

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 №** **5**

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

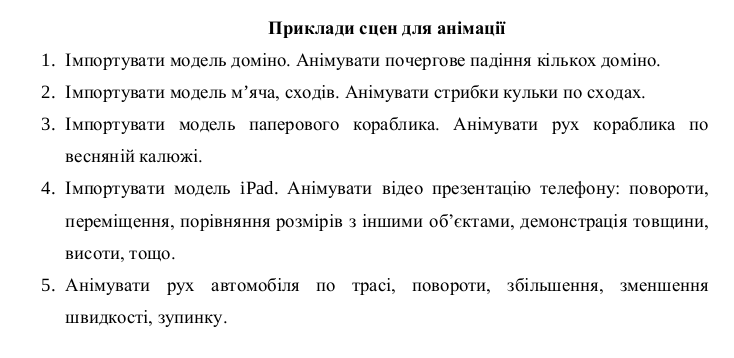
|  |  |  |  |
| --- | --- | --- | --- |
| Виконав  студент III курсу  групи КП-81  Черняєв Іван  (*прізвище, ім’я, по батькові*)  варіант № 19 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2021

**Варіант завдання**

**Завдання**:

**Варіант 19**



**Лістинг коду програми**

package com.company;import com.sun.j3d.loaders.Scene;import com.sun.j3d.loaders.objectfile.ObjectFile;import com.sun.j3d.utils.image.TextureLoader;import com.sun.j3d.utils.universe.SimpleUniverse;import com.sun.j3d.utils.universe.ViewingPlatform;import javax.media.j3d.\*;import javax.swing.\*;import javax.vecmath.Color3f;import javax.vecmath.Point3d;import javax.vecmath.Vector3d;import javax.vecmath.Vector3f;import java.awt.\*;import java.io.FileReader;import java.io.IOException;import java.util.Map;class IPad extends JFrame { private Canvas3D canvas; private SimpleUniverse universe; private BranchGroup root; private TransformGroup lego; private Map<String, Shape3D> shapeMap; IPad() throws IOException { configureWindow(); configureCanvas(); configureUniverse(); root = new BranchGroup(); root.setCapability(BranchGroup.*ALLOW\_CHILDREN\_EXTEND*); addImageBackground("/home/ivan/workspace/Lab5/src/resources/room.jpg"); addLightToUniverse(); changeViewAngle(); lego = getIPadGroup(); TransformGroup room = new TransformGroup(); room.addChild(lego); root.addChild(room); IPadAnimation lego = new IPadAnimation(this); canvas.addKeyListener(lego); root.compile(); universe.addBranchGraph(root); } private void configureWindow() { setTitle("IPad Animation"); setSize(1000, 600); setResizable(true); setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*); } private void configureCanvas() { canvas = new Canvas3D(SimpleUniverse.*getPreferredConfiguration*()); canvas.setDoubleBufferEnable(true); canvas.setFocusable(true); add(canvas, BorderLayout.*CENTER*); } private void configureUniverse() { universe = new SimpleUniverse(canvas); universe.getViewingPlatform().setNominalViewingTransform(); } private void addImageBackground(String imagePath) { TextureLoader t = new TextureLoader(imagePath, canvas); Background background = new Background(t.getImage()); background.setImageScaleMode(Background.*SCALE\_FIT\_ALL*); BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), 100.0); background.setApplicationBounds(bounds); root.addChild(background); } private void addLightToUniverse() { BoundingSphere bounds = new BoundingSphere(); bounds.setRadius(1000); DirectionalLight directionalLight = new DirectionalLight( new Color3f(new Color(255, 255, 255)), new Vector3f(0, -0.5f, -0.5f)); directionalLight.setInfluencingBounds(bounds); AmbientLight ambientLight = new AmbientLight( new Color3f(new Color(255, 255, 245))); ambientLight.setInfluencingBounds(bounds); root.addChild(directionalLight); root.addChild(ambientLight); } private TransformGroup getIPadGroup() throws IOException { Transform3D scale = new Transform3D(); scale.setScale(new Vector3d(0.8, 0.8, 0.8)); Transform3D yRotation = new Transform3D(); yRotation.rotY(Math.*PI*); Transform3D zRotation = new Transform3D(); zRotation.rotZ(.06); Transform3D xRotation = new Transform3D(); xRotation.rotX(-.5); zRotation.mul(xRotation); yRotation.mul(zRotation); scale.mul(yRotation); TransformGroup group = getModelGroup("src/resources/ipad\_pro.obj"); group.setTransform(scale); return group; } private TransformGroup getModelGroup(String path) throws IOException { Scene scene = *getSceneFromFile*(path); shapeMap = scene.getNamedObjects(); printModelElementsList(shapeMap); TransformGroup group = new TransformGroup(); for (String shapeName : shapeMap.keySet()) { Shape3D shape = shapeMap.get(shapeName); scene.getSceneGroup().removeChild(shape); group.addChild(shape); } group.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*); return group; } private void printModelElementsList(Map<String, Shape3D> shapeMap) { for (String name : shapeMap.keySet()) { System.*out*.printf("Name: %s\n", name); } } /\*private Appearance getAppearance(Color materialColor) { Appearance appearance = new Appearance(); appearance.setMaterial(getMaterial(materialColor)); return appearance; } private Material getMaterial(Color defaultColor) { Material material = new Material(); material.setEmissiveColor(new Color3f(Color.BLACK)); material.setAmbientColor(new Color3f(defaultColor)); material.setDiffuseColor(new Color3f(defaultColor)); material.setSpecularColor(new Color3f(defaultColor)); material.setShininess(80); material.setLightingEnable(true); return material; }\*/ private void changeViewAngle() { ViewingPlatform vp = universe.getViewingPlatform(); TransformGroup vpGroup = vp.getMultiTransformGroup().getTransformGroup(0); Transform3D vpTranslation = new Transform3D(); vpTranslation.setTranslation(new Vector3f(0, 0, 6)); vpGroup.setTransform(vpTranslation); } private static Scene getSceneFromFile(String location) throws IOException { ObjectFile file = new ObjectFile(ObjectFile.*RESIZE*); file.setFlags(ObjectFile.*RESIZE* | ObjectFile.*TRIANGULATE* | ObjectFile.*STRIPIFY*); return file.load(new FileReader(location)); } TransformGroup getLegoTransformGroup() { return lego; }}

**Результат**

