МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 4

з дисципліни “Математичні та алгоритмічні основи комп’ютерної графіки”

Виконала
студентка III курсу
групи КП-82
Морщак Каріна
(*прізвище, ім’я, по батькові*)
варіант № 14

Зарахована
“____” “_____” 20___ р.
викладачем
Шкурат Оксаною Сергіївною
(*прізвище, ім’я, по батькові*)

Київ 2021

Варіант завдання

Тема: Побудова найпростіших тривимірних об'єктів за допомогою бібліотеки Java3D та їх анімація

Завдання: За допомогою засобів, що надає бібліотека Java3D, побудувати тривимірний об’єкт. Для цього скористатися основними примітивами, що буде доцільно використовувати згідно варіанту: сфера, конус, паралелепіпед, циліндр. Об'єкт має складатися з 5-15 примітивів. Задати матеріал кожного примітиву, в разі необхідності накласти текстуру. В сцені має бути мінімум одне джерело освітлення. Виконати анімацію сцени таким чином, щоб можна було розглянути об'єкт з усіх сторін. За бажанням можна виконати інтерактивні взаємодію з об'єктом за допомогою миші та клавіатури.

Варіант: Годинник

Лістинг коду програми

Clock.java

```
package main;

import javax.media.j3d.*;
import javax.vecmath.Color3f;
import javax.vecmath.Point3d;
import javax.vecmath.Vector3f;
import java.util.Date;

import com.sun.j3d.utils.image.TextureLoader;
import java.awt.*;



public class Clock {
	private TransformGroup objectTransformGroup;
	private Transform3D clockTransform3D = new Transform3D();
	private TransformGroup clockTransformGroupSeconds = new
TransformGroup();
	private TransformGroup clockTransformGroupMinutes = new
TransformGroup();
	private TransformGroup clockTransformGroupHours = new
TransformGroup();
	private float angle = 0;

	public BranchGroup createSceneGraph() {
		// створюємо групу об'єктів
		BranchGroup objRoot = new BranchGroup();

		// створюємо об'єкт, що будемо додавати до групи
		objectTransformGroup = new TransformGroup();

	objectTransformGroup.setCapability(TransformGroup.ALLOW_TRANSFO
RM_WRITE);
		buildObject();
		objRoot.addChild(objectTransformGroup);
```

```
            TextureLoader loader = new
TextureLoader("Lab4/src/honklhonk.jpg", "LUMINANCE", new
Container());
        var texture = loader.getImage();

        Background background = new Background(texture);
        background.setImageScaleMode(Background.SCALE_FIT_MAX);
        background.setCapability(Background.ALLOW_IMAGE_WRITE);
        BoundingSphere sphere = new BoundingSphere(new
Point3d(0,0,0), 100000);
        background.setApplicationBounds(sphere);
        objRoot.addChild(background);

            // налаштування освітлення
            BoundingSphere bounds = new BoundingSphere(new
Point3d(0.0, 0.0, 0.0),100.0);
            Color3f light1Color = new Color3f(1.0f, 1f, 1f);
            Vector3f light1Direction = new Vector3f(4.0f, -7.0f, -
12.0f);
            DirectionalLight light1 = new
DirectionalLight(light1Color, light1Direction);
            light1.setInfluencingBounds(bounds);
            objRoot.addChild(light1);

            // встановимо навколишнє освітлення
            Color3f ambientColor = new Color3f(1.0f, 1.0f, 1.0f);
            AmbientLight ambientLightNode = new
AmbientLight(ambientColor);
            ambientLightNode.setInfluencingBounds(bounds);
            objRoot.addChild(ambientLightNode);

            return objRoot;
