# SPRAWOZDANIE

Zajęcia: Grafika komputerowa
Prowadzący: prof. dr hab. Vasyl Martsenyuk

**Laboratorium II**
Data
**Temat: Zadanie PYGAME**

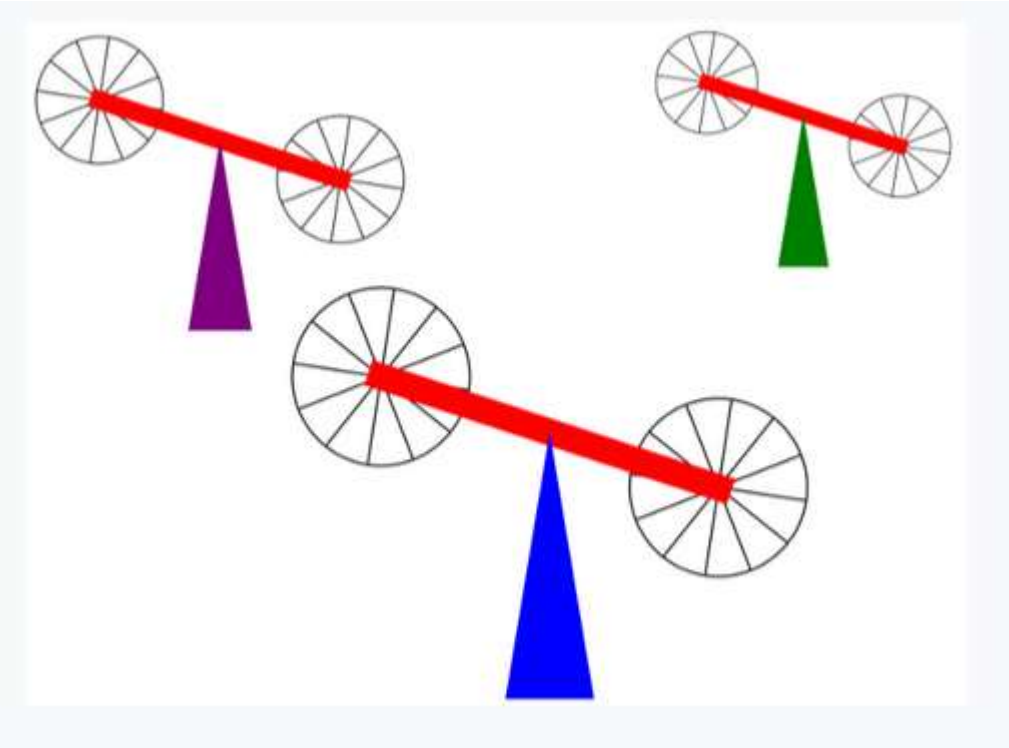
**Wariant 8 + 4**

Jakub Bąk
Informatyka I stopień,
stacjonarne,
4 semestr,
Gr.3b

# 1. Polecenie

Opracować scenę hierarchiczną zgodnie z obrazem używając zamiast kół wielokąty obracające się (animacja!) według wariantu na dwa sposoby:
(a) używając hierarchiczne funkcje (sposób subrutynowy)
(b) tworząc graf sceny (sposób obiektowy).

# 2. Wprowadzane dane:



Na podstawie podanego obrazka, należało napisać funkcje za pomocą sposobu subrutonyowego i obiektowego. Do programu zostały dołączone przykładowe pliki.

## 3. Wykorzystane komendy:

Link do github: https://github.com/Szeladin/grafika.git

Fragment kodu do sposobu graf sceny, sposób obiektowy

```
var canvas;
    var graphics;
    var X_LEFT = -4;
    var X_RIGHT = 4;
    var Y_BOTTOM = -3;
    var Y_TOP = 3;
    var BACKGROUND = "white";  // The display is filled with this color before the scene is drawn.
    var pixelSize;  // The size of one pixel, in the transformed coordinates.

    var frameNumber = 0;  // Current frame number. goes up by one in each frame.

    var world;  // A SceneGraphNode representing the entire scene.
    var blueWeight, purpleWeight, greenWeight;

    function createWorld() {
        world = new CompoundObject();  // Root node for the scene graph.

        // Create the blue weight
        blueWeight = new TransformedObject(weight);
        blueWeight.setColor("blue");
        world.add(blueWeight);

        // Create the purple weight on the left with scale 0.7
        purpleWeight = new TransformedObject(weight);
        purpleWeight.setScale(0.7, 0.7).setTranslation(-2, 0).setColor("purple");
        world.add(purpleWeight);

        // Create the green weight on the right with scale 0.5
        greenWeight = new TransformedObject(weight);
        greenWeight.setScale(0.5, 0.5).setTranslation(2, 0).setColor("green");
        world.add(greenWeight);
    }

```

W podejściu obiektowym do tworzenia grafu sceny, struktura kodu jest zorganizowana w klasy i obiekty, które reprezentują różne elementy sceny.

