SPRAWOZDANIE

Zajęcia: Grafika komputerowa

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium: 1

Data: 24.02.2020r

Temat: Przekształcenia 2D w bibliotece Java 2D

Wariant: 17-kąt, figura 1

Imię Nazwisko: Marek Żyła

Informatyka I stopień,

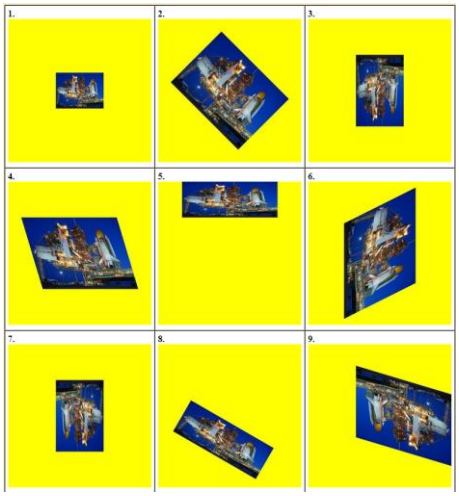
stacjonarne,

4 semestr, Gr.1b

1. Polecenie:

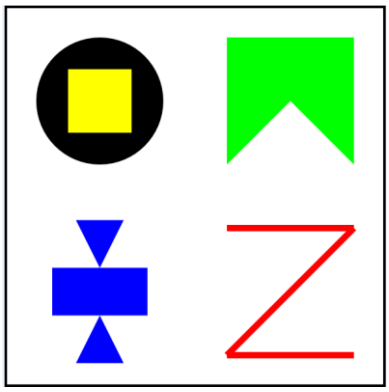
Zad.1.

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 r ownie z yskakujące menu z etykieta ”Transform:”.Opcje w menu to ”None” i cyfry od 1 do 9. W tym programie menu yskakujące nie dzia la. Zadanie polega na dodaniu kodu do metody paintComponent (). (Miejsce jest oznaczone TODO.) Kiedy wybór mawartość 0, strona powinna wy swietla c obraz nietransformowany. W przypadku innych mo zliwych wartości musisz zastosować przekształcenie (lub będziesz potrzebowa l kombinacji przeksztа lce n) dla ka zdej z wartości od 1 do 9 (patrz Fig. 1).



Zad.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.



2. Wprowadzone Dane:

Zad.1. transformacje shuttle.jpg

```
// TODO Apply transforms here, depending on the value of whichTransform!
if(whichTransform==1)
{
    g2.scale(0.5, 0.5);
}
switch(whichTransform)
{
case 1: g2.scale(0.5, 0.5);
        break;
case 2: g2.rotate(3.14/4);
        break;
case 3: g2.scale(-0.25, 0.75);
        g2.rotate(3.14);
        break;
case 4: g2.shear(0.5, 0.0);
    break;
case 5: g2.scale(1, 0.5);
        g2.translate(0, -450);
    break;
case 6: g2.shear(0.0, -0.5);
        g2.rotate(3.14/2);
    break;
case 7: g2.scale(0.25, 0.75);
        g2.rotate(3.14);
    break;
case 8: g2.rotate(3.14/8);
        g2.scale(1, 0.5);
        g2.translate(0, 250);
    break;
case 9: g2.shear(0.0, 0.5);
g2.scale(-1.0, -1.0);
g2.translate(-110, 50);
    break;
}
//g2.drawImage(pic, -200, -150, null); // Draw image with center at (0,0).
```

Rysowanie 17-kąta

```
int promien = 150; //starting point
Polygon siedemnastokat = new Polygon();

//start
siedemnastokat.addPoint(promien, 0);

for(int i = 1; i<18; i++)
{
siedemnastokat.addPoint((int)(promien*Math.cos(2*i*Math.PI /17)), (int)(promien*Math.sin(2*i*Math.PI /17)));
}

g2.drawPolygon(siedemnastokat);
```

Zad.2

```
// TODO Draw the required image, using ONLY the four methods defined above,
// along with g2.setColor, g1.scale, g2.translate, and g2.rotate.

g2.setColor(Color.black);
  g2.translate(150,150);
  g2.scale(2.2,2.2);
  circle();

  resetTransform();
  g2.setColor(Color.yellow);
  g2.translate(150,150);
  g2.scale(1.1,1.1);
  square();
/* ---------------------------------------------------------------------- */
```

3. Wykorzystane komendy

Zad.1.

Shear(), rotate(), scale(), translate(), drawpolygon(), addPoint(,)

Zad.2.

resetTransform();

setColor(Color.yellow);

translate(150,150);

scale(1,1);

