Praktikum Pemrograman Berorientasi Objek
INF2153

Laporan Praktikum
Ujian Akhir Semester

Oleh :
Syahbani Pratama – 2211102441094
Moh. Hikmal Yusuf – 2211102441158
Thifal Ngazizul Azmi – 2211102441232

Teknik Informatika
Fakultas Sains & Teknologi
Universitas Muhammadiyah Kalimantan Timur

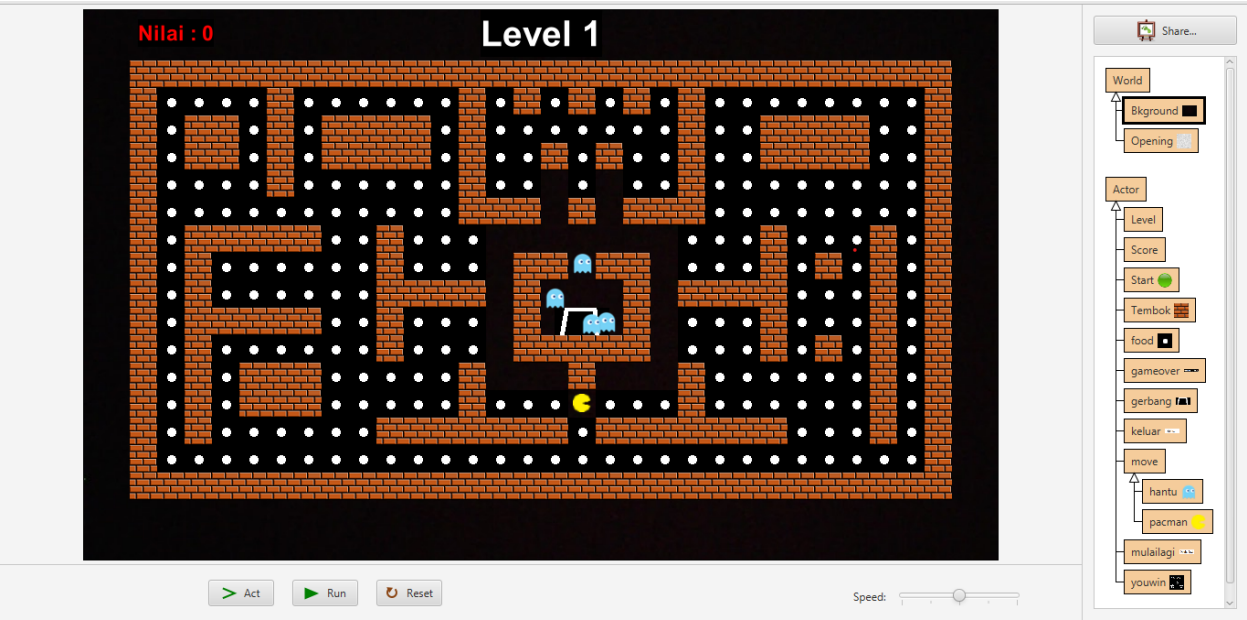
Samarinda, 2023

Pendahuluan

Pada praktikum ini, kami membuat permainan sederhana menggunakan platform Greenfoot dengan bahasa pemrograman Java. Tujuan praktikum ini adalah untuk memahami konsep OOP (Object-Oriented Programming) dalam pengembangan permainan. Permainan yang kami buat adalah permainan Pac-Man sederhana.

Deskripsi Singkat

Dalam proyek ini, kami menggunakan Greenfoot sebagai lingkungan pengembangan yang memfasilitasi visualisasi konsep-konsep pemrograman berorientasi objek. Greenfoot membantu dalam representasi grafis entitas permainan serta interaksi antara entitas tersebut.



- **Bkground**
  Kelas ini adalah kelas utama yang mewakili dunia permainan. Berisi logika untuk membuat dunia, level, dan objek-objek permainan seperti Pac-Man, hantu, makanan, serta menang atau kalahnya permainan.

  Atribut :
  backgroundImage : Menyimpan gambar latar belakang.

