SPRAWOZDANIE

Zajęcia: Grafika Komputerowa
Prowadzący: mgr inż. Mikołaj Grygiel

Laboratorium: 6
Data: 26.03.2025
Temat: " Geometria trójwymiarowa OpenGL"

Wariant: 14

Illia Bryka,
Informatyka I stopień,
stacjonarne,
4 semestr,
Gr.1a

Zadanie 1

1. Polecenie:

Stworzyć dwa obiekty przy użyciu OpenGL (w języku C lub Java). Po uruchomieniu zakończonego programu naciśnięcie jednego z klawiszy numerycznych 1 lub 2 spowoduje wybranie wyświetlanego obiektu. Program już ustawia wartość zmiennej globalnej, objectNumber, aby powiedzieć, który obiekt ma zostać narysowany. Użytkownik może obracać obiekt za pomocą klawiszy strzałek, PageUp, PageDown i Home. Podprogram display() jest wywoływany, aby narysować obiekt. Podprogram ten z kolei wywołuje draw() i właśnie w draw() powinieneś wykonać podstawową pracę. (Miejsce jest oznaczone TODO.) Dodaj również kilka nowych podprogramów do programu. Obiekt 1. Korkociąg wokół osi {x | y | z} zawierający 14 obrotów. Punkty są stopniowo powiększane. Ustalić aktualny kolor rysujący na {zielony | niebieski | brązowy | … }. Obiekt 2. Pyramida, wykorzystując dwa wachlarze trójkątów oraz modelowanie hierarchiczne (najpierw tworzymy podprogramę rysowania jednego trójkonta; dalej wykorzystując przekształcenia geometryczne tworzymy pyramidę). Podstawą pyramidy jest wielokąt o 14 wierzchówkach.

2. Wprowadzane dane:

Do zadania został wykorzystany wariant otrzymany od prowadzącego (14-kąt).

3. Wykorzystane komendy:

Do wykonania zadania należało zmodyfikować kod w Javie.

Wyświetlanie:

```
function doKeyDown(evt) {
  let key = evt.keyCode;
  if ( key == 49 ) { // klawisz '1'
    objectNumber = 1;
  }
  else if ( key == 50 ) { // klawisz '2'
    objectNumber = 2;
  }
  else if ( key == 37 ) // lewo
     rotateY -= 15;
  else if ( key == 39 ) // prawo
     rotateY += 15;
  else if ( key == 40) // dół
     rotateX += 15;
  else if ( key == 38 ) // góra
     rotateX -= 15;
  else if ( key == 33 ) // Page Up
     rotateZ += 15;
  else if ( key == 34 ) // Page Down
     rotateZ -= 15;
  else if ( key == 36 )  // Home
     rotateX = rotateY = rotateZ = 0;

  if ( (key >= 34 && key <= 40) )
    evt.preventDefault();
  display();
}
```

```
function display() {
  glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
  glLoadIdentity();
  // Ustawienie globalnych rotacji
  glRotatef(rotateZ,0,0,1);
  glRotatef(rotateY,0,1,0);
  glRotatef(rotateX,1,0,0);
  // Wybór obiektu na podstawie wartości globalnej objectNumber
  if (objectNumber === 1)
    drawSpiral();
  else if (objectNumber === 2)
    drawPyramid();
}
```

Spirala:

```
function drawSpiral() {
  glColor3f(0, 1, 0);
  glBegin(GL_LINE_STRIP);
  let turns = 14;
  let numPoints = turns * 100;
  // Parametr theta od 0 do turns*2PI
  for (let i = 0; i <= numPoints; i++) {
    let theta = i * turns * 2 * Math.PI / numPoints;
    let radius = 0.05 + 0.005 * theta;
    let x = radius * Math.cos(theta);
    let y = radius * Math.sin(theta);
    let z = 0.05 * theta;
    glVertex3f(x, y, z);
  }
  glEnd();
}
```

Trójkąt, piramida:

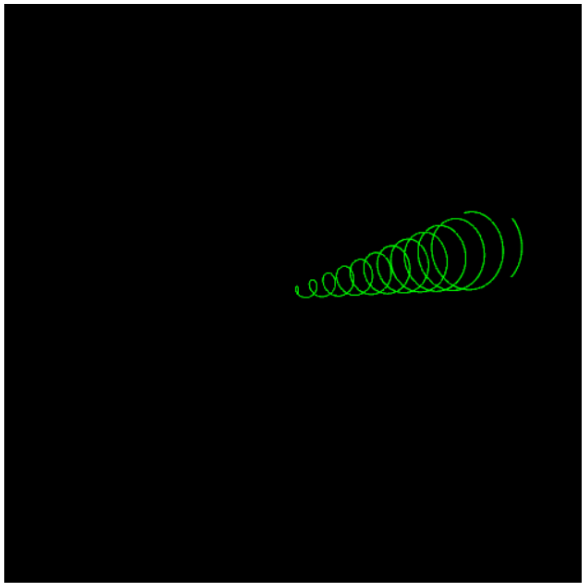
```
function drawPyramid() {

  glColor3f(0, 0, 1);
  glBegin(GL_TRIANGLE_FAN);
    glVertex3f(0, 0, 0);
    let sides = 14;
    let angleStep = 2 * Math.PI / sides;
    for (let i = 0; i <= sides; i++) {
      let angle = i * angleStep;
      let x = Math.cos(angle);
      let y = Math.sin(angle);
      glVertex3f(x, y, 0);
    }
  glEnd();

  glColor3f(1, 0.5, 0);

  let apex = [0, 0, 1.5];
  for (let i = 0; i < sides; i++) {
    glPushMatrix();
      glRotatef(i * angleStep * 180/Math.PI, 0, 0, 1);
      glBegin(GL_TRIANGLES);
        glVertex3f(1, 0, 0);
        let nextAngle = angleStep;
        glVertex3f(Math.cos(nextAngle), Math.sin(nextAngle), 0);
        // Wierzchołek piramidy
        glVertex3f(apex[0], apex[1], apex[2]);
      glEnd();
    glPopMatrix();
  }
}
```

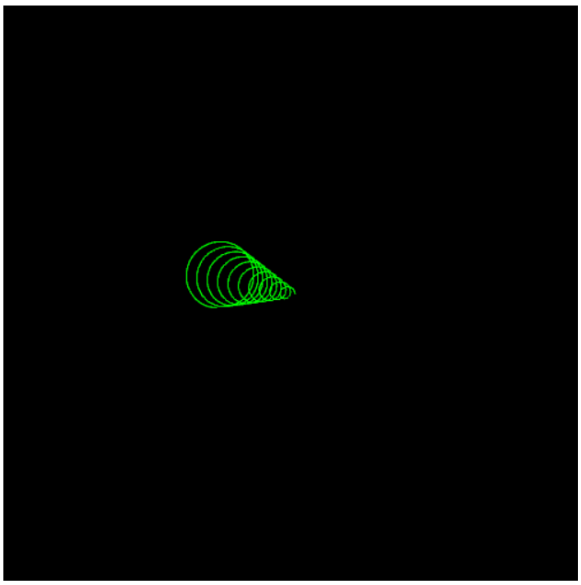
```
function drawTriangle() {
  glBegin(GL_TRIANGLES);
    glVertex3f(0, 0, 0);
    glVertex3f(1, 0, 0);
    glVertex3f(0.5, 1, 0);
  glEnd();
}
```

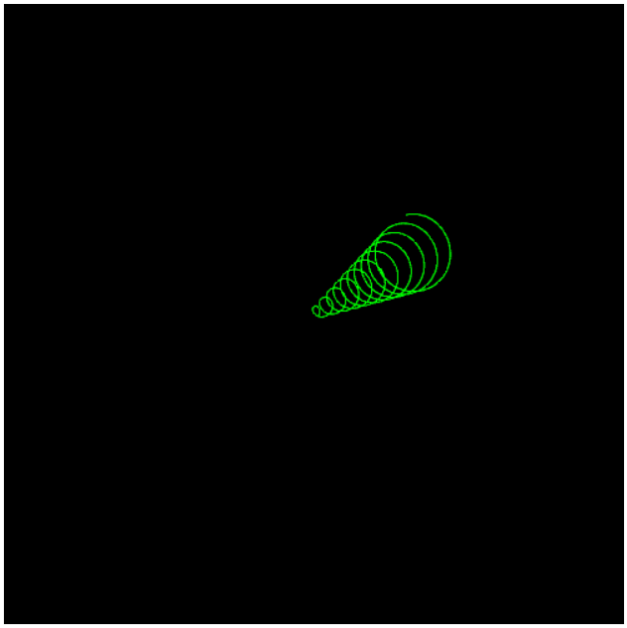
Link do Repozytorium:
https://github.com/bebrabimba/Grafika-Komputerowa/tree/main/Lab6

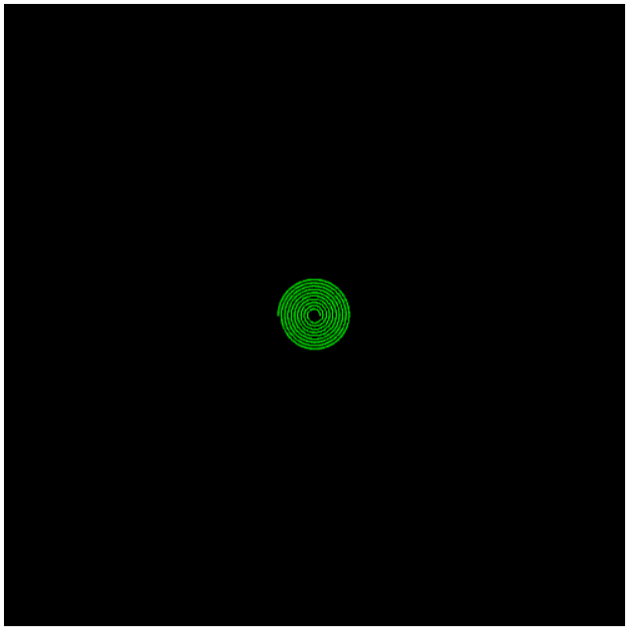
4. Wynik działania:

Zadanie 1:

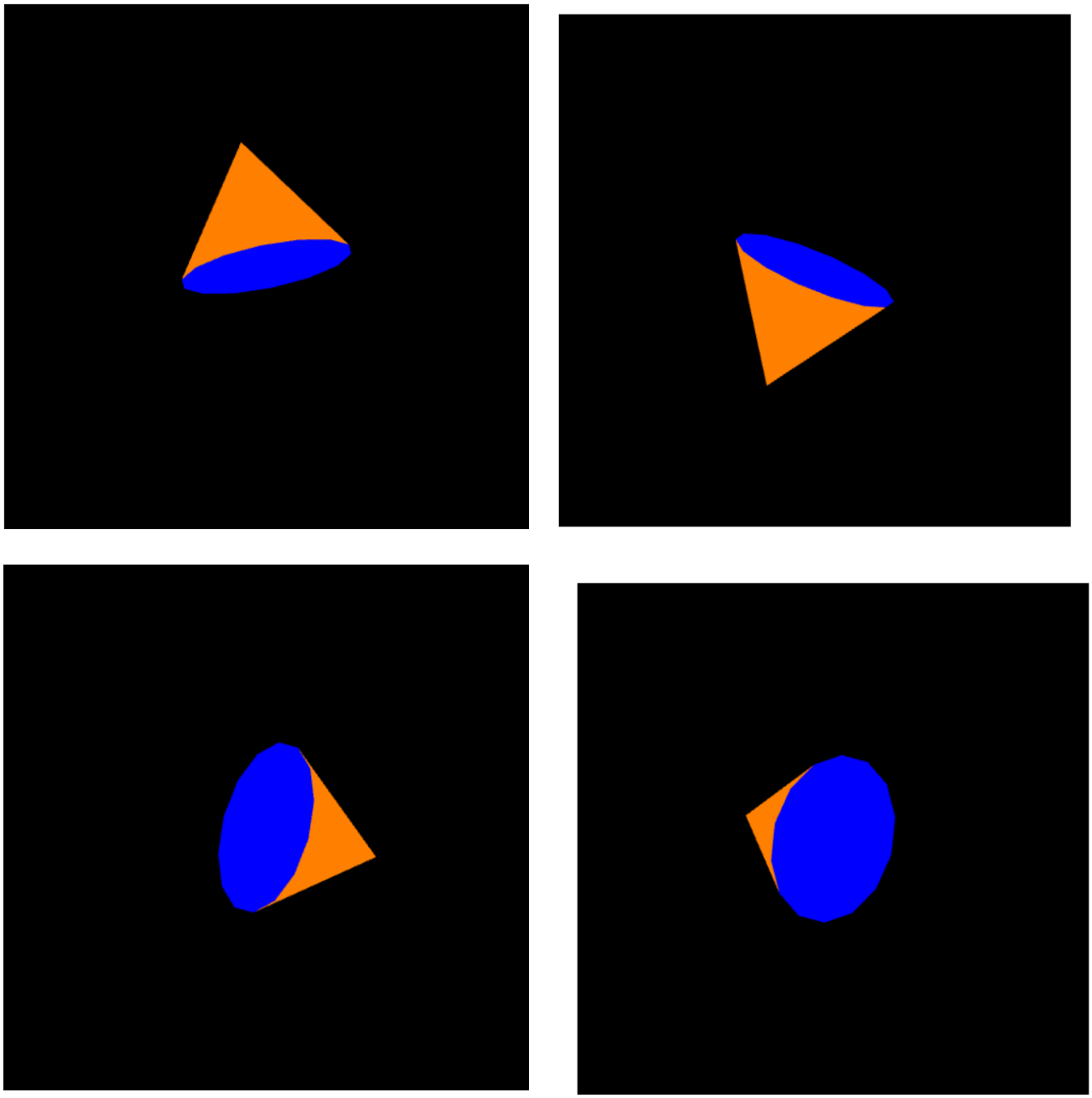








Zadanie 2:



Wnioski:

Przy użyciu odpowiednich równań matematycznych oraz funkcji można uzyskać ‘dynamiczną’ grafikę dzięki Javie oraz wykorzystaniu OpenGL.