Fragment kodu z podejścia hierarchicznego:

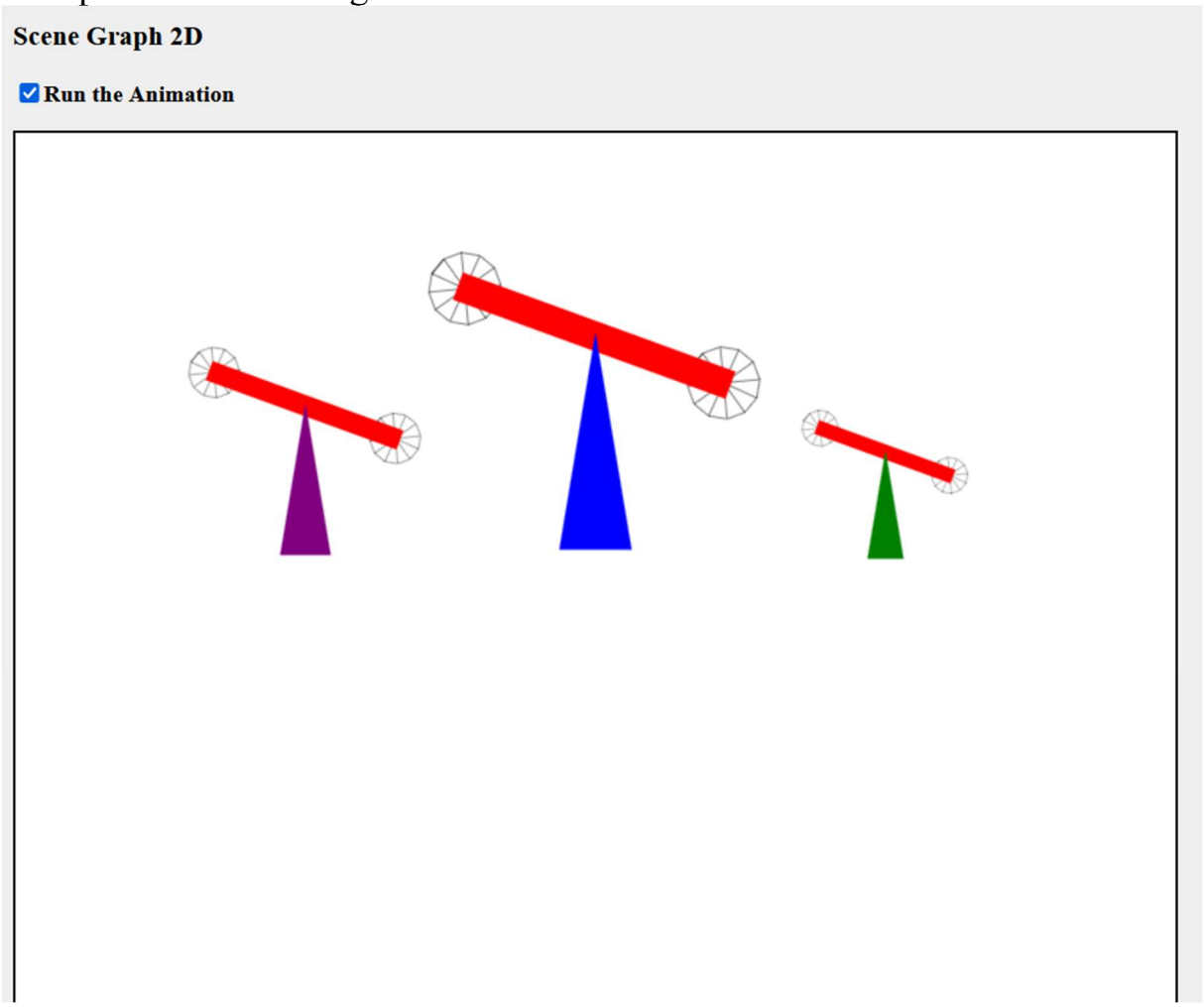
```
var canvas;
    var graphics;
    var X_LEFT = -4;
    var X_RIGHT = 4;
    var Y_BOTTOM = -3;
    var Y_TOP = 3;
    var BACKGROUND = "white";
    var pixelSize;
    var frameNumber = 0;

    function drawWorld() {
```

```
12.
13.         graphics.save();
14.         graphics.scale(1, 1);
15.         drawWeight("blue");
16.
17.         // Draw the purple weight on the left of the blue weight with scale 0.7
18.         graphics.save();
19.         graphics.translate(-2, 0);  // Adjust the position to the left
20.         graphics.scale(0.7, 0.7);
21.         drawWeight("purple");
22.
23.         // Draw the green weight on the right of the blue weight with scale 0.5
24.         graphics.save();
25.         graphics.translate(2, 0);  // Adjust the position to the right
26.         graphics.scale(0.5, 0.5);
27.         drawWeight("green");
28.         graphics.restore();
29.     }
30.
```