  Metode :
  - setBkgroundImage() : Mengatur gambar latar belakang.
  - scroll() : Menggerakkan latar belakang (parallax scrolling).

```
import greenfoot.*;
import java.util.List;

import java.awt.*;

public class Bkground extends World
{
    public int levelup = 1;
    private mulailagi lagi;
    private keluar exit;
    private Score updatescore;
    //public move mv = new move();
    private static GreenfootSound start = new GreenfootSound("start.wav");
    private static GreenfootSound win = new GreenfootSound("winning.wav");
    public Bkground()
    {
        // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
        super(1000, 600, 1);

setPaintOrder(mulailagi.class,keluar.class,youwin.class,Tembok.class,pacman.c
lass,hantu.class);
        start.play();
        chooselvl();
    }

    public void chooselvl(){
        if(levelup == 1)
            Level_1();
        else if(levelup == 2)
            Level_2();
        else if(levelup ==3)
            Level_3();
        else
            GameWin();
    }

    private void Level_2(){
        removeObjects(getObjects(Actor.class));
        addObject(new Level(levelup), 500, 25);
        int[] Level2 = {
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                0,2,2,2,2,2,2,2,0,2,2,2,2,2,2,2,0,2,2,2,2,0,2,2,2,2,2,2,2,0,
                0,2,0,0,0,2,2,2,0,2,2,2,2,0,0,0,0,0,0,0,2,0,2,2,2,0,0,0,2,0,
```

```
            0,2,0,2,2,2,0,2,0,0,0,0,2,2,2,2,0,2,2,2,2,0,2,0,2,2,2,0,2,0,
            0,2,0,2,0,2,0,2,2,2,2,2,2,2,2,2,0,2,2,2,2,2,2,0,2,0,2,0,2,0,
            0,2,2,2,0,2,0,0,0,2,0,0,2,2,0,0,0,0,0,2,2,0,0,0,2,0,2,2,2,0,
            0,2,2,2,0,2,0,2,2,2,2,2,2,1,1,1,1,1,1,1,2,2,2,0,2,0,2,2,2,0,
            0,0,0,0,0,2,0,2,2,2,2,2,2,1,0,0,1,0,0,1,2,2,2,0,2,0,0,0,0,0,
            0,0,0,0,0,2,2,2,2,2,2,0,0,2,1,0,3,1,3,0,1,2,2,2,2,2,2,0,0,0,0,0,
            0,2,2,2,0,2,0,0,0,2,0,0,2,1,0,3,3,3,0,1,2,0,0,0,2,0,2,2,2,0,
            0,2,2,2,0,2,0,2,2,2,2,2,2,1,0,0,0,0,0,1,2,2,2,0,2,0,2,2,2,0,
            0,2,0,2,0,2,0,2,0,2,2,2,2,2,1,1,1,0,1,1,1,2,0,2,0,2,0,2,0,2,0,
            0,2,0,2,2,2,2,2,0,2,0,0,2,2,2,2,1,2,2,2,2,0,2,2,2,2,2,0,2,0,
            0,2,0,0,0,2,0,0,0,2,2,2,2,0,0,0,0,0,0,0,2,0,0,0,2,0,0,0,2,0,
            0,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,0,2,2,2,2,2,2,2,2,2,2,0,
            0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        };
    addObject(new gerbang(),545,340);
    updatescore = new Score();
    addObject(updatescore, 100, 25);
    addObject(new pacman(),545,429);

    createArena(Level2);
}

private void Level_3(){
    removeObjects(getObjects(Actor.class));
    addObject(new Level(levelup), 500, 25);
    int[] Level3 = {
            0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
            0,2,2,2,0,2,2,2,2,2,0,2,2,2,0,0,2,2,2,2,2,2,2,2,2,2,2,2,2,0,
            0,2,0,2,2,2,0,2,0,2,2,2,0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,
            0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,0,2,0,
            0,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,0,
            0,2,0,0,0,2,2,0,2,0,0,0,0,2,0,0,0,0,0,2,0,0,0,2,0,2,0,0,2,0,
            0,2,2,2,2,2,2,2,0,2,2,2,2,2,0,1,1,1,1,1,1,1,0,2,2,2,0,2,2,2,2,0,
            0,2,0,0,2,0,2,0,0,0,0,2,0,1,0,0,1,0,0,1,0,2,0,0,0,2,0,2,2,0,
            0,2,2,0,2,0,2,0,2,2,2,2,2,1,0,3,3,3,0,1,2,2,2,2,0,2,0,2,2,0,
            0,2,2,0,0,0,2,0,2,2,0,0,0,1,0,3,3,3,0,1,0,0,0,2,0,2,0,0,2,0,
            0,2,2,2,2,0,2,2,2,2,0,2,2,1,0,0,0,0,0,1,2,2,0,2,2,2,0,2,2,0,
            0,2,0,0,0,0,0,0,0,2,0,2,0,1,1,1,0,1,1,1,0,2,0,2,0,0,0,2,2,0,
            0,2,2,2,2,2,2,2,2,2,2,2,0,2,2,2,1,2,2,2,0,2,2,2,2,2,2,2,2,0,
            0,2,2,0,2,0,2,0,2,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,2,2,0,2,2,0,
