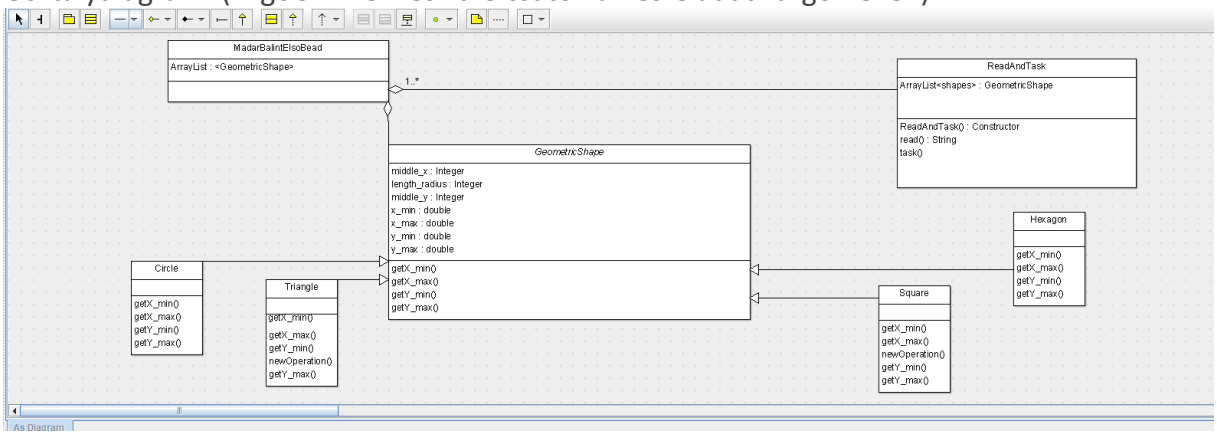
Készítette: Madár Bálint MRPBLG

A feladat leírása:

Rögzítsen a síkon egy pontot, és töltsön fel egy gyűjteményt különféle szabályos (kör, szabályos háromszög, négyzet, szabályos hatszög) síkidomokkal! Határozza meg a legkisebb téglalapot, amely lefedi az összes síkidomot és oldalai párhuzamosak a tengelyekkel! Minden síkidom reprezentálható a középpontjával és az oldalhosszal, illetve a sugárral, ha feltesszük, hogy a sokszögek esetében az egyik oldal párhuzamos a koordináta rendszer vízszintes tengelyével, és a többi csúcs ezen oldalra fektetett egyenes felett helyezkedik el. A síkidomokat szövegfájlból töltse be! A fájl első sorában szerepeljen a síkidomok száma, majd az egyes síkidomok. Az első jel azonosítja a síkidom fajtáját, amit követnek a középpont koordinátái és a szükséges hosszúság. A feladatokban a beolvasáson kívül a síkidomokat egységesen kezelje, ennek érdekében a síkidomokat leíró osztályokat egy közös ősosztályból származtassa!

A megoldás terve, az osztályokat és kapcsolataikat bemutató osztálydiagram

Osztálydiagram: (ArgoUML-el készítve csatolva 7esfeladat.zargo néven)



A megoldás terve:
GeometricShape közös absztrakt ősosztály
Circle a körök osztálya
Triangle szabályos háromszögek osztálya
Square négyzetek osztálya
Hexagon hatszögek osztálya
ReadAndTask a feladatot megvalósító publikus metódus ami a ReadAndTask osztályban található
ArrayList<GeometricShape> az alakzatokat tartalmazó ArrayList

A megoldáshoz szükséges típusok (osztályok) absztrakt leírása

GeometricShape: közös absztrakt ősosztály: ebből az osztályból vannak származtatva a speciális alakzatok, az osztály azért absztrakt Mivel ebből az osztályból nem lehet objektumokat példányosítani ezért absztraktnak választottam, habár getterje és setterje van. Ezen osztály általános tulajdonságokkal rendelkezik.

Az osztály változói

- middle_x ami egész típusú és az alakzat középpontjának x koordinátát adja meg;
- middle_y egy egész típus az adott alakzat középpontjának y koordinátát adja meg
- length_radius egy egész típus és az adott alakzat középpontjának oldalhasszát, vagy a sugarát adja meg
- x_min double típusú változó, az adott alakzat meddig ér minimum az x koordináta tengelyen
- x_max double típusú változó az adott alakzat meddig ér maximum az x koordináta tengelyen
- y_min double típusú változó, az adott alakzat meddig ér minimum az y koordináta tengelyen
- y_max double típusú változó, az adott alakzat meddig ér maximum az y koordináta tengelyen

  Az x_min, x_max, y_min, y_max változóknak az ősosztályban van egy általános getterjük, ami felül van definiálva a származtatott osztályokban.

A **GeometricShape** aggregációval kapcsolódik a MadarBalintElsoBead osztályhoz, mivel egy komponense az egész programnak.

Circle a körök speciális osztálya ami a ***GeometricShape ősosztályból származik***
Triangle szabályos háromszögek speciális osztálya ***GeometricShape ősosztályból származik***
Square négyzetek osztálya ***GeometricShape ősosztályból származik***
Hexagon hatszögek osztálya ***GeometricShape ősosztályból származik***

A **Circle, Triangle, Square, Hexagon s**zármaztatott osztályosztályok, melyeknek általános osztálya (ősosztály, superclass) a **GeometricShape.**
Azért az Általánosítás és specializációt választottam mivel a kör a háromszög és a négyzet illetve a hatszög mind egy alakzat. Ezen alakzatok mind rendelkeznek az általános tulajdonságokkal (középpont, oldalhossz/sugár), ezáltal a speciális osztályok meg megöröklik az általános tulajdonságokat. Minden speciális osztálynak a x_min, x_max, y_min, y_max változó getterjét felül kell definiálnia, mert minden alakzat esetén máshogy kapom meg az értékeket.
A felüldefiniált getter metódusok leírása a program kódban az adott alakzat kommentjénél található.
A specializáció ebben az esetben többszörös, mivel egy általánosításból több származott osztály jön létre.
ReadAndTask egy publikus osztály amiben a read() és a task() metódus található, a read() metódusban a fálj beolvasása valósul meg a task() metódusban a feladat megvalósítása történik. Aggregációval kapcsolódik a MadarBalintElsoBead osztályhoz mivel egy komponense az egész programnak.
A **task()** metódus részletes leírása kódban található
A főprogram a MadarBalintElsoBead osztályban található: részletes leírása kódban található

Lényeges importált Java osztályok:

Fájlbeolvasáshoz:

- import java.io.BufferedReader;
- import java.io.FileNotFoundException;
- import java.io.FileReader;
- import java.util.ArrayList;
- import java.util.Scanner;

Kivételkezeléshez:

- import java.io.FileNotFoundException;
- import java.util.NoSuchElementException;

Részletes tesztelési terv

Tesztfájlok:

1. empty.txt
2. first.txt
3. second.txt
4. third.txt
5. badfile.txt

A hibák elkapását a try-catch szerkezettel végeztem, az elkapott hibák

1. FileNotFoundException
2. InvalidInputException
3. NoSuchElementException

Tesztesetek

Hiba tesztek

1. empty.txt: Empty file input!
   A tesztelő a NoSuchElementException ágra fut
2. xyz.txt: File not found!
   Nem létező fájl beírása a FileNotFoundException ág fut le.
3. badfile.txt tartalma 3 x y z enterrel elválasztva
   Invalid input! hibát ír a konzol a InvalidInputException ág fut le

Normális működés tesztek

1. first.txt
   P(0,0) kezdőponttal rendelkező 4 alakzat

   Bemenet:
   4
   c 0 0 15
   t 0 0 14
   s 0 0 13
   h 0 0 12

   Kimenet:
   A lekisebb téglalap ami lefedi a síkidomot, hossza: 30.0 és magassága: 30.0
   A lekisebb téglalap ami lefedi a síkidomot, középpontja: (0.0, 0.0)
   A téglalap 4 pontja:
   A téglalap A pontja (-15.0,15.0)
   A téglalap B pontja: (-15.0,-15.0)
   A téglalap C pontja: (15.0,15.0)
   A téglalap D pontja: (15.0,-15.0)

2. second.txt Különböző középponttal rendelkező 6 alakzat
   Bementet:
   6
   s 3 3 8
   c 2 1 4
   h 4 3 2
   t 5 5 12
   t 4 6 3
   s 5 7 5

   Kimenet
   A lekisebb téglalap ami lefedi a síkidomot, hossza: 8.0 és magassága: 8.0
   A lekisebb téglalap ami lefedi a síkidomot, középpontja: (3.0, 3.0)
   A téglalap 4 pontja:
   A téglalap A pontja (-1.0,7.0)
   A téglalap B pontja: (-1.0,-1.0)
   A téglalap C pontja: (7.0,7.0)
   A téglalap D pontja: (-1.0,-1.0)
3. triangle.txt P(0,) középponttal rendelkező 1 darab négyzet alakzat

   Bemenet:

   s 5 5 3

   Kimenet
   A lekisebb téglalap ami lefedi a síkidomot, hossza: 1.0 és magassága: 1.5

   A lekisebb téglalap ami lefedi a síkidomot, középpontja: (0.0, -0.25)

   A téglalap 4 pontja:

   A téglalap A pontja (-0.5,0.375)

   A téglalap B pontja: (-0.5,-1.125)

   A téglalap C pontja: (0.5,0.375)

   A téglalap D pontja: (-0.5,-1.125)

4. *square.txt* P(0,0) ponttal rendelkező háromszög alakzat
   Bemenet: s 0 0 1
   Kimenet:
   A lekisebb téglalap ami lefedi a síkidomot, hossza: 1.0 és magassága: 1.0
   A lekisebb téglalap ami lefedi a síkidomot, középpontja: (0.0, 0.0)
   A téglalap 4 pontja:
   A téglalap A pontja (-0.5,0.5)
   A téglalap B pontja: (-0.5,-0.5)
   A téglalap C pontja: (0.5,0.5)
   A téglalap D pontja: (0.5,-0.5)

5. *hexagon.txt P(0,0)*
   Bemenet h 0 0 1
   Kimenet:
   A lekisebb téglalap ami lefedi a síkidomot, hossza: 2.0 és magassága: 1.7320508075688772
   A lekisebb téglalap ami lefedi a síkidomot, középpontja: (0.0, 0.1339745962155614)
   A téglalap 4 pontja:
   A téglalap A pontja (-1.0,0.8660254037844386)
   A téglalap B pontja: (-1.0,-0.8660254037844386)
   A téglalap C pontja: (1.0,0.8660254037844386)
   A téglalap D pontja: (1.0,-0.8660254037844386)
6. *Round.txt*
   Bemenet: c 0 0 1
   Kimenet:
   A lekisebb téglalap ami lefedi a síkidomot, hossza: 2.0 és magassága: 2.0
   A lekisebb téglalap ami lefedi a síkidomot, középpontja: (0.0, 0.0)
   A téglalap 4 pontja:
   A téglalap A pontja (-1.0,1.0)
   A téglalap B pontja: (-1.0,-1.0)
   A téglalap C pontja: (1.0,1.0)
   A téglalap D pontja: (1.0,-1.0)