      }

      private void buildObject() {
            double angle = 0;

                  Transform3D transform3D = new Transform3D();
```

```
                TransformGroup transformGroup = new TransformGroup();
                transform3D.rotX(Math.PI/2);
                transform3D.setTranslation(new Vector3f(0f, 0f,
0.01f));
                transformGroup.setTransform(transform3D);

                transformGroup.addChild(ClockElements.getBase());
                objectTransformGroup.addChild(transformGroup);

                Transform3D transform3D1 = new Transform3D();
                TransformGroup transformGroup1 = new
TransformGroup();
                transform3D1.rotX(Math.PI/2);
                transform3D1.setTranslation(new Vector3f(0f, 0f,
0.0f));
                transformGroup1.setTransform(transform3D1);

                transformGroup1.addChild(ClockElements.getBack());
                objectTransformGroup.addChild(transformGroup1);


     clockTransformGroupSeconds.setCapability(TransformGroup.ALLOW_T
RANSFORM_WRITE);

     clockTransformGroupMinutes.setCapability(TransformGroup.ALLOW_T
RANSFORM_WRITE);

     clockTransformGroupHours.setCapability(TransformGroup.ALLOW_TRA
NSFORM_WRITE);

     clockTransformGroupSeconds.addChild(ClockElements.getSecondsHan
d());

     clockTransformGroupMinutes.addChild(ClockElements.getMinutesHan
d());

     clockTransformGroupHours.addChild(ClockElements.getHoursHand())
;
```

```
      objectTransformGroup.addChild(clockTransformGroupHours);

      objectTransformGroup.addChild(clockTransformGroupMinutes);

      objectTransformGroup.addChild(clockTransformGroupSeconds);

                  updateClock();

            for(int i = 0; i < 12; ++i) {
                  Transform3D transform3D2 = new Transform3D();
                  transform3D2.rotZ(angle);
                  transform3D2.setTranslation(new
Vector3f((float)Math.cos(angle)*.4f, (float)Math.sin(angle)*.4f,
0.1f));
                  TransformGroup transformGroup2 = new
TransformGroup();
                  transformGroup2.setTransform(transform3D2);
                  transformGroup2.addChild(ClockElements.getDash());
                  angle += Math.PI/6;
                  objectTransformGroup.addChild(transformGroup2);
            }

      }

      private static void rotateHand(double angle, float handLength,
TransformGroup tg) {
            angle += Math.PI/2;
            Transform3D transform = new Transform3D();
            transform.rotZ(angle);
            transform.setTranslation(new
Vector3f((float)Math.cos(angle)*handLength,
                        (float)Math.sin(angle)*handLength, 0.1f));
            tg.setTransform(transform);
      }

      public void rotate() {
            clockTransform3D.rotY(angle);
```

```
            angle += 0.05;
            objectTransformGroup.setTransform(clockTransform3D);
      }

      public void updateClock() {
            Date date = new Date();
            date.setTime(System.currentTimeMillis());
            rotateHand(-Math.PI*2*date.getHours()/12.0, 0.2f,
clockTransformGroupHours);
            rotateHand(-Math.PI*2*date.getMinutes()/60.0, 0.25f,
clockTransformGroupMinutes);
            rotateHand(-Math.PI*2*date.getSeconds()/60.0, 0.25f,
clockTransformGroupSeconds);
      }
}
```

ClockElements.java

```
package main;

import com.sun.j3d.utils.geometry.Box;
import com.sun.j3d.utils.geometry.Cone;
import com.sun.j3d.utils.geometry.Cylinder;
import com.sun.j3d.utils.geometry.Primitive;

import javax.media.j3d.Appearance;
import javax.media.j3d.Material;
import javax.vecmath.Color3f;

public class ClockElements {
      public static Primitive getDash() {
            int primflags = Primitive.GENERATE_NORMALS +
Primitive.GENERATE_TEXTURE_COORDS;
            return new Box(0.1f, 0.01f, 0.01f, primflags,
getBlackAppearance());
      }

      //задня частина годинника
      public static Primitive getBack() {
```

```
            Appearance appearance = new Appearance();
            Color3f emissive = new Color3f(0f, 0f, 0f);
            Color3f ambient = new Color3f(0f, 0f, 0f);
            Color3f diffuse = new Color3f(.3f, .2f, .3f);
            Color3f specular = new Color3f(.7f, .8f, .9f);
            appearance.setMaterial(new Material(ambient, emissive,
diffuse, specular, 1f));
            int primflags = Primitive.GENERATE_NORMALS +
Primitive.GENERATE_TEXTURE_COORDS;
            return new Cylinder(0.6f, 0.2f, primflags, appearance);
      }

      //циферблат годинника
      public static Primitive getBase() {
            Appearance appearance = new Appearance();
            Color3f emissive = new Color3f(0f, 0f, 0f);
            Color3f ambient = new Color3f(0f, 0f, 0f);
            Color3f diffuse = new Color3f(.8f, .1f, .2f);
            Color3f specular = new Color3f(.8f, .1f, .2f);
            appearance.setMaterial(new Material(ambient, emissive,
diffuse, specular, 1f));
            int primflags = Primitive.GENERATE_NORMALS +
Primitive.GENERATE_TEXTURE_COORDS;
            return new Cylinder(0.5f, 0.19f, primflags, appearance);
      }

      //годинна стрілка
      public static Primitive getHoursHand() {
            int primflags = Primitive.GENERATE_NORMALS +
Primitive.GENERATE_TEXTURE_COORDS;
            return new Box(0.15f, 0.03f, 0.01f, primflags,
getBlackAppearance());
      }

      //хвилинна стрілка
      public static Primitive getMinutesHand() {
            int primflags = Primitive.GENERATE_NORMALS +
Primitive.GENERATE_TEXTURE_COORDS;
```

```
            return new Box(0.22f, 0.03f, 0.01f, primflags,
getBlackAppearance());
      }

      //секундна стрілка
      public static Primitive getSecondsHand() {
            int primflags = Primitive.GENERATE_NORMALS +
Primitive.GENERATE_TEXTURE_COORDS;
            return new Box(0.25f, 0.02f, 0.01f, primflags,
getBlackAppearance());
      }



      public static Appearance getBlackAppearance() {
            Appearance appearance = new Appearance();
            Color3f emissive = new Color3f(0.08f, 0.08f, 0.08f);
            Color3f ambient = new Color3f(0.08f, 0.08f, 0.08f);
            Color3f diffuse = new Color3f(0.2f, 0.2f, 0.2f);
            Color3f specular = new Color3f(0.1f, 0.1f, 0.1f);
            appearance.setMaterial(new Material(ambient, emissive,
diffuse, specular, 1.2f));
            return appearance;
      }
}
```

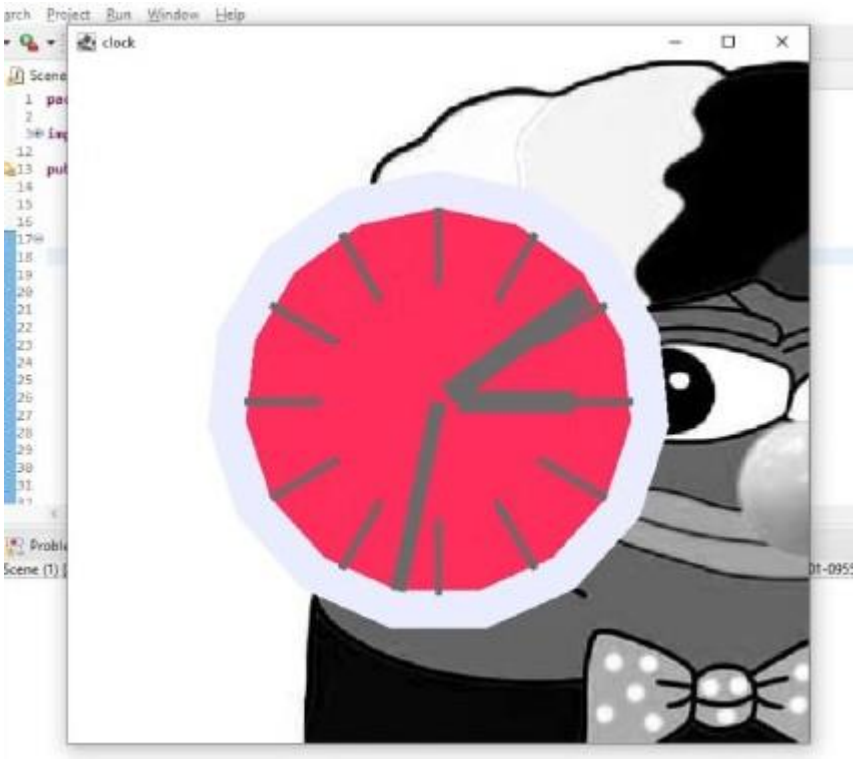
Scene.java

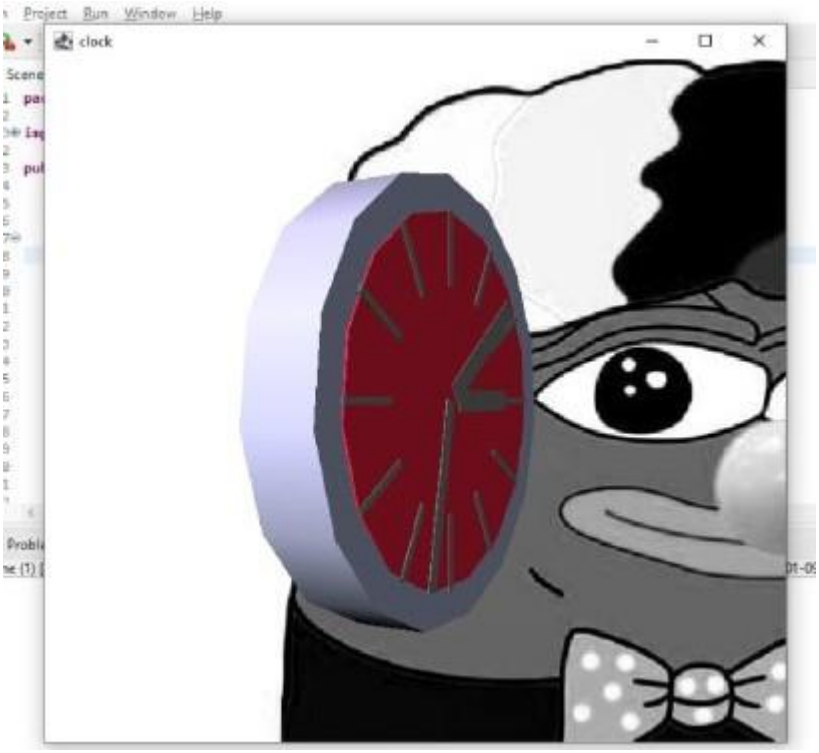
```
package main;

import com.sun.j3d.utils.universe.SimpleUniverse;

import javax.media.j3d.BranchGroup;
import javax.media.j3d.Canvas3D;
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
```

```
public class Scene extends JFrame implements ActionListener,
KeyListener {
	Clock clock;
	boolean rotateClock = false;

	public Scene() {
		super("clock");
		clock = new Clock();
		Canvas3D canvas3D = new
Canvas3D(SimpleUniverse.getPreferredConfiguration());
		add(canvas3D);
		canvas3D.addKeyListener(this);

		Timer timer = new Timer(75, this);
		timer.start();
		BranchGroup scene = clock.createSceneGraph();
		SimpleUniverse universe = new SimpleUniverse(canvas3D);

	universe.getViewingPlatform().setNominalViewingTransform();
		universe.addBranchGraph(scene);

		setSize(500, 500);
		setLocationRelativeTo(null);
		setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
		setVisible(true);
	}

	public static void main(String[] args) {
		new Scene();
	}

	@Override
	public void actionPerformed(ActionEvent e) {
		if(rotateClock) {
			clock.rotate();
		}
	}
```

```
	@Override
	public void keyTyped(KeyEvent keyEvent) {}

	@Override
	public void keyPressed(KeyEvent keyEvent) {
		if(keyEvent.getKeyCode() == KeyEvent.VK_ALT) {
			rotateClock = true;
		}
	}

	@Override
	public void keyReleased(KeyEvent keyEvent) {
		if(keyEvent.getKeyCode() == KeyEvent.VK_ALT) {
			rotateClock = false;
		}
	}
}
```

Результат




Project Run Window Help
clock