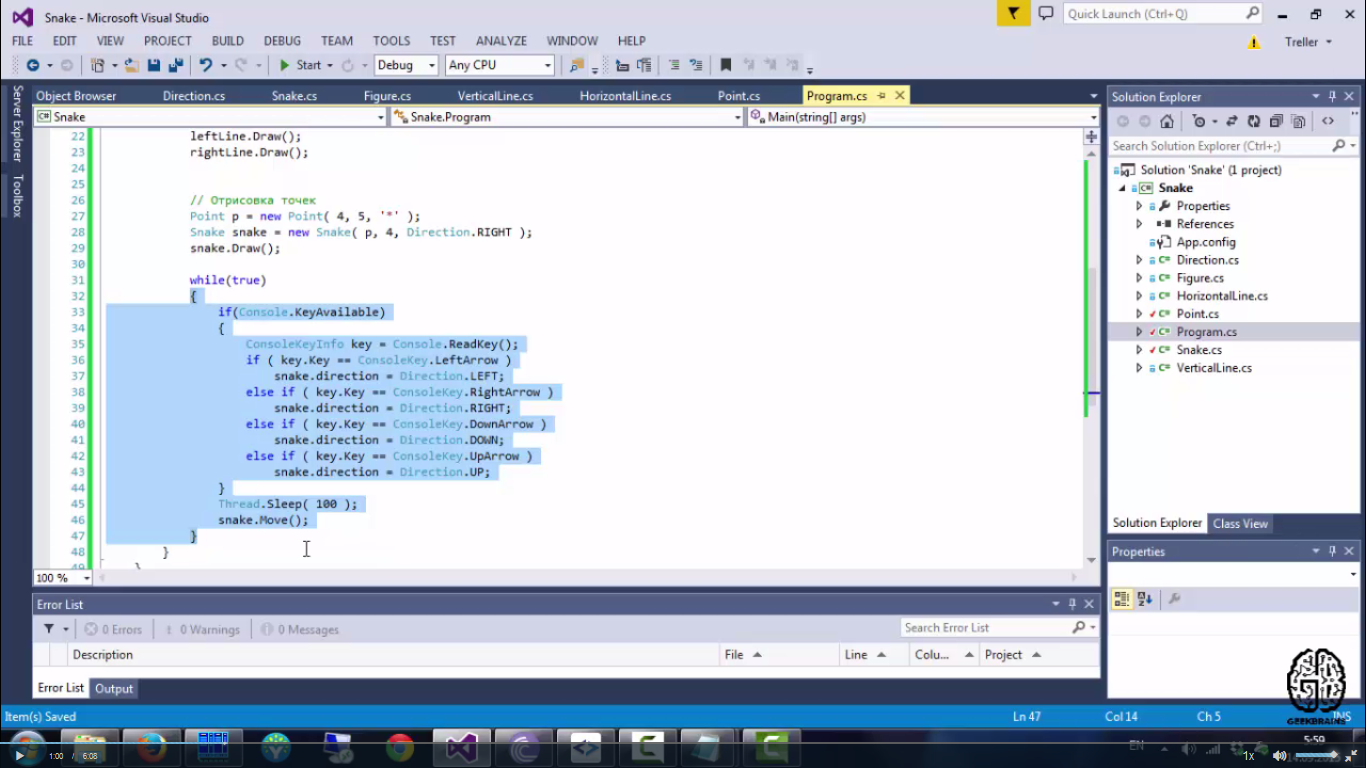
public class MyGdxGame extends ApplicationAdapter {  
 //float time = 0.0f;  
 BitmapFont bmf;  
 private final String[] colorMushroom = {"Гриб.png", "ГрибЗел.png", "ГрибСин.png"};  
 SpriteBatch batch;  
 private final int MUSHROOM\_COUNT = 1;  
 Mushrooms[] mushrooms = new Mushrooms[MUSHROOM\_COUNT];  
 Random random = new Random();  
  
