



# **Pemrograman Berbasis Objek**

**INF2143 & INF2153**

## **LAPORAN UAS**

Oleh :

*Muhammad Ansar : NIM*

*Rahman : NIM*

*Reza Fahlevy : NIM*

Teknik Informatika  
Fakultas Sains & Teknologi  
Universitas Muhammadiyah Kalimantan Timur

Samarinda, 2023

# Laporan UAS

## Permainan Ikan Greenfoot

Berikut adalah proyek akhir semester yaitu membuat game menggunakan platform Greenfoot dengan menerapkan beberapa konsep PBO/OOP yaitu:

- Inheritance
- Polimorfisme
- Enkapsulasi
- Overriding

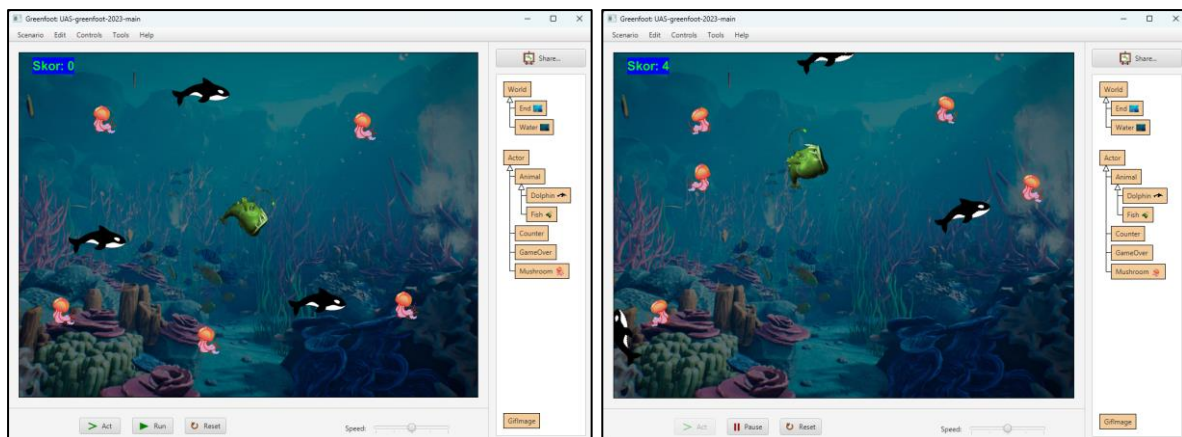
Tujuan permainan :

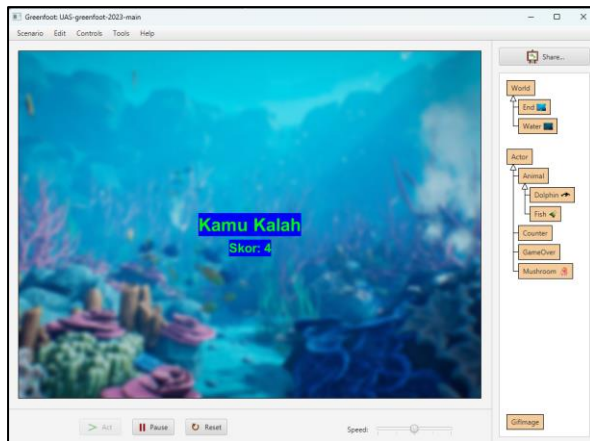
- Mencari skor sebanyak-banyaknya dengan cara memakan ubur-ubur
- Setiap skor bertambah maka kecepatan ikan akan semakin cepat
- Jika menabrak paus maka permainan akan berakhir

Cara Bermain :

- Untuk memulai permainan tekan Run pada greenfoot
- Untuk mengendalikan ikan gunakan arrow key pada keyboard
- Paus akan bergerak secara acak dan ubur-ubur akan spawn secara acak setelah dimakan
- Jika menabrak paus maka permainan akan berakhir. Untuk memulai ulang permainan klik kiri pada layar permainan menggunakan mouse

Beberapa screenshot dari permainan :





Screenshot Code Greenfoot:

