**SPRAWOZDANIE**

Zajęcia: Grafika komputerowa

Prowadzący: prof. dr hab. Vasyl Martsenyuk

**Laboratorium** 1

22.02.2022

**Temat:** Przekształcanie 2D w bibliotece Java 2D

**Wariant:** zadanie 1 – 15-kąt, zadanie 2 – figura 3.

Natalia Pierzchała

Informatyka I stopień,

stacjonarne,

4 semestr,

Gr. 2a

# Polecenie:

Program Transform2D.java rysuje obraz shuttle.jpg w panelu. Narysować zamiast obrazu wielokąt według wariantu (liczba n) w panelu wyświetlania.

Panel ma wymiary 600 na 600 pikseli, a wielokąt ma promie ́n 150 pikseli. Okno zawiera również wyskakujące menu z etykieta ”Transform:”.

Opcje w menu to ”None” i cyfry od 1 do 9. W tym programie menu wyskakujące nie działa. Zadanie polega na dodaniu kodu do metody paintComponent (). (Miejsce jest oznaczone TODO.) Kiedy wybór ma wartość 0, strona powinna wyświetlać obraz nietransformowany. W przypadku innych możliwych wartości musisz zastosować przekształcenie (lub będziesz potrzebował kombinacji przekształceń) dla każdej z wartości od 1 do 9 (patrz Fig. 1).

2. Narysować figurę określoną wariantem (patrz Fig. 2). Dostępne są trzy podstawowe kształty: circle (), square () i triangle (). Zacznij od programu TransformedShapes.java. TODO. Możesz użyć poleceń do rysowania, takich jak g.fillRect () itp.

# Wprowadzane dane:

W zadaniu pierwszym należało utworzyć figurę składającą się z 15 kątów. Współrzędna zostały wyliczone za pomocą wzorów i pętli for.

Zadanie drugie polegało na odtworzeniu podanej figury, która w tym przypadku składała się z trójkątów i prostokąta. Figura została ustawiona na środku okna wynikowego, poprzez przestawienie w odpowiednie miejsca każdego elementu.

# Wykorzystane komendy:

1. Zadanie 1

* Generowanie figury

**int**[] xpts = **new** **int**[15];

**int**[] ypts = **new** **int**[15];

**for**(**int** i = 1; i < 16; i++)

{

xpts[i-1] = (**int**) (200\*Math.*cos*((2\*Math.***PI***/15)\*i));

}

**for**(**int** i = 1; i < 16; i++)

{

ypts[i-1] = (**int**) (200\*Math.*sin*((2\*Math.***PI***/15)\*i));

}

Polygon pietnastokat = **new** Polygon(xpts, ypts, 15)

g2.setColor(Color.***black***);

g2.fillPolygon(pietnastokat);

* Transformacje figury

**switch**(whichTransform)

{

**case** 0: **break**;

**case** 1:

g2.scale(0.5,0.5);

**break**;

**case** 2:

g2.rotate(Math.*toRadians*(45));

**break**;

**case** 3:

g2.scale(-0.5,0.8);

g2.rotate(Math.*toRadians*(180));

**break**;

**case** 4:

g2.shear(0.35,0);

**break**;

**case** 5:

g2.translate(0,-225);

g2.scale(1,0.5);

**break**;

**case** 6:

g2.shear(0,-0.5);

g2.rotate(Math.*toRadians*(90));

**break**;

**case** 7:

g2.scale(0.5, 1);

g2.rotate(Math.***PI***);

**break**;

**case** 8:

g2.translate(0,130);

g2.rotate(Math.*toRadians*(30));

g2.scale(1,0.5);

**break**;

**case** 9:

g2.translate(100, 0);

g2.shear(0, 0.25);

g2.rotate(Math.*toRadians*(180));

**break**;

}

1. Zadanie 2

[***private***](https://github.com/Mieri013/GrafikaKomputerowa/tree/main/lab1/lab0101)***void*** *triangle() {*

*g2.fillPolygon(****new******int****[] {-50,50,0},* ***new******int****[] {50,50,-50}, 3);*

***protected******void*** *paintComponent(Graphics g) {*

***super****.paintComponent(g);*

*g2 = (Graphics2D)g.create();*

*g2.setRenderingHint(RenderingHints.****KEY\_ANTIALIASING****, RenderingHints.****VALUE\_ANTIALIAS\_ON****);*