 @Override  
 public void create () {  
 batch = new SpriteBatch();  
 bmf = new BitmapFont(Gdx.*files*.internal("myText.fnt"), Gdx.*files*.internal("myText.png"), false);  
 initMushroom();  
 }  
  
 private void initMushroom() {  
 for (int i = 0; i < MUSHROOM\_COUNT; i++) {  
 int randColor = (int) (Math.*random*() \* colorMushroom.length);  
 mushrooms[i] = new Mushrooms(new Vector2(random.nextInt(500), random.nextInt(500)),  
 new Vector2(3.0f \* (random.nextFloat() - 0.5f), 3.0f \* (random.nextFloat() - 0.5f)),  
 colorMushroom[randColor]); // 3.0f \* (random.nextFloat() - 0.5f) от -1.5 до 1.5  
 }  
 }  
  
 @Override  
 public void render () {  
 update();  
 Gdx.*gl*.glClearColor(1, 0, 1, 1);  
 Gdx.*gl*.glClear(GL20.*GL\_COLOR\_BUFFER\_BIT*);  
 batch.begin();  
 //time += 2.5f;  
 //batch.draw(new Texture("Гриб.png"), InputHandler.getMousePosition().x - 53, //картинка, расположение по Х,  
 // InputHandler.getMousePosition().y - 53, // расп. по Y  
 // 53, 53, // якорь Х, якорь Y,  
 // 106, 106,  
 // 1.0f, 1.0f, time,  
 // 0, 0,  
 // 106, 106, false, false);  
 drawMushroom();  
 bmf.draw(batch, "testText", InputHandler.*getMousePosition*().x, InputHandler.*getMousePosition*().y);  
 batch.end();  
 }  
  
 public void update(){ //метод для обновления игровой логики  
 for (int i = 0; i < MUSHROOM\_COUNT; i++) {  
 mushrooms[i].update();  
 }  
 }  
  
 public void drawMushroom(){  
 for (int i = 0; i < MUSHROOM\_COUNT; i++) {  
 mushrooms[i].render(batch);  
 }  
 }  
}

public class Mushrooms {  
 private Vector2 position; //координаты  
 private Vector2 velocity; //скорость  
  
 public Mushrooms(Vector2 position, Vector2 velocity, String color) {  
 setMushroomTexture(new Texture(color));  
 this.position = position;  
 this.velocity = velocity;  
 }  
  
 private static Texture *mushroomTexture*;  
  
 public void setMushroomTexture(Texture mushroomTexture) {  
 Mushrooms.*mushroomTexture* = mushroomTexture;  
 }  
  
 public void update(){  
 position.add(velocity);  
 // velocity.scl(1.009f); // ускорение  
 transferPictures();  
 }  
  
 void transferPictures(){  
 if (position.x < -*mushroomTexture*.getWidth()) position.x = Gdx.*graphics*.getWidth();  
 if (position.y < -*mushroomTexture*.getHeight()) position.y = Gdx.*graphics*.getHeight();  
 if (position.x > Gdx.*graphics*.getWidth()) position.x = -*mushroomTexture*.getWidth();  
 if (position.y > Gdx.*graphics*.getHeight()) position.y = -*mushroomTexture*.getHeight();  
  
 if(InputHandler.*inPressed*()){  
 if (position.cpy().sub(InputHandler.*getMousePosition*()).len() < 250)  
 velocity = position.cpy().sub(InputHandler.*getMousePosition*()).nor().scl(-1.0f);  
 // скопировали положение гриба.вычли координаты мыши.нормировали.в противоположную сторону  
 }  
 }  
  
 public void render(SpriteBatch batch){  
 batch.draw(*mushroomTexture*, position.x, position.y);  
 }  
}

public class InputHandler {  
 public static boolean inClicked(){  
 return Gdx.*input*.justTouched();  
 }  
  
 public static boolean inPressed(){  
 return Gdx.*input*.isTouched();  
 }  
  
 public static Vector2 getMousePosition(){  
 return new Vector2(Gdx.*input*.getX(),Gdx.*graphics*.getHeight() - Gdx.*input*.getY());  
 }  
}

public class DesktopLauncher {  
 public static void main (String[] arg) {  
 LwjglApplicationConfiguration config = new LwjglApplicationConfiguration();  
 new LwjglApplication(new MyGdxGame(), config);  
 config.width = 500;  
 config.height = 500;  
 }  
}



//10.40.76.3/общая\прикладное программирование\asoiu/user