**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

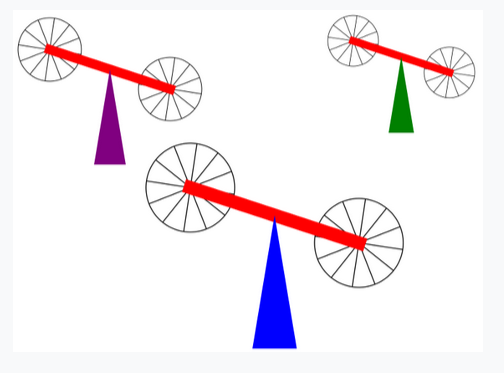
# Polecenie

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).

# Wprowadzane dane:



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

# Wykorzystane komendy:

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

Fragment kodu do sposobu graf sceny, sposób obiektowy

1. var canvas;

2.     var graphics;

3.     var X\_LEFT = -4;

4.     var X\_RIGHT = 4;

5.     var Y\_BOTTOM = -3;

6.     var Y\_TOP = 3;

7.     var BACKGROUND = "white";  // The display is filled with this color before the scene is drawn.

8.     var pixelSize;  // The size of one pixel, in the transformed coordinates.

9.

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

11.

12.     var world;  // A SceneGraphNode representing the entire scene.

13.     var blueWeight, purpleWeight, greenWeight;

14.

15.     function createWorld() {

16.         world = new CompoundObject();  // Root node for the scene graph.

17.

18.         // Create the blue weight

19.         blueWeight = new TransformedObject(weight);

20.         blueWeight.setColor("blue");

21.         world.add(blueWeight);

22.

23.         // Create the purple weight on the left with scale 0.7

24.         purpleWeight = new TransformedObject(weight);

25.         purpleWeight.setScale(0.7, 0.7).setTranslation(-2, 0).setColor("purple");

26.         world.add(purpleWeight);

27.

28.         // Create the green weight on the right with scale 0.5

29.         greenWeight = new TransformedObject(weight);

30.         greenWeight.setScale(0.5, 0.5).setTranslation(2, 0).setColor("green");

31.         world.add(greenWeight);

32.     }

33.

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:

1. var canvas;

2.     var graphics;

3.     var X\_LEFT = -4;

4.     var X\_RIGHT = 4;

5.     var Y\_BOTTOM = -3;

6.     var Y\_TOP = 3;

7.     var BACKGROUND = "white";

8.     var pixelSize;

9.     var frameNumber = 0;

10.

11.     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();

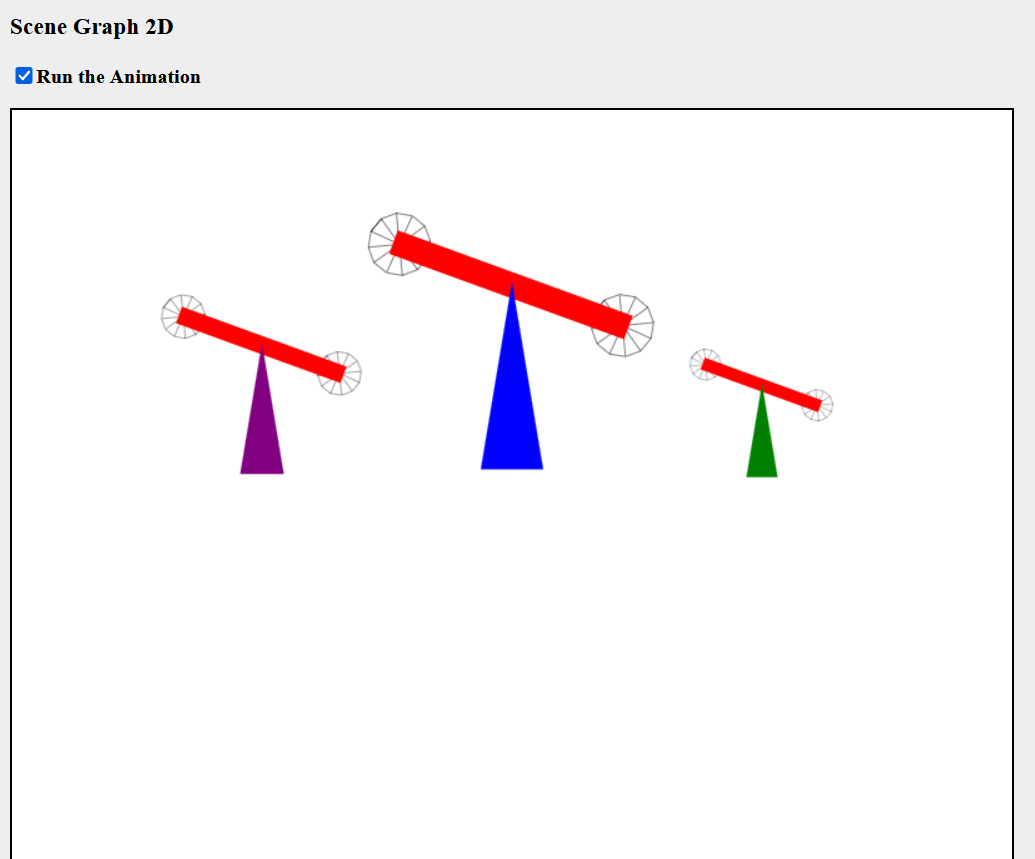
29.     }

30.

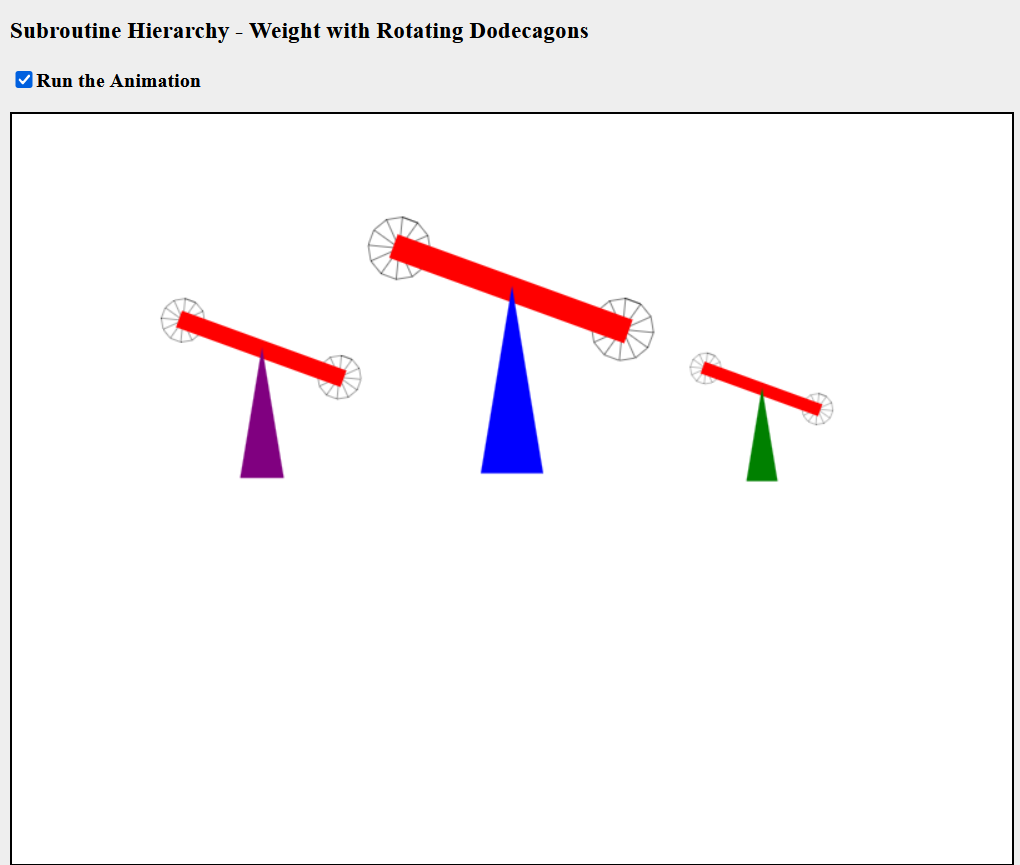
Podejście hierarchiczne do rysowania sceny polega na wywoływaniu funkcji w określonej kolejności, aby zbudować złożone obiekty z prostszych elementów.

# Wyniki działania

Dla sposobu obiektowego:



Dla sposobu hierarchicznego:



# 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.