Kelas Water :

```
import greenfoot.*;

public class Water extends World
{
    Counter counter = new Counter();
    public static int lives = 1;
    public static int score = 0;

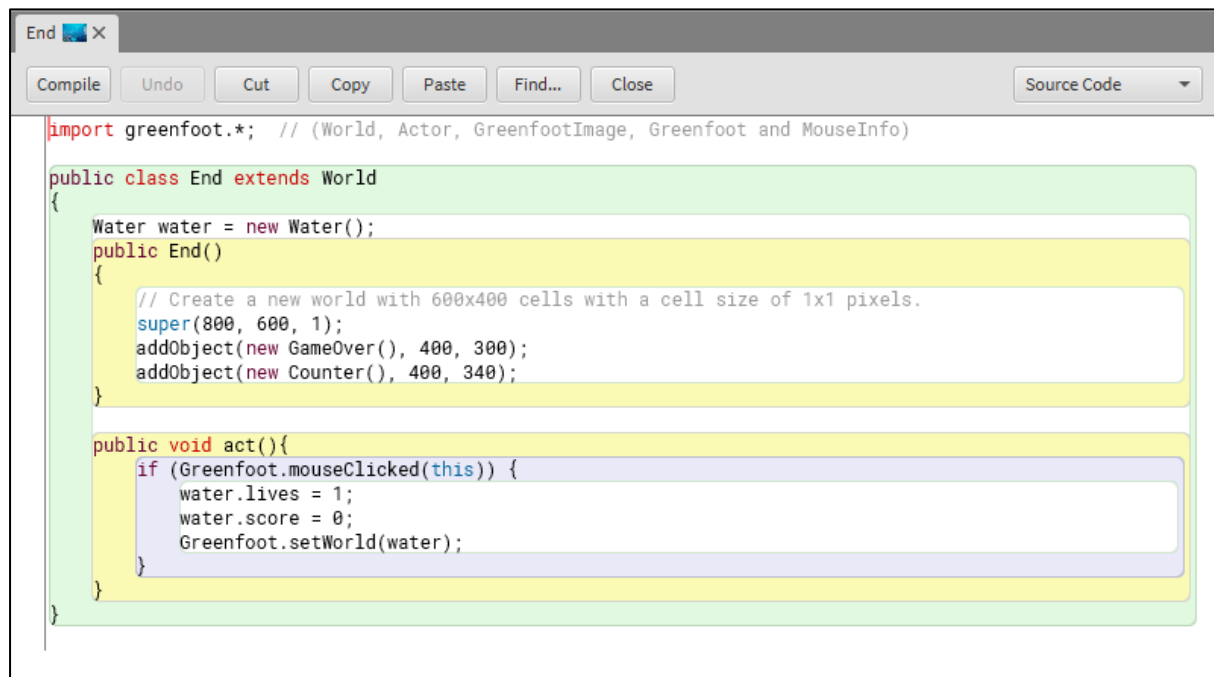
    public Water()
    {
        super(800, 600, 1);
        prepare();
    }

    public void act(){
        if (lives < 1) {
            Greenfoot.setWorld(new End());
        }
    }

    public void prepare()
    {
        Fish fish = new Fish();
        Dolphin dolphin = new Dolphin();
        Dolphin dolphin2 = new Dolphin();
        Dolphin dolphin3 = new Dolphin();
        Mushroom mushroom = new Mushroom();
        Mushroom mushroom2 = new Mushroom();
        Mushroom mushroom3 = new Mushroom();
        Mushroom mushroom4 = new Mushroom();
        Mushroom mushroom5 = new Mushroom();
        addObject(fish, 396, 280);
        addObject(dolphin, 317, 68);
        addObject(dolphin2, 136, 325);
        addObject(dolphin3, 520, 434);
        addObject(mushroom, 603, 129);
        addObject(mushroom2, 675, 441);
        addObject(mushroom3, 77, 451);
        addObject(mushroom4, 148, 117);
        addObject(mushroom5, 330, 502);

        addObject(counter, 58, 41);
        counter.setLocation(59, 22);
    }
}
```

Kelas End :



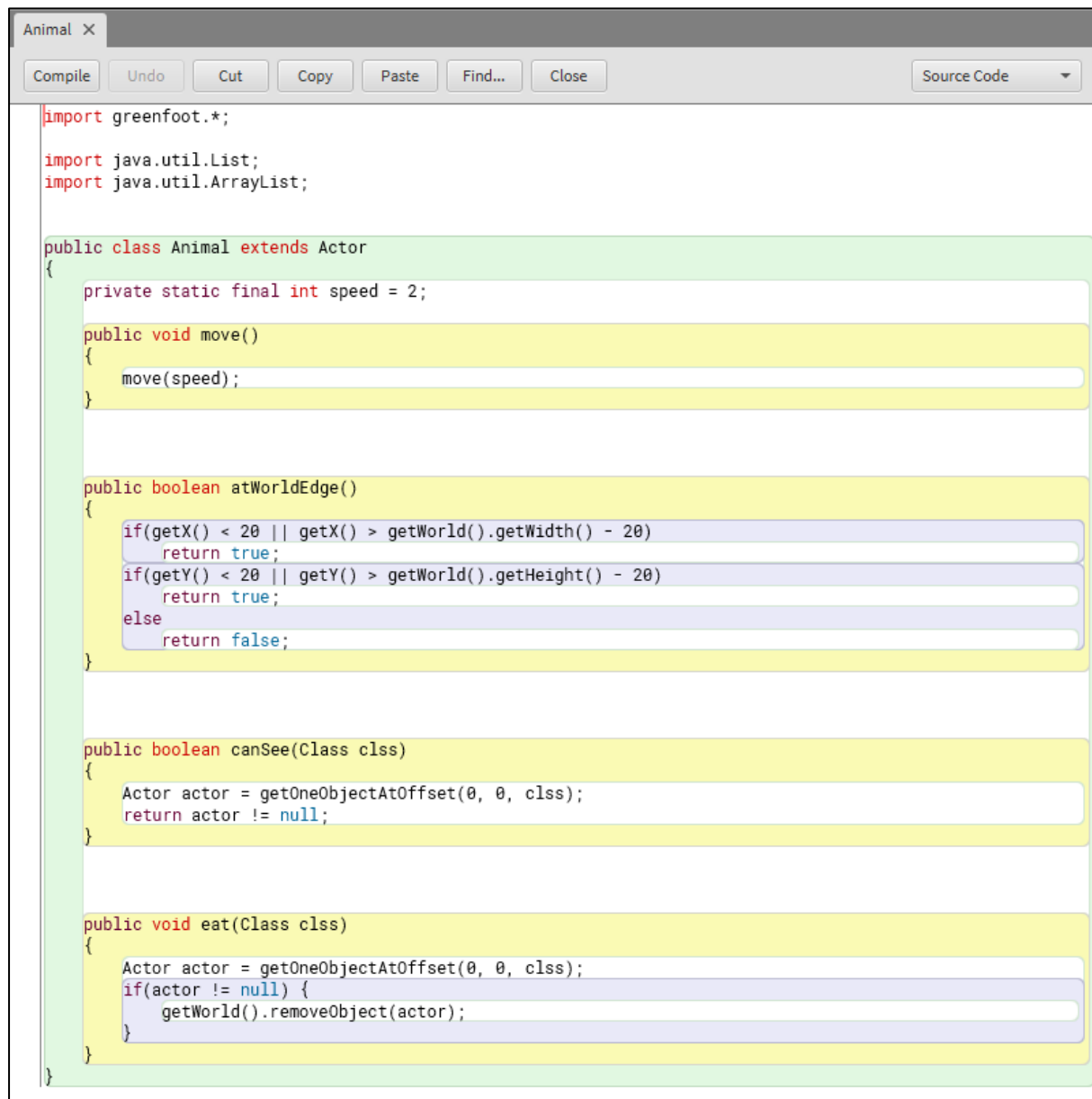
The screenshot shows a Java IDE window titled 'End'. The window has a menu bar with 'Compile', 'Undo', 'Cut', 'Copy', 'Paste', 'Find...', and 'Close'. There is also a 'Source Code' dropdown menu. The main area displays the following Java code:

```
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class End extends World
{
    Water water = new Water();
    public End()
    {
        // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
        super(800, 600, 1);
        addObject(new GameOver(), 400, 300);
        addObject(new Counter(), 400, 340);
    }

    public void act(){
        if (Greenfoot.mouseClicked(this)) {
            water.lives = 1;
            water.score = 0;
            Greenfoot.setWorld(water);
        }
    }
}
```

## Kelas Animal :



```
import greenfoot.*;

import java.util.List;
import java.util.ArrayList;

public class Animal extends Actor
{
    private static final int speed = 2;

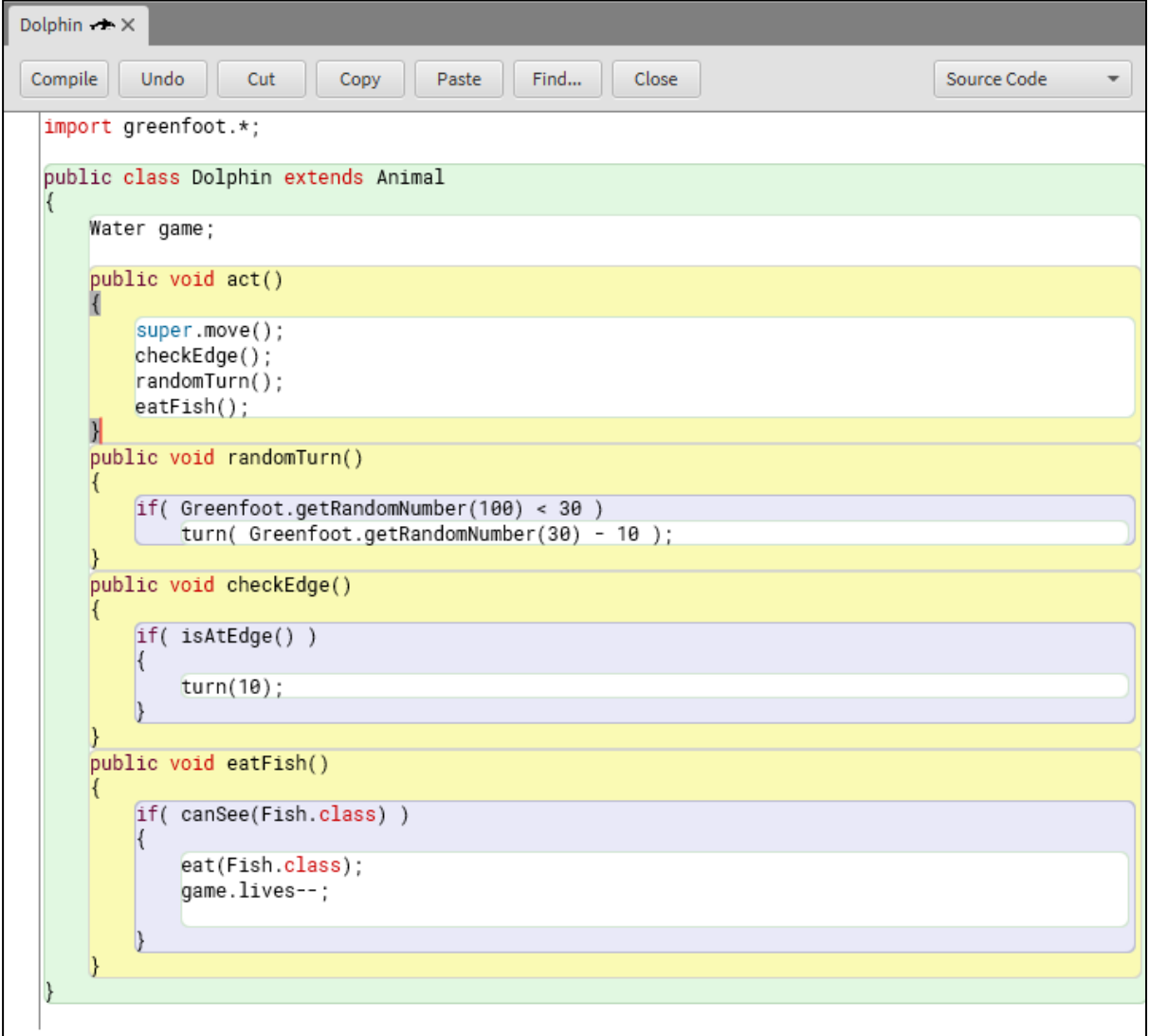
    public void move()
    {
        move(speed);
    }

    public boolean atWorldEdge()
    {
        if(getX() < 20 || getX() > getWorld().getWidth() - 20)
            return true;
        if(getY() < 20 || getY() > getWorld().getHeight() - 20)
            return true;
        else
            return false;
    }

    public boolean canSee(Class clss)
    {
        Actor actor = getOneObjectAtOffset(0, 0, clss);
        return actor != null;
    }

    public void eat(Class clss)
    {
        Actor actor = getOneObjectAtOffset(0, 0, clss);
        if(actor != null) {
            getWorld().removeObject(actor);
        }
    }
}
```