            0,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,0,
            0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        };
```

```
        addObject(new gerbang(),545,340);
        updatescore = new Score();
        addObject(updatescore, 100, 25);
        addObject(new pacman(),545,429);

        createArena(Level3);
    }

    private void Level_1(){
        removeObjects(getObjects(Actor.class));
        addObject(new Level(levelup), 500, 25);
        int[] Level1 = {
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                0,2,2,2,2,2,0,2,2,2,2,2,2,0,2,0,2,0,2,0,2,0,2,2,2,2,2,2,2,2,2,0,
                0,2,0,0,2,0,2,0,0,0,0,2,0,2,2,2,2,2,2,2,2,0,2,2,0,0,0,0,2,2,0,
                0,2,0,0,2,0,2,0,0,0,0,2,0,2,2,0,2,0,2,2,0,2,2,0,0,0,0,2,2,0,
                0,2,2,2,2,2,0,2,2,2,2,2,2,2,0,2,2,1,2,1,2,2,0,2,2,2,2,2,2,2,2,0,
                0,2,2,2,2,2,2,2,2,2,2,2,2,2,0,0,0,0,1,0,1,0,0,0,2,2,2,2,2,2,2,2,0,
                0,2,0,0,0,0,0,2,2,0,2,2,2,1,1,1,1,1,1,1,2,2,2,0,2,2,2,0,2,2,0,2,0,
                0,2,0,2,2,2,2,2,2,2,0,2,2,2,1,0,0,1,0,0,1,2,2,2,0,2,0,2,0,2,2,0,2,0,
                0,2,0,2,2,2,2,2,2,2,2,0,0,0,0,1,0,1,3,1,0,1,0,0,0,0,2,2,2,0,2,2,0,2,0,
                0,2,0,0,0,0,0,2,2,0,2,2,2,2,1,0,3,3,3,0,1,2,2,2,2,0,2,2,2,2,0,2,2,0,2,0,
                0,2,0,2,2,2,2,2,2,2,2,0,2,2,2,1,0,0,0,0,0,0,1,2,2,2,2,0,2,0,2,0,2,2,0,2,0,
                0,2,0,2,0,0,0,2,2,2,2,2,2,0,1,1,1,0,1,1,1,0,2,2,2,2,2,2,2,2,0,2,2,0,2,0,
                0,2,0,2,0,0,0,2,2,2,2,2,2,2,0,2,2,2,1,2,2,2,2,0,2,2,2,2,2,2,2,2,0,2,2,0,2,0,
                0,2,0,2,2,2,2,2,2,2,2,0,0,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0,2,2,2,2,0,2,2,0,2,0,
                0,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,0,
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
            };
        addObject(new gerbang(),545,340);
        updatescore = new Score();
        addObject(updatescore, 100, 25);
        addObject(new pacman(),545,429);

        createArena(Level1);
    }

    public Score Update(){
        return updatescore;
    }

    private void createArena(int arena[]){
        int a=0;
```

```
        for(int y=70;y<18*30;y+=30){
            for(int x=65;x<32*30;x+=30){
                int b=arena[a];
                if(b==0){
                    addObject(new Tembok(),x,y);}
                else if(b==1){}
                else if(b==2){
                    addObject(new food(),x,y);
                }else if(b==3){
                    addObject(new hantu(),x,y);}
                if(a<arena.length-1)
                    a++;
            }
        }
    }

    private void GameWin(){
        win.play();
        List mv = getObjects(move.class);
        for(int i = 0;i<mv.size();i++) {
            move mov = (move) mv.get(i);
            mov.bisagerak = false; }
        addObject(new youwin(), 545,250);
        lagi = new mulailagi();
        addObject(lagi, 425,340);
        exit = new keluar();
        addObject(exit, 665,340);
    }

    public void GameOver(){
        List mv = getObjects(move.class);
        for(int i = 0;i<mv.size();i++) {
            move mov = (move) mv.get(i);
            mov.bisagerak = false; }
        addObject(new gameover(), 545,250);
        lagi = new mulailagi();
        addObject(lagi, 425,340);
        exit = new keluar();
        addObject(exit, 665,340);
    }

    public void act(){
        if(Greenfoot.mouseClicked(lagi)){
```

```
                Greenfoot.setWorld(new Bkground());
            }else if(Greenfoot.mouseClicked(exit)){
                Greenfoot.setWorld(new Opening());
                Greenfoot.stop();}
        }

    }
```

- **Opening**
  Kelas ini menggambarkan layar pembuka permainan. Saat layar ini ditampilkan, pengguna dapat memulai permainan dengan mengklik objek Start.

  Metode :
  showOpeningScreen() : Menampilkan teks atau gambar untuk layar pembukaan.

```
    import greenfoot.*;

    public class Opening extends World
    {
        private Start mulai;
        public Opening()
        {
            super(1000, 600, 1);

            mulai = new Start();
            addObject(mulai, 499, 389);
        }

        public void act(){
            if(Greenfoot.mouseClicked(mulai))
                Greenfoot.setWorld(new Bkground());

        }

        private void createArena(int arena[]){
            int a=0;
            for(int y=50;y<6*30;y+=30){
                for(int x=90;x<29*30;x+=30){
                    int b=arena[a];
                    if(b==0){
                        addObject(new Tembok(),x,y);}
                    else if(b==1){}
```

```
                    if(a<arena.length-1)
                        a++;
                }
            }
        }
}
```

- **Level**
  Menampilkan level permainan saat ini pada layar.

  Atribut :
  levelNumber: Menyimpan nomor level.

  Metode :
  - displayLevelNumber() : Menampilkan nomor level pada layar.
  - increaseLevel() : Menaikkan level permainan.

```
import greenfoot.*;

public class Level extends Actor
{
   public Level(int level){
      setImage(new GreenfootImage("Level " + level, 50, Color.WHITE,
Color.BLACK));
   }
}
```

- **Score**
  Merupakan tampilan skor pada layar. Berperan dalam memantau dan menampilkan skor saat pengguna bermain.

  Atribut :
  scoreValue: Menyimpan nilai skor.

  Metode :
  - displayScore() : Menampilkan nilai skor pada layar.
  - updateScore() : Memperbarui skor berdasarkan peristiwa dalam permainan.

```
import greenfoot.*;

public class Score extends Actor
{
```

```
    int nilai = 0;
    public Score(){
      setImage(new GreenfootImage("Nilai : " + nilai, 30, Color.RED,
Color.BLACK));
    }

   public void update(int a){
      nilai += a;
      setImage(new GreenfootImage("Nilai : " + nilai, 30, Color.RED,
Color.BLACK));
    }
}
```

- **Start**
  Objek yang digunakan untuk memulai permainan saat pengguna mengkliknya pada layar pembuka.

  Metode :
  startGame() : Memulai permainan setelah aksi tertentu dari pengguna.

```
import greenfoot.*;

public class Start extends Actor
{
    public void act()
    {
       // Add your action code here.
    }
}
```

- **Tembok**
  Representasi dari dinding atau rintangan dalam permainan.

  Metode :
  createWall() : Membuat tembok di area tertentu.

```
import greenfoot.*;

public class Tembok extends Actor
{
    public void act()
    {
```

```
            // Add your action code here.
        }
}
```

- **Food**
  Objek yang mewakili makanan yang harus dikonsumsi oleh Pac-Man.

  Metode :
  - displayFood() : Menampilkan makanan di posisi tertentu pada layar.
  - consume() : Menangani aksi ketika pemain memakan makanan.

```
import greenfoot.*;

public class food extends Actor
{
    public void act()
    {
        // Add your action code here.
    }
}
```

- **Gameover dan youwin**
  Menampilkan pesan game over atau you win saat permainan selesai.

  Metode :
  showGameOverScreen() : Menampilkan teks atau gambar untuk layar akhir permainan.

```
import greenfoot.*;

public class gameover extends Actor
{
    public void act()
    {
        // Add your action code here.
    }
}

import greenfoot.*;

public class youwin extends Actor
{
    public void act()
```

```
        {
            // Add your action code here.
        }
}
```

- **Gerbang dan Keluar**
  Objek yang digunakan untuk memindahkan pemain ke level berikutnya atau keluar dari permainan.

  Metode:
  - openGate() : Membuka gerbang ke level berikutnya.
  - moveToNextArea() : Memindahkan pemain ke area berikutnya setelah kondisi tertentu terpenuhi.
  - exitGame() : Keluar dari permainan dan menutup aplikasi.

```
import greenfoot.*;
public class gerbang extends Actor
{
    public void act()
    {
        if(isTouching(pacman.class)){
            Bkground ground = (Bkground)getWorld();
            ground.levelup++;
            ground.chooselvl();
        }
    }
}

import greenfoot.*;

public class keluar extends Actor
{
    public void act()
    {
        // Add your action code here.
    }
}
```

- **Move**
  Kelas dasar yang diwarisi oleh Pac-Man dan hantu. Berisi logika untuk gerakan objek di dalam permainan.

Metode :

- moveObject() : Menggerakkan objek dalam arah tertentu.
- checkBoundary() : Memeriksa batas pergerakan objek.

```
import greenfoot.*;

public class move extends Actor
{
   public boolean bisagerak = true;

   public boolean Hitwall(String arah){
      int x;
      int y;
      if(arah == "kiri"){
         x=getX()-2;
         y=getY();
      }else if(arah == "kanan"){
         x=getX()+2;
         y=getY();
      }else if(arah == "atas"){
         x=getX();
         y=getY()-2;
      }else{
         x=getX();
         y=getY()+2;}
      int Xttp = getX();
      int Yttp = getY();
      setLocation(x,y);
      Actor dinding = getOneIntersectingObject(Tembok.class);
      setLocation(Xttp,Yttp);
      return dinding == null;
   }

   public boolean Hit(Class cls){
      Actor ac = