SPRAWOZDANIE

Zajęcia: Grafika komputerowa

Prowadzący: prof. dr hab. Vasyl Martenyuk

Laboratorium 1
28.02.2023
Temat: “Przekształcenia 2D w bibliotece Java 2D”
Wariant: 15-kąt, figura nr 3

Jakub Światłoch
Informatyka I stopień,
stacjonarne,
4 semestr,
Gr.2a

Pokazuję tylko kod który został zmodyfikowany, reszta kodu jest taka sama jaka została podana na zajęciach.

ZADANIE 1

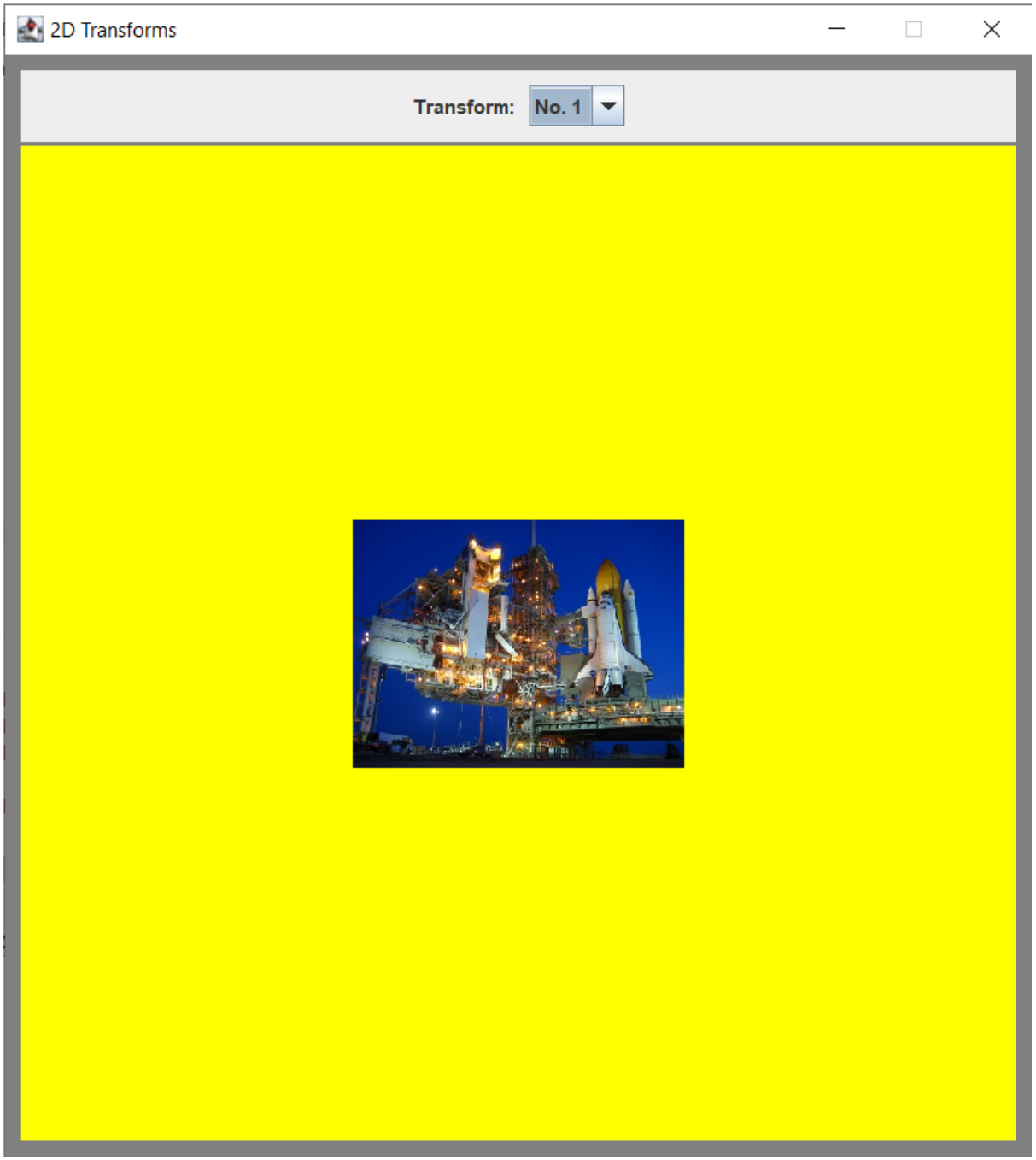
1. **Polecenie**: Zadanie polega na dodaniu kodu do metody paintComponent (). Kiedy wybór ma wartość 0, strona powinna wyświetlać obraz nie transformowany. W przypadku innych mo˙zliwych wartości musisz zastosować przekształcenie dla każdej z wartości od 1 do 9
2. **Wprowadzane dane**: brak
3. **Wykorzystane komendy wraz z opisem**:

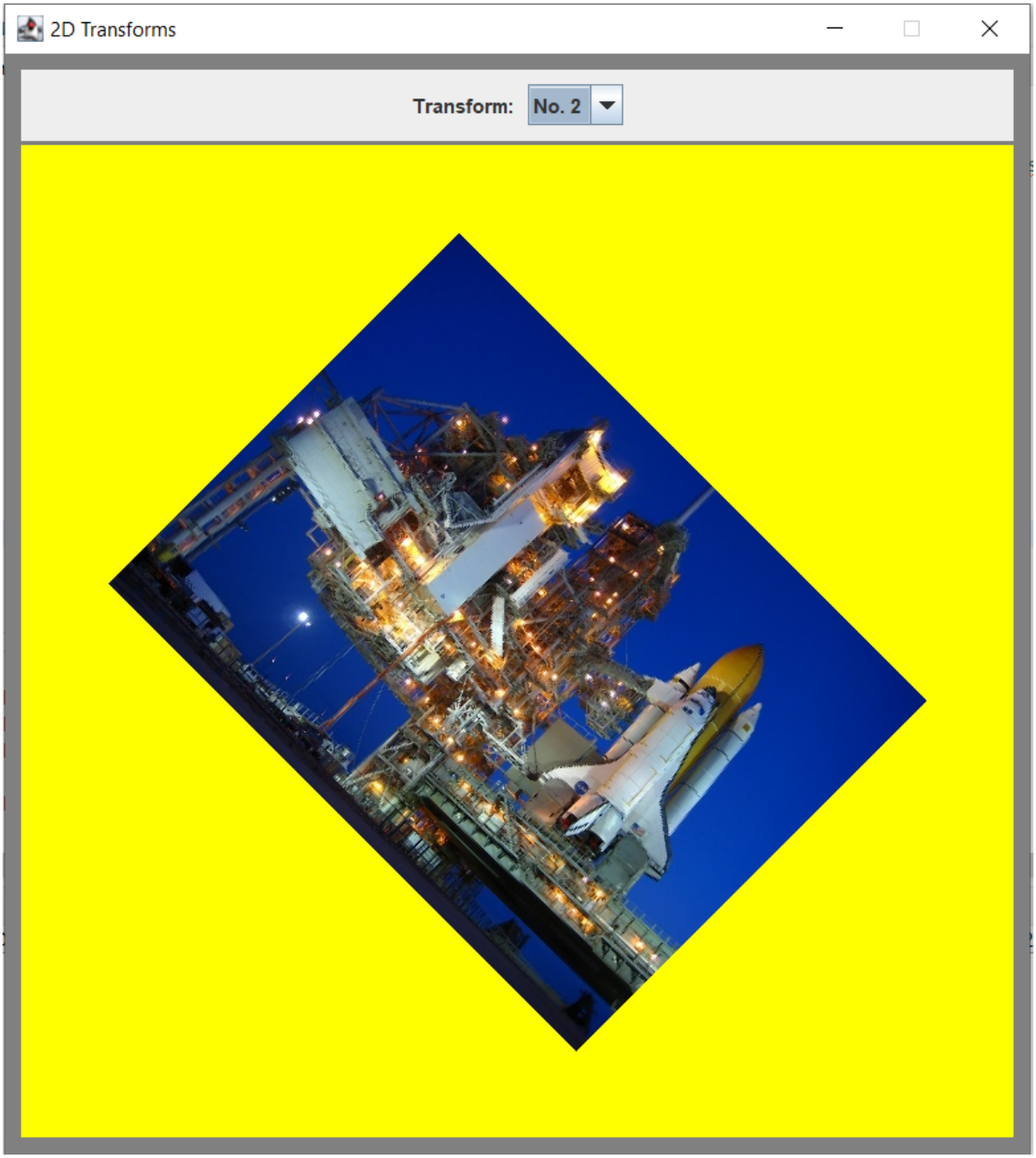
```
27          // Wybór odpowiedniej transformacji na podstawie wartości zmiennej whichTransform
28          int whichTransform = transformSelect.getSelectedIndex();
29          switch (whichTransform) {
30              case 1:
31                  // Skalowanie o 50%
32                  g2.scale(0.5, 0.5);
33                  break;
34              case 2:
35                  // Obrót o 45 stopni
36                  g2.rotate(Math.PI/4);
37                  break;
38              case 3:
39                  // Obrót o 180 stopni, skalowanie o 50% w osi X, odbicie lustrzane w osi Y
40                  g2.rotate(Math.PI);
41                  g2.scale(0.5, 1);
42                  g2.scale(-1, 1);
43                  break;
44              case 4:
45                  // Zniekształcenie przez pochylenie w osi X o 50%
46                  g2.shear(0.5, 0);
47                  break;
48              case 5:
49                  // Skalowanie o 50% w osi Y, przesunięcie w górę o 450 pikseli
50                  g2.scale(1, 0.5);
51                  g2.translate(0, -450);
52                  break;
53              case 6:
54                  // Obrót o 90 stopni, zniekształcenie przez pochylenie w osi X o 50%
55                  g2.rotate(Math.PI/2);
56                  g2.shear(0.5, 0);
57                  break;
58              case 7:
59                  // Obrót o 180 stopni, skalowanie o 50% w osi X
60                  g2.rotate(Math.PI);
61                  g2.scale(0.5, 1);
62                  break;
63              case 8:
64                  // Przesunięcie o (-50, 100), obrót o 30 stopni, skalowanie o 100% w osi X i 50% w osi Y
65                  g2.translate(-50, 100);
66                  g2.rotate(Math.toRadians(30));
67                  g2.scale(1.0, 0.5);
68                  break;
69              case 9:
70                  // Obrót o 180 stopni, zniekształcenie przez pochylenie w osi Y o 25%, przesunięcie o (-100, 50)
71                  g2.rotate(Math.PI);
72                  g2.shear(0, 0.25);
73                  g2.translate(-100, 50);
74                  break;
75              default:
76                  // Brak transformacji
77                  break;
78          }
```

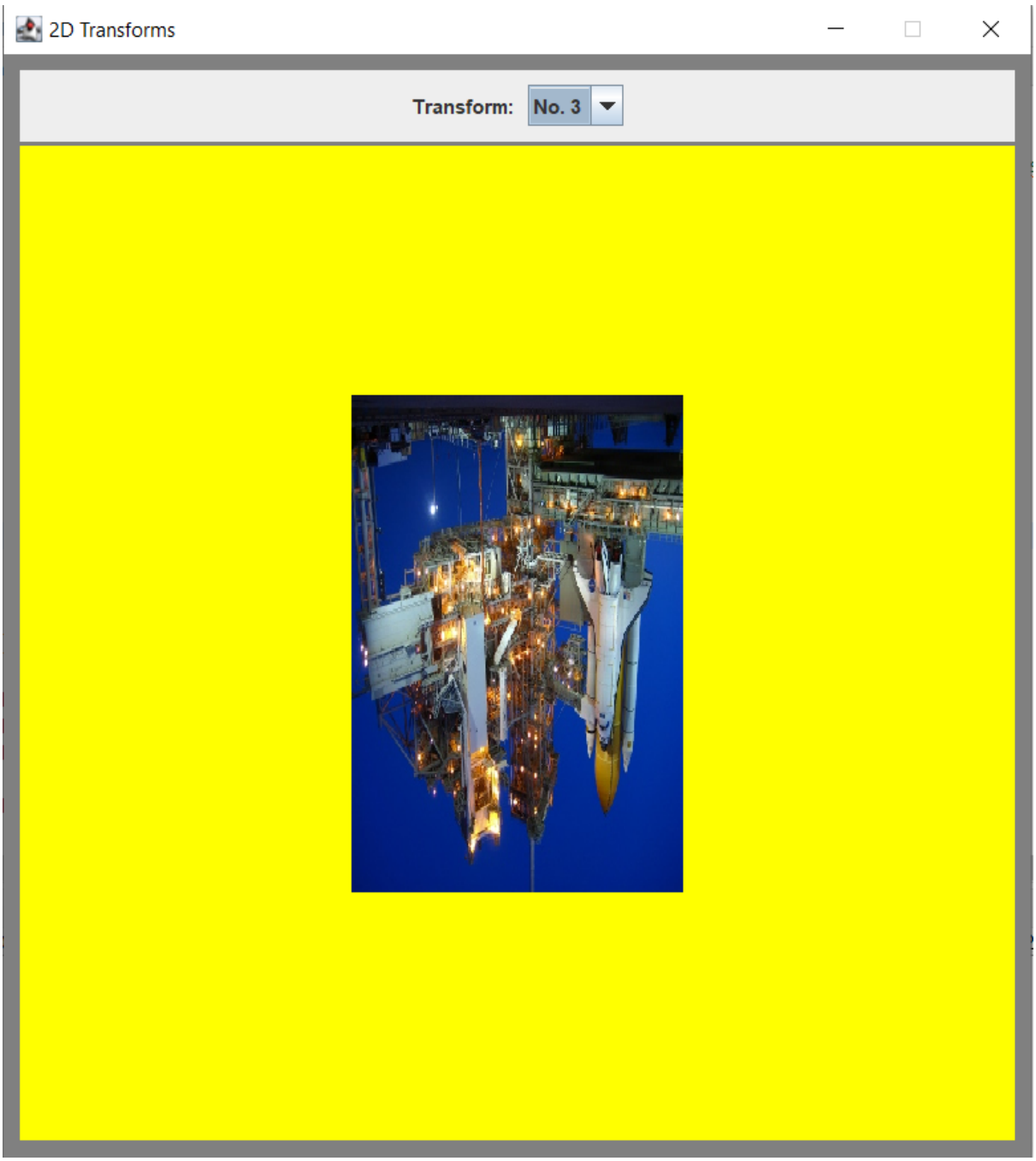
Math.PI = 180 stopni, odpowiednio ją dzieląc np Math.PI / 2 można uzyskać 90 stopni, dzięki czemu nie trzeba używać Math.toRadians(x)

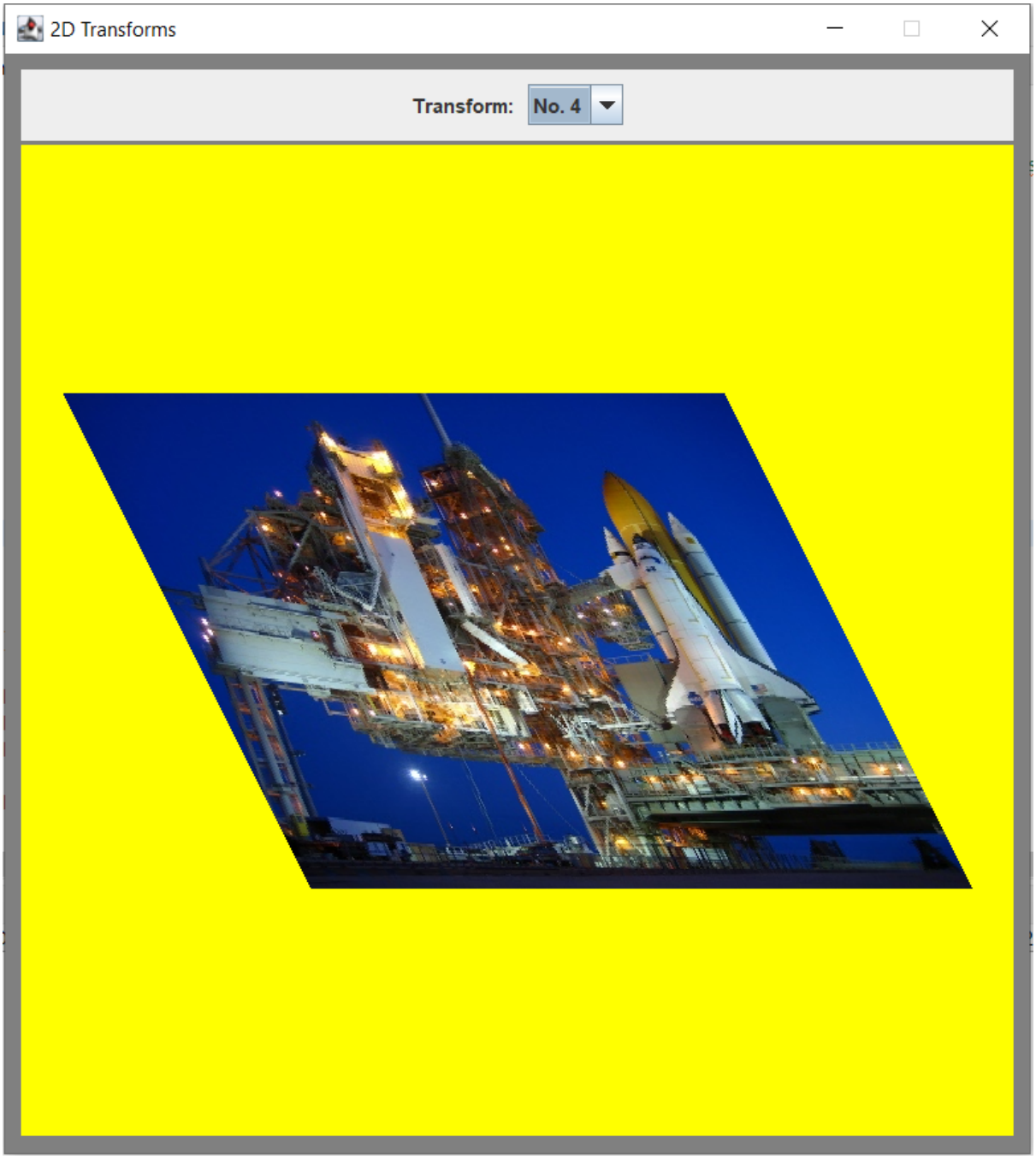
Link do GitHub w którym znajduje się kod z zadania oraz wyniki zadania
https://github.com/JakubSwiatlochgit/GKcw1

4. **Wynik działania**:

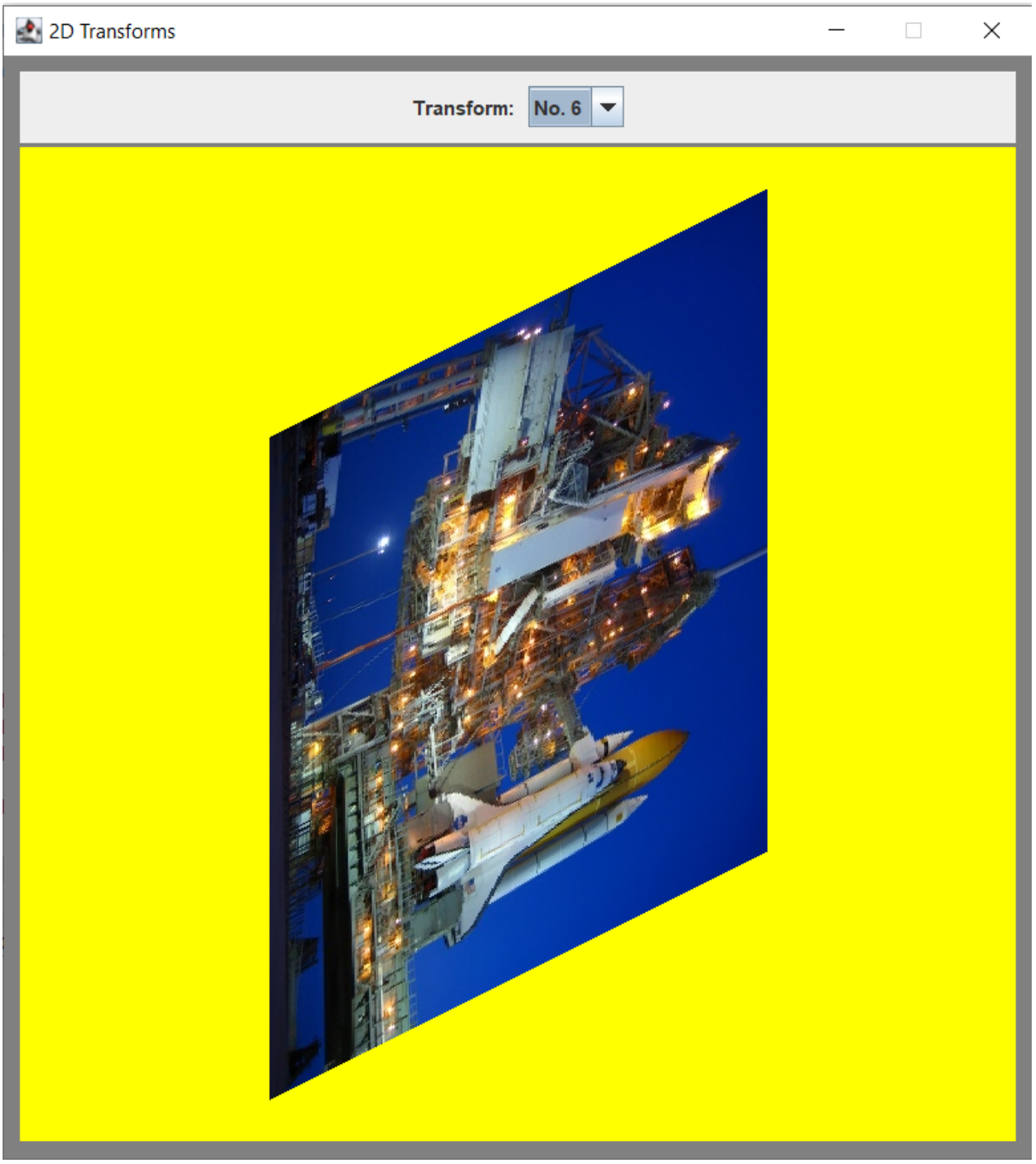

2D Transforms
Transform:
No. 1

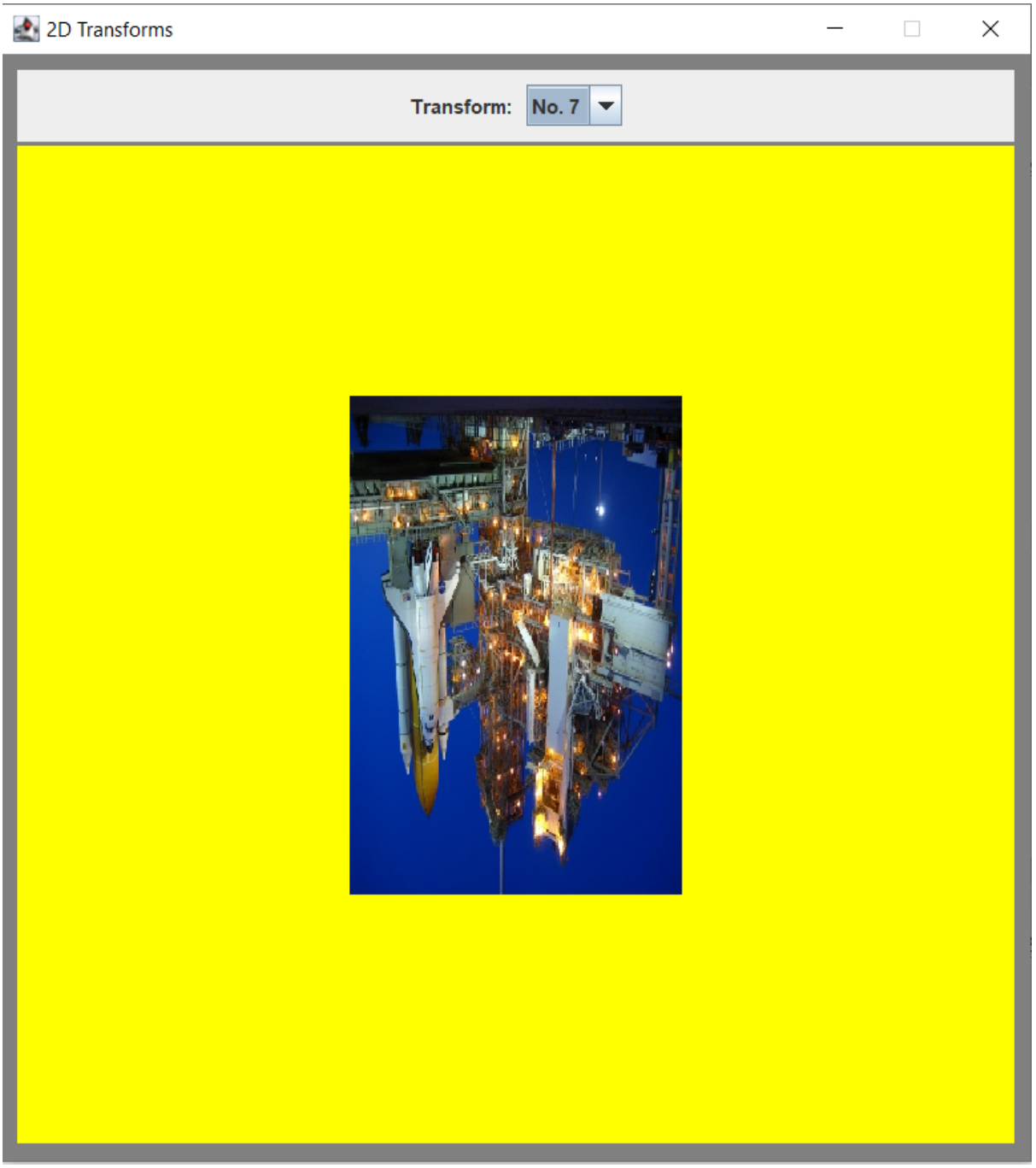

2D Transforms
Transform:
No. 2


2D Transforms
Transform:
No. 3

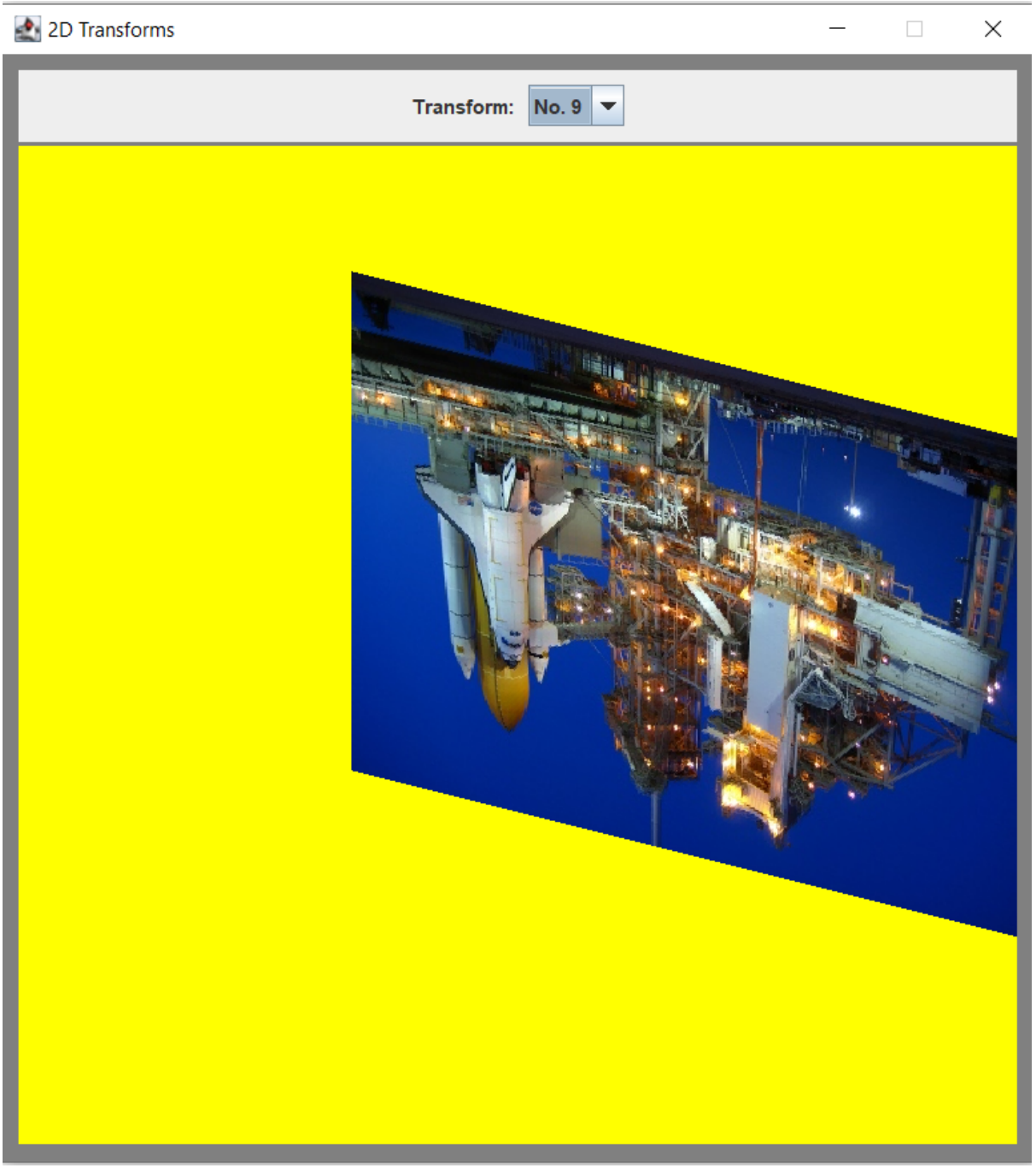

2D Transforms
Transform:
No. 4


2D Transforms
Transform:
No. 5


2D Transforms
Transform:
No. 6


2D Transforms
Transform:
No. 7


2D Transforms
Transform:
No. 8



5. **Wnioski**:

Ten kod to fragment programu napisanego w języku Java, który rysuje grafikę 2D i pozwala na wybór różnych transformacji geometrycznych. Transformacja wybierana jest z listy rozwijanej transformSelect.

Kod wykorzystuje instrukcję switch do wyboru odpowiedniej transformacji na podstawie wybranej wartości z listy rozwijanej. Każda transformacja jest zdefiniowana za pomocą metod klasy Graphics2D, takich jak rotate, scale, shear i translate, które wykonują odpowiednie operacje przekształcenia geometrycznego na grafice.

W przypadku, gdy użytkownik nie wybierze żadnej transformacji, program nie wykonuje żadnych przekształceń.

Ogólnie kod wykorzystuje prostą logikę i składnię języka Java do wykonania różnych przekształceń geometrycznych na grafice 2D w zależności od wyboru użytkownika.

ZADANIE 2

1. **Polecenie**: Narysuj piętnastokąt
2. **Wprowadzane dane**:

R - to promień piętnastokąta
xPoints - tablica przechowująca współrzędne x wierzchołków piętnastokąta
yPoints - tablica przechowująca współrzędne y wierzchołków piętnastokąta
t - ze wzoru jest obliczany kąt obrotu dla i-tego wierzchołka

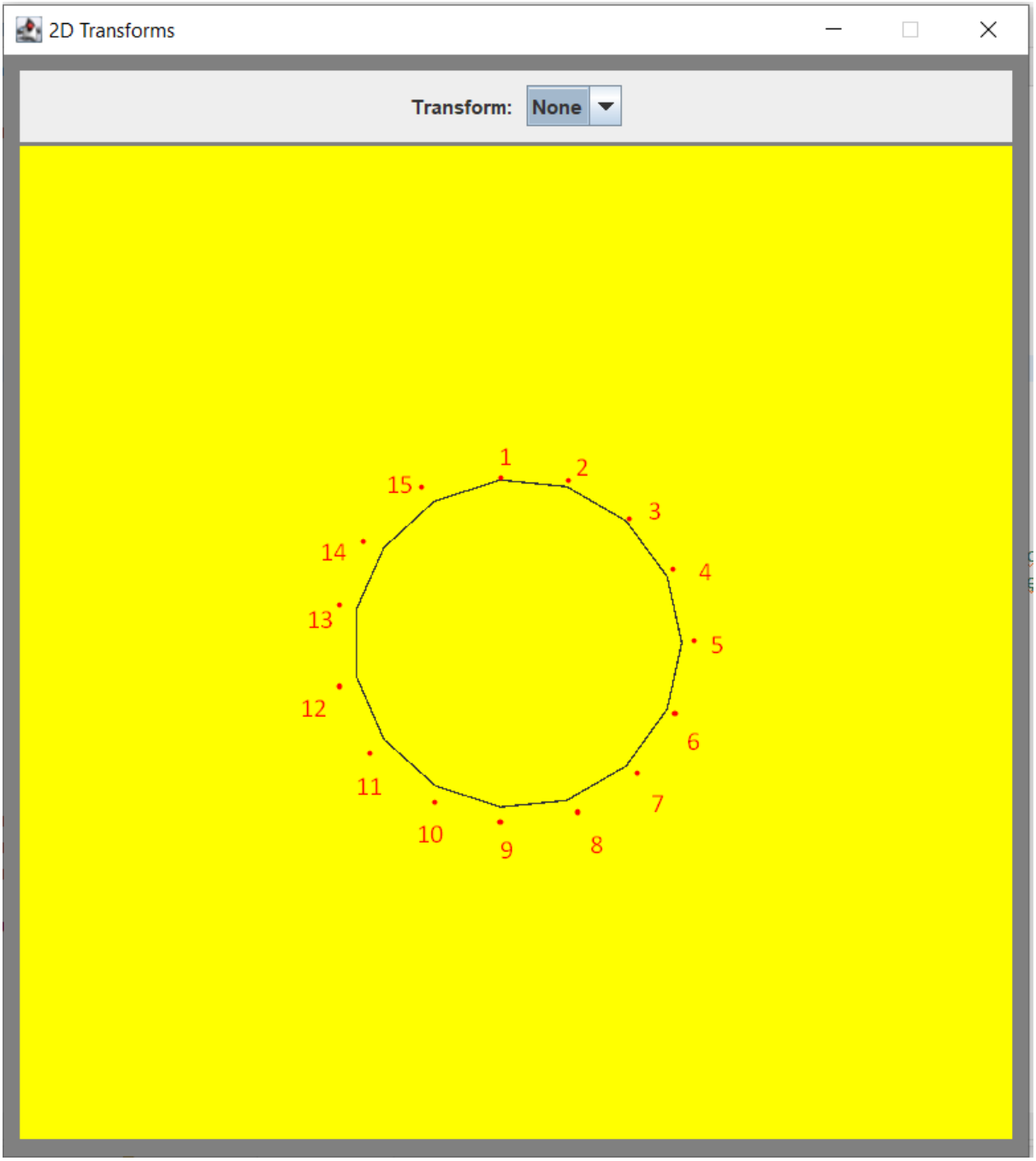
3. **Wykorzystane komendy**:

```
    private static final long serialVersionUID = 1L;

    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        Graphics2D g2 = (Graphics2D)g;
        g2.translate(300*0,300*0);  // Moves (0,0) to the center of the display.


        int R = 100; // promień piętnastokąta
        int[] xPoints = new int[15]; // tablica przechowująca współrzędne x wierzchołków piętnastokąta
        int[] yPoints = new int[15]; // tablica przechowująca współrzędne y wierzchołków piętnastokąta

        // Pętla tworząca tablice z współrzędnymi wierzchołków piętnastokąta
        for (int i = 0; i < 15; i++) {
            double t = i * 2 * Math.PI / 15; // kąt obrotu dla i-tego wierzchołka
            // obliczenie współrzędnej x i-tego wierzchołka
            xPoints[i] = (int) (R * Math.cos(t)) + getWidth() / 2;

            // obliczenie współrzędnej y i-tego wierzchołka
            yPoints[i] = (int) (R * Math.sin(t)) + getHeight() / 2;

        }
        // narysowanie piętnastokąta na ekranie przy użyciu współrzędnych z tablic xPoints i yPoints.
        g2.drawPolygon(xPoints, yPoints, 15);



        //g2.drawImage(pic, -200, -150, null); // Draw image with center at (0,0).
    }
}
```

Link do GitHub w którym znajduje się kod z zadania oraz wyniki zadania
https://github.com/JakubSwiatlochgit/GKcw1

Ten kod rysuje piętnastokąt (wielokąt o piętnastu bokach) o promieniu 100 pikseli. W pętli for obliczane są położenia wierzchołków piętnastokąta na podstawie wzorów matematycznych dla współrzędnych x i y. Następnie te punkty są zapisywane w tablicach xPoints i yPoints. Na końcu g2.drawPolygon rysuje piętnastokąt na ekranie z użyciem tablic zapisanych wcześniej punktów.

4. **Wyniki działania:**



5. **Wnioski:**

Rysowanie piętnastokąta jest możliwe przy użyciu funkcji trygonometrycznych.
Dzięki użyciu pętli for, możliwe jest zautomatyzowanie rysowania wielu figur o podobnej geometrii.
Rysowanie figur geometrycznych wymaga korzystania z właściwych narzędzi i metod dostępnych w danej bibliotece programistycznej.

ZADANIE 3

1. **Polecenie**: Narysować figurę określoną wariantem. 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.
   Do narysowania jest figura nr 3
2. **Wprowadzane dane**: brak
3. **Wykorzystane komendy wraz z opisem**:

Na potrzeby programu metody zostały zmodyfikowane :