- Subclass dari Animal, Dolphin :



```
import greenfoot.*;

public class Dolphin extends Animal
{
    Water game;

    public void act()
    {
        super.move();
        checkEdge();
        randomTurn();
        eatFish();
    }

    public void randomTurn()
    {
        if( Greenfoot.getRandomNumber(100) < 30 )
            turn( Greenfoot.getRandomNumber(30) - 10 );
    }

    public void checkEdge()
    {
        if( isAtEdge() )
        {
            turn(10);
        }
    }

    public void eatFish()
    {
        if( canSee(Fish.class) )
        {
            eat(Fish.class);
            game.lives--;
        }
    }
}
```

- Subclass dari Animal, Fish :

```

import greenfoot.*;

public class Fish extends Animal
{
    GifImage gifImage = new GifImage("fish.gif");
    Water game;
    int noOfShroomEaten = 0;
    public void act()
    {
        super.move();
        move();
        checkKeys();
        eatShroom();
        gifimage();
    }
    public void move()
    {
        move(3+noOfShroomEaten);
        if(isAtEdge())
        {
            if (getX() == 0 )
                setLocation(799, getY());
            else if (getX() == 799 )
                setLocation(0, getY());
            else if (getY() == 0 )
                setLocation(getX(), 599);
            else if (getY() == 599 )
                setLocation(getX(),0);
        }
    }
    public void checkKeys()
    {
        if( Greenfoot.isKeyDown("left") )
            turn(-5);
        if( Greenfoot.isKeyDown("right") )
            turn(5);
    }
    public void eatShroom()
    {
        if( canSee(Mushroom.class) )
        {
            eat( Mushroom.class);
            noOfShroomEaten++;
            updateScore();
            createShroom();
        }
        if ( noOfShroomEaten == 100 )
        {
            World w = getWorld();
            w.addObject(new GameOver(), w.getWidth() / 2, w.getHeight() / 2);
            Greenfoot.stop();
        }
    }
    public void createShroom()

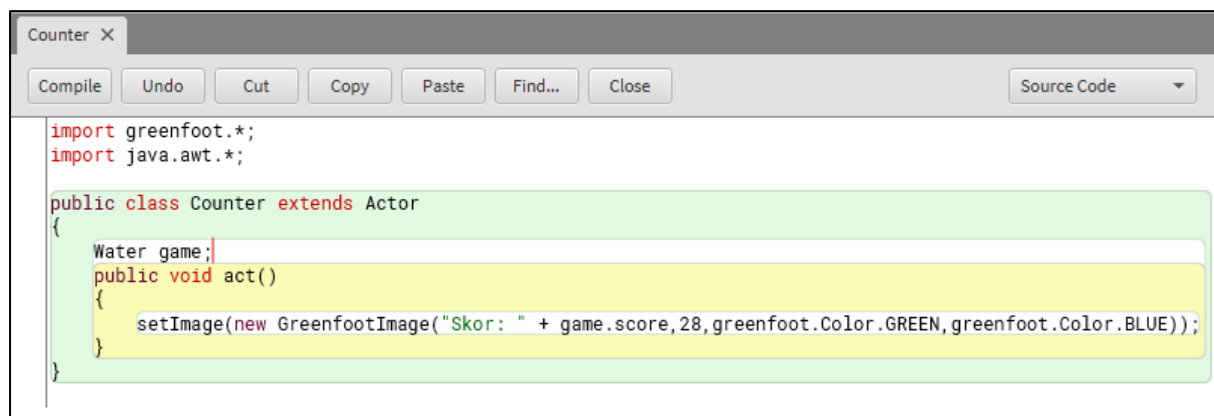
```

```
public void createShroom()
{
    Mushroom m = new Mushroom();
    World w;
    w = getWorld();
    w.addObject(m, Greenfoot.getRandomNumber(800), Greenfoot.getRandomNumber(600));
}

public void updateScore()
{
    game.score++;
}

public void gifimage()
{
    setImage(gifImage.getCurrentImage());
}
}
```