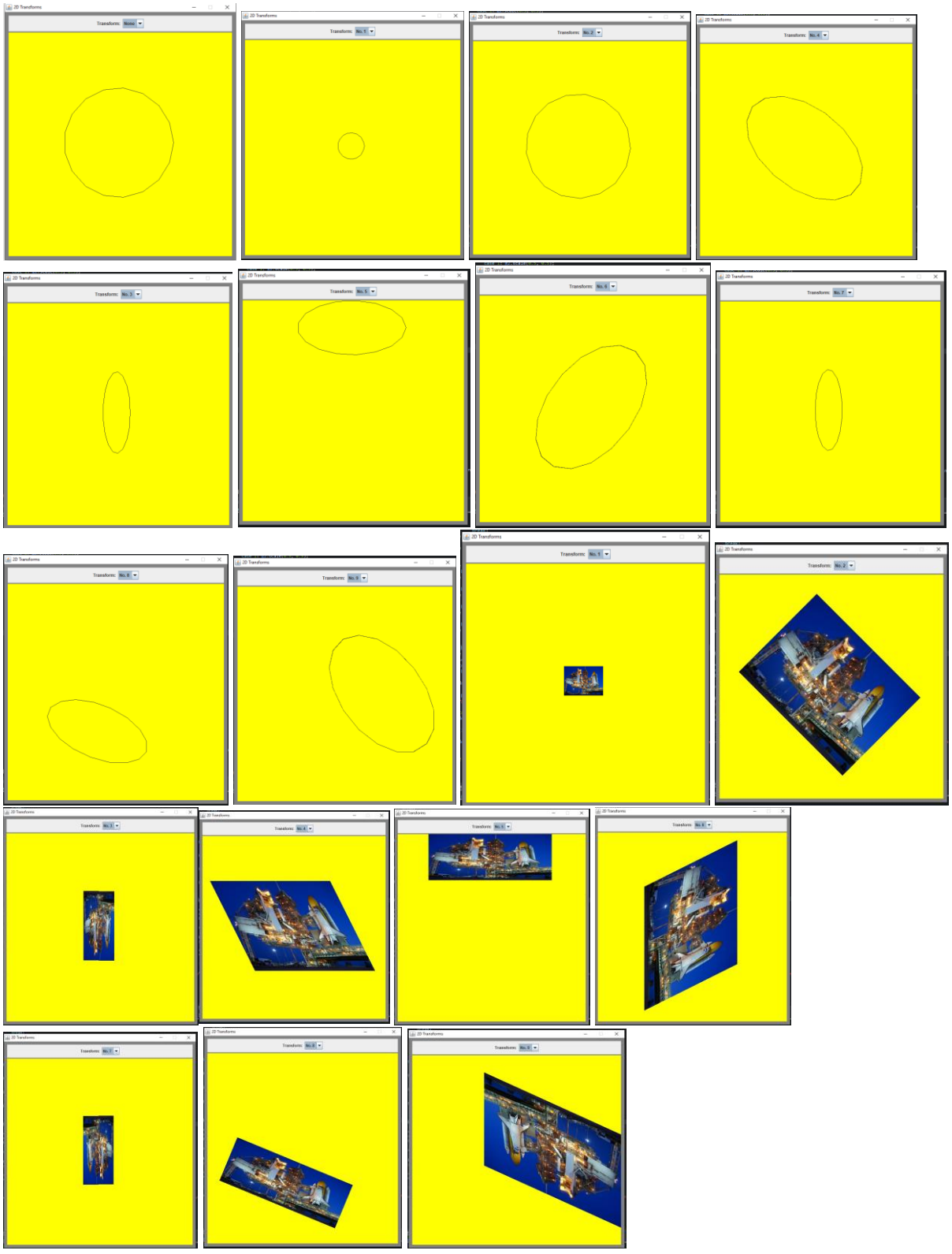
square();

circle();

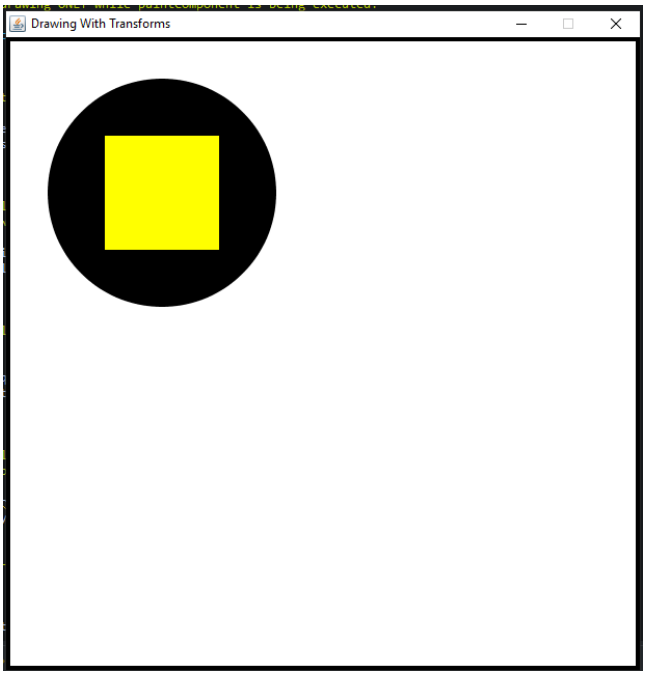
https://github.com/Samo276/GK_lab_1

4. Wynik Działania:

Zad.1.



Zad.2.



5. Wnioski:

Na podstawie zadania 1 można stwierdzić że nie zależnie czy do transformacji podstawiamy zdjęcie czy narysowany przez nas obiekt można go tak samo transformować.

Przy rysowaniu obiektu w zadania 2 co będzie leżeć na czym (kolejność warstw) zależy od kolejności tworzenia obiektów.

Przy użyciu zaledwie kilku komend można wprowadzić drastyczne zmiany w wyglądzie obrazu poprzez zastosowanie ich różnych kombinacji.