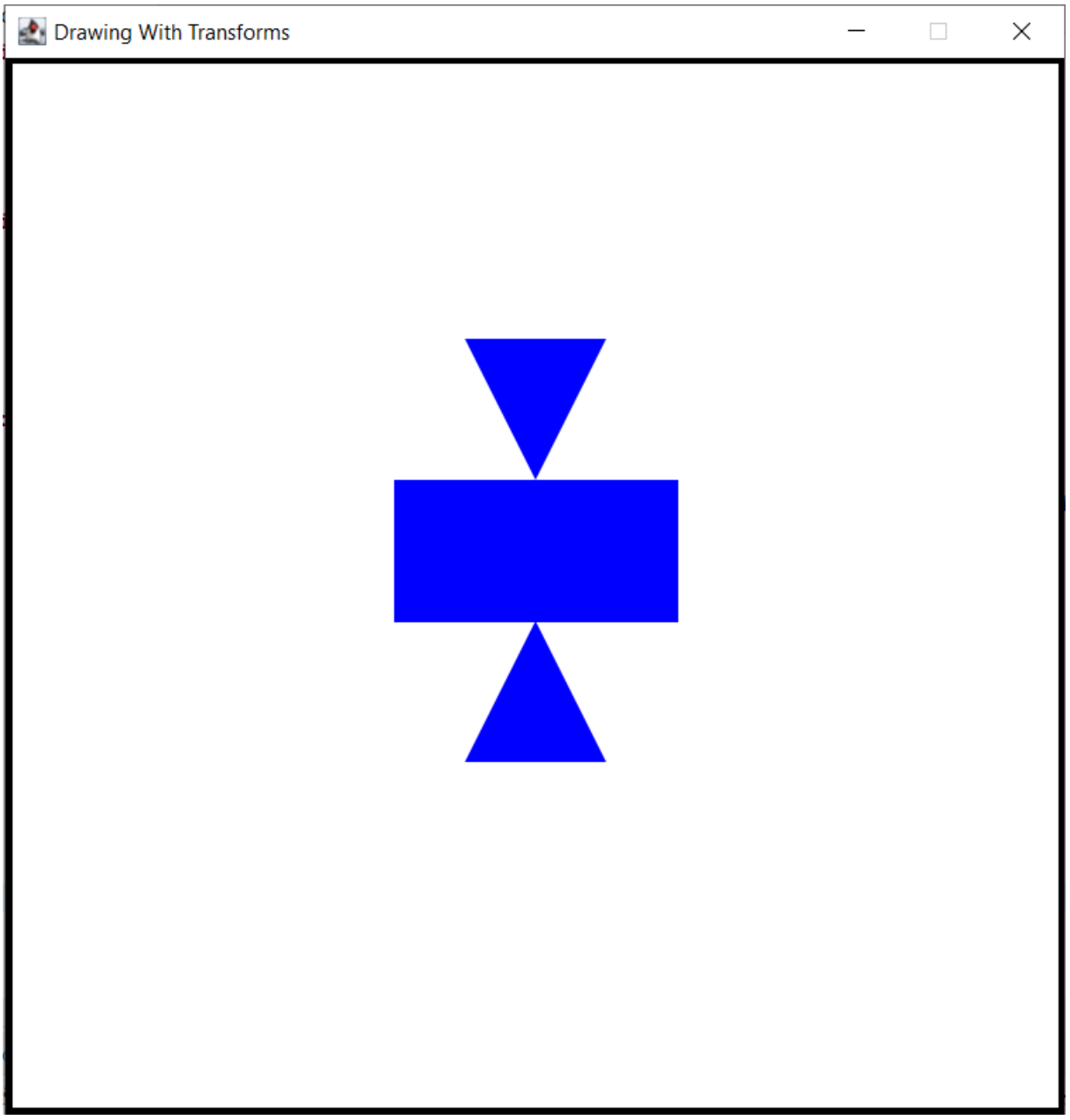
```
private Graphics2D g2;

private void resetTransform() {
    // ustawia transformację na wartość domyślną
    g2.setTransform(new AffineTransform());
}

private void triangle() {
    // rysuje trójkąt o wierzchołkach w punktach (0, -50), (-50, 50), (50, 50)
    int[] xPoints = { 0, -50, 50 };
    int[] yPoints = { -50, 50, 50 };
    g2.fillPolygon(xPoints, yPoints, 3);
}

private void square() {
    // rysuje kwadrat o szerokości 200 i wysokości 100 w punkcie (0,0)
    int width = 200;
    int height = 100;
    g2.drawRect(0, 0, width, height);
    g2.fillRect(0, 0, width, height);
}
```

Główna część programu rysowania tej figury:

```
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    g2 = (Graphics2D) g.create();
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);
    //ustawia kolor g2 na niebieskie
    g2.setColor(Color.BLUE);
    //resetuje transformację do wartości domyślnej
    resetTransform();

    // przesuwa układ o (275, 300) i rysuje kwadrat
    g2.translate(275,300);
    square();

    // przesuwa układ o (100, -50), obraca o 180 stopni i rysuje trójkąt
    g2.translate(100, -50);
    g2.rotate(Math.PI);
    triangle();

    // obraca układ o 180 stopni i przesuwa o wektor (0, 200), następnie rysuje trójkąt
    g2.rotate(Math.PI);
    g2.translate(0, 200);
    triangle();

    // resetuje transformację do wartości domyślnej
    resetTransform();

}
```

Link do GitHub w którym znajduje się kod z zadania oraz wyniki zadania
https://github.com/JakubSwiatlochgit/GKcw1

4. **Wyniki działania:**



5. **Wnioski:**

Klasa TransformedShapes dziedziczy po klasie JPanel i zawiera metody do rysowania kwadratu i trójkąta, wykorzystując Graphics2D.

W metodzie paintComponent tworzony jest obiekt Graphics2D i ustawiane są odpowiednie właściwości do rysowania grafiki, takie jak wygładzanie krawędzi.

Metoda resetTransform(), ustawia transformację na wartość domyślną, co pozwala rysować kształty od razu na płótnie.

Przy użyciu metod translate() i rotate() ustawiane są odpowiednie przesunięcia i obroty, po których rysowane są kwadraty i trójkąty.

Klasa TransformedShapes jest wykorzystywana w aplikacji okienkowej z interfejsem użytkownika, w której wyświetlane są rysunki.