Kelas Counter :

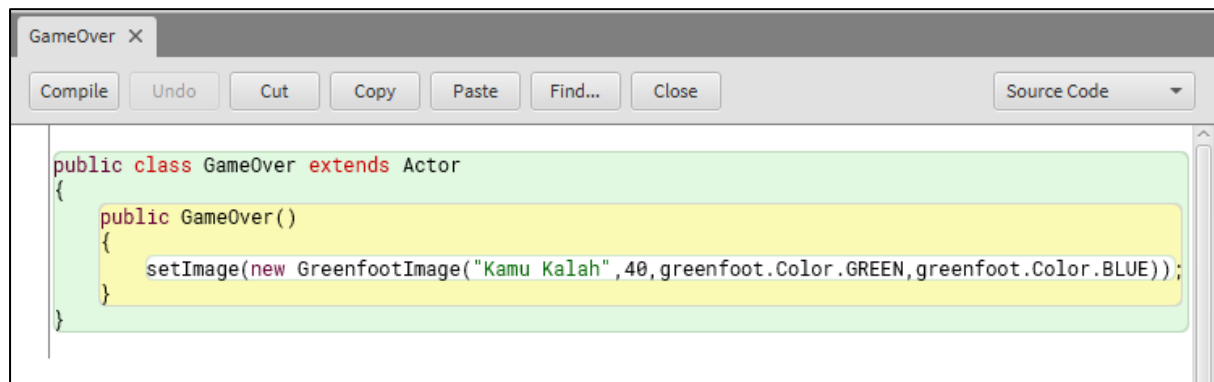


The screenshot shows a Java IDE window titled "Counter X". The code defines a class "Counter" that extends "Actor". It includes imports for "greenfoot.\*" and "java.awt.\*". Inside the class, there is a variable "Water game;" and a method "act()" that calls "setImage(new GreenfootImage('Skor: ' + game.score, 28, greenfoot.Color.GREEN, greenfoot.Color.BLUE));".

```
import greenfoot.*;
import java.awt.*;

public class Counter extends Actor
{
    Water game;
    public void act()
    {
        setImage(new GreenfootImage("Skor: " + game.score, 28, greenfoot.Color.GREEN, greenfoot.Color.BLUE));
    }
}
```

Kelas GameOver :

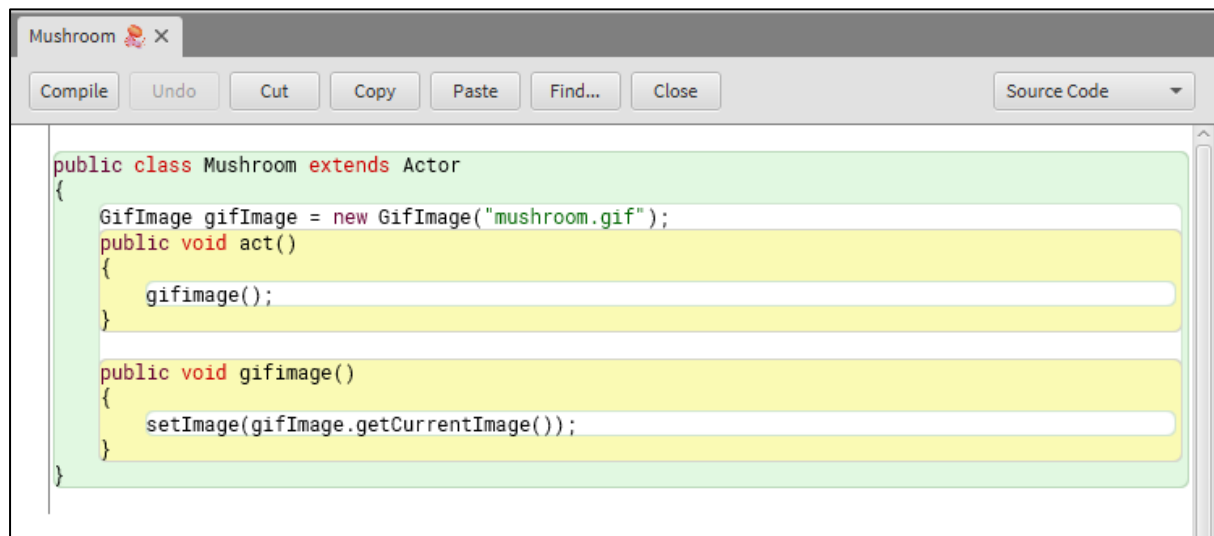


The screenshot shows a Java IDE window titled "GameOver X". The code defines a class "GameOver" that extends "Actor". It includes a constructor "public GameOver()" that calls "setImage(new GreenfootImage('Kamu Kalah', 40, greenfoot.Color.GREEN, greenfoot.Color.BLUE));".

```
public class GameOver extends Actor
{
    public GameOver()
    {
        setImage(new GreenfootImage("Kamu Kalah", 40, greenfoot.Color.GREEN, greenfoot.Color.BLUE));
    }
}
```



### Kelas Mushroom :



### Penjelasan Kelas :

- Water : Kelas Water adalah kelas subclass dari World yang berfungsi sebagai dunia dari permainan.
- End : Kelas End adalah subclass dari World yang difungsikan untuk memanggil kelas GameOver dan mereset semua variable kembali menjadi seperti semula.
- Animal : Kelas Animal merupakan kelas yang diimport dari greenfoot untuk menjalankan method seperti bergerak, memakan, dan dimakan.
- Dolphin : Dolphin merupakan subclass dari Animal yang berfungsi sebagai musuh dari permainan.
- Fish : Fish merupakan subclass dari animal yang berfungsi sebagai karakter yang dimainkan oleh player.
- Mushroom : Mushroom merupakan kelas dari ubur-ubur atau target dari player.
- Counter : Counter merupakan kelas yang berisi objek berupa papan skor.
- GameOver : Gameover merupakan kelas yang berisi objek berupa peringatan bahwa game telah berakhir.
- GifImage : GifImage merupakan kelas yang diimport dari greenfoot untuk memberikan animasi berupa gif pada beberapa actor agar terlihat lebih menarik.

### Penjelasan konsep yang digunakan :

#### Inheritance atau Pewarisan :

Inheritance yang digunakan pada game ini berada pada Kelas Animal dan subclassnya. Kelas animal berbagi method yang diwarisi dari Animal yaitu, move, eat, dan canSee.

#### Polimorfisme :

Polymorphisme pada game ini berada pada kelas Dolphin dan Fish. Kelas Dolphin dan Fish sama-sama memiliki kelas move() tetapi berbeda bentuknya.

#### Enkapsulasi :

Enkapsulasi pada game ini berada pada kelas Animal berupa variable speed yang bersifat private, yang berarti hanya bisa diakses oleh kelasnya.

Overriding :

Overriding pada game ini berada pada kelas Fish. kelas Fish memiliki method method move() yang diwarisi dari kelas Animal tetapi pada kelas Fish method move() ditulis ulang sehingga memiliki fungsi yang berbeda.