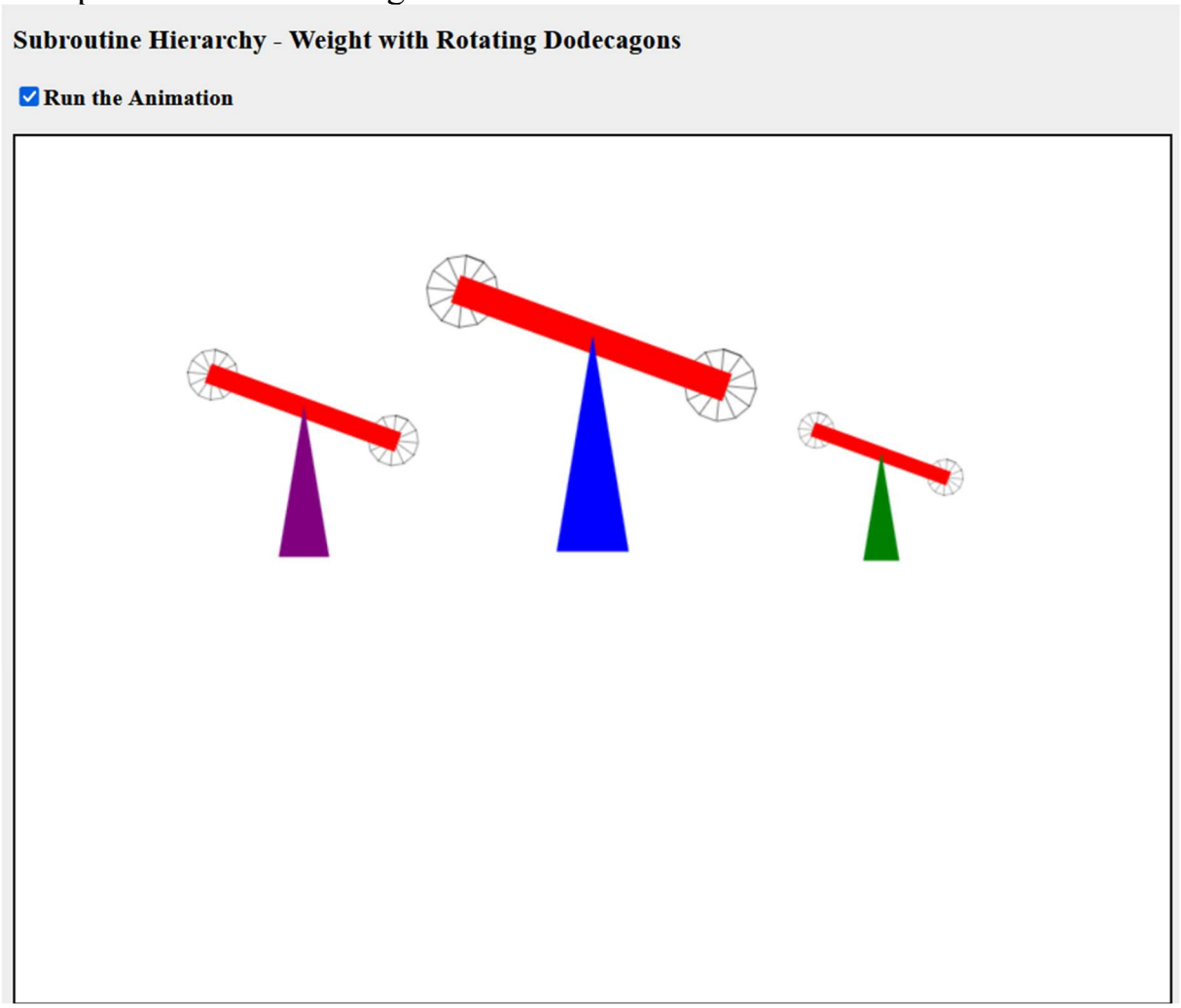
Podejście hierarchiczne do rysowania sceny polega na wywoływaniu funkcji w określonej kolejności, aby zbudować złożone obiekty z prostszych elementów.

## 4. Wyniki działania

Dla sposobu obiektowego:



Dla sposobu hierarchicznego:




## 5. Wnioski:

Podejście obiektowe do tworzenia grafu sceny pozwala na łatwe zarządzanie złożonymi scenami poprzez hierarchię obiektów i transformacji. Każdy obiekt w scenie może być niezależnie transformowany i rysowany, co ułatwia tworzenie złożonych animacji i scen. W podejściu hierarchicznym do rysowania sceny, struktura kodu jest zorganizowana w funkcje, które są wywoływane w określonej kolejności, aby zbudować złożone obiekty z prostszych elementów.