*g2.translate(300,175);*

*g2.setColor(Color.****blue****);*

*g2.rotate(Math.****PI****);*

*g2.scale(1,1);*

*triangle();*

*resetTransform();*

*g2.translate(300,275);*

*g2.scale(2.25,1);*

*square();*

*resetTransform();*

*g2.translate(300,375);*

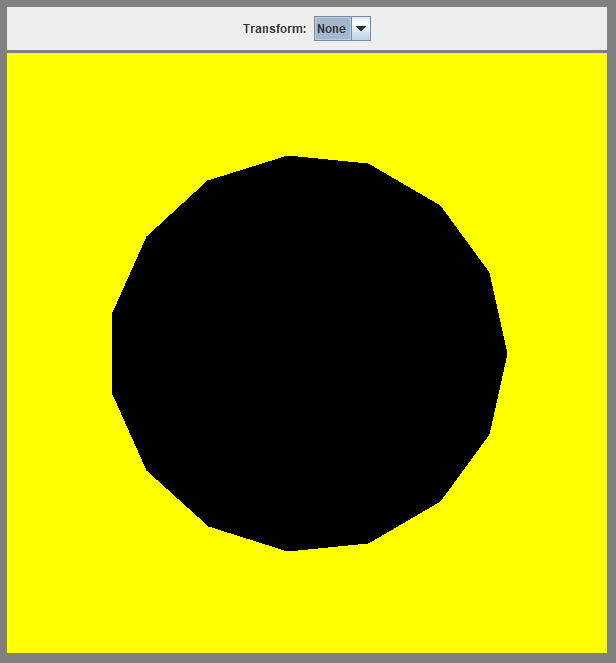
*g2.scale(1,1);*

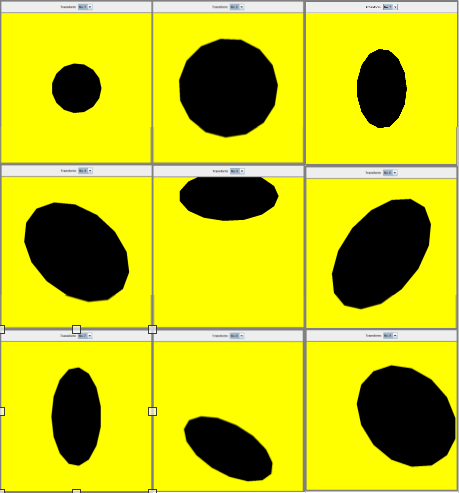
*triangle();*

*resetTransform();*

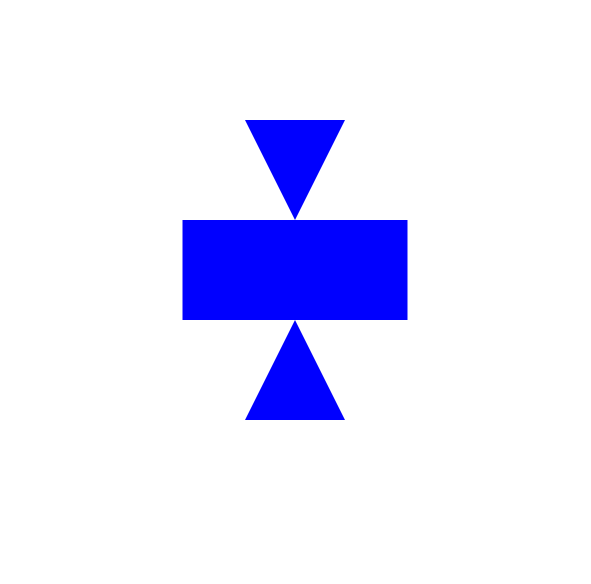
# Wyniki działania

Zadanie 1





Zadanie 2



# Wnioski

Na podstawie zadania pierwszego można stwierdzić, że korzystając z podstawowych metod z klasy Graphics biblioteki JAVA 2D możemy stworzyć proste rysunki np. figury.

Z zadania drugiego możemy wywnioskować, że pisząc funkcję square() określamy współrzędne powstałej figury. Figurę możemy wywołać wielokrotnie, ustalać jej położenie, transformować np. obracać i skalować.