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”

Виконав
студент III курсу
групи КП-83
Палій Дмитро Володимирович
(*прізвище, ім’я, по батькові*)

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

варіант № 16

Київ 2021

Варіант завдання

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: 16. Жолудь

Лістинг коду програми

```
package sample;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import com.sun.j3d.utils.geometry.Cylinder;
import com.sun.j3d.utils.universe.SimpleUniverse;
import javax.media.j3d.*;
import javax.swing.Timer;
import javax.vecmath.*;
import javax.media.j3d.Appearance;
import javax.media.j3d.Material;
import javax.vecmath.Color3f;
import com.sun.j3d.utils.geometry.Cone;
import com.sun.j3d.utils.geometry.Primitive;
import com.sun.j3d.utils.geometry.Sphere;

public class Main implements ActionListener {
    private TransformGroup acornTG;
    private Transform3D treeTransform3D = new Transform3D();
    private Timer timer;
    private float angle = 0;
    public static void main(String[] args) {
        new Main();
    }
    public Main() {
        timer = new Timer(50, this);
        timer.start();
        BranchGroup scene = createSceneGraph();
        SimpleUniverse u = new SimpleUniverse();
        u.getViewingPlatform().setNominalViewingTransform();
        u.addBranchGraph(scene);
    }
    public BranchGroup createSceneGraph() {
        BranchGroup objRoot = new BranchGroup();

        acornTG = new TransformGroup();
        Transform3D zoom = new Transform3D();
        acornTG.setTransform(zoom);
        acornTG.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
```

```
        buildAcorn();
        objRoot.addChild(acornTG);

        BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0,
0.0),100.0);
        Color3f light1Color = new Color3f(1.0f, 0.5f, 0.4f);
        Vector3f light1Direction = new Vector3f(4.0f, -7.0f, -12.0f);
        DirectionalLight light1 = new DirectionalLight(light1Color,
                light1Direction);
        light1.setInfluencingBounds(bounds);
        objRoot.addChild(light1);

        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 buildAcorn() {

        int primflags = Primitive.GENERATE_NORMALS +
Primitive.GENERATE_TEXTURE_COORDS;
        Appearance branchAppearance = new Appearance();
        Color3f branchEmissive = new Color3f(0.0f, 0.05f, 0.0f);
        Color3f branchAmbient = new Color3f(0.02f, 0.1f, 0.0f);
        Color3f branchDiffuse = new Color3f(0.02f, 0.1f, 0.0f);
        Color3f branchSpecular = new Color3f(0.0f, 0.2f, 0.0f);
        branchAppearance.setMaterial(new Material(branchEmissive,
branchAmbient, branchDiffuse, branchSpecular, 5.0f));

        Appearance bodyAppearance = new Appearance();
        Color3f bodyEmissive = new Color3f(0.0f, 0.1f, 0.0f);
        Color3f bodyAmbient = new Color3f(0.0f, 0.2f, 0.0f);
        Color3f bodyDiffuse = new Color3f(0.0f, 0.2f, 0.0f);
        Color3f bodySpecular = new Color3f(0.0f, 0.3f, 0.0f);
        bodyAppearance.setMaterial(new Material(bodyEmissive, bodyAmbient,
bodyDiffuse, bodySpecular, 70.0f));

        Transform3D rotation = new Transform3D();
        rotation.rotX(Math.PI);

        // branch
        TransformGroup tgBranch = new TransformGroup();
        Transform3D transformBranch = new Transform3D();
        Cone coneBranch = new Cone(0.1f, 0.3f, primflags, branchAppearance);
        Vector3f vectorBranch = new Vector3f(.0f, .5f, .0f);
        transformBranch.setTranslation(vectorBranch);
        transformBranch.mul(rotation);
        tgBranch.setTransform(transformBranch);
        tgBranch.addChild(coneBranch);
        acornTG.addChild(tgBranch);

        // cap
        TransformGroup tgCap = new TransformGroup();
        Transform3D transformCap = new Transform3D();
        Sphere cap = new Sphere(.5f, primflags, branchAppearance);
        Vector3f vectorCap = new Vector3f(.0f, .0f, .0f);
        transformCap.setTranslation(vectorCap);
        tgCap.setTransform(transformCap);
        tgCap.addChild(cap);
        acornTG.addChild(tgCap);

        // body
        TransformGroup tgBody = new TransformGroup();
```

```
        Transform3D transformBody = new Transform3D();
        Cylinder body = new Cylinder(.45f, 0.7f, primflags, bodyAppearance);
        Vector3f vectorBody = new Vector3f(.0f, -0.35f, .0f);
        transformBody.setTranslation(vectorBody);
        tgBody.setTransform(transformBody);
        tgBody.addChild(body);
        acornTG.addChild(tgBody);

        // body bottom
        TransformGroup tgBodyBottom = new TransformGroup();
        Transform3D transformBodyBottom = new Transform3D();
        Sphere bodyBottom = new Sphere(.45f, primflags, bodyAppearance);
        Vector3f vectorBodyBottom = new Vector3f(.0f, -0.7f, .0f);
        transformBodyBottom.setTranslation(vectorBodyBottom);
        tgBodyBottom.setTransform(transformBodyBottom);
        tgBodyBottom.addChild(bodyBottom);
        acornTG.addChild(tgBodyBottom);

        // spike
        TransformGroup tgSpike = new TransformGroup();
        Transform3D transformSpike = new Transform3D();
        Cone spike = new Cone(0.1f, 0.1f, primflags, bodyAppearance);
        Vector3f vectorSpike = new Vector3f(.0f, -1.15f, .0f);
        transformSpike.setTranslation(vectorSpike);
        transformSpike.mul(rotation);
        tgSpike.setTransform(transformSpike);
        tgSpike.addChild(spike);
        acornTG.addChild(tgSpike);
    }
    @Override
    public void actionPerformed(ActionEvent e) {
        treeTransform3D.rotX(angle);
        acornTG.setTransform(treeTransform3D);
        angle += 0.05;
    }
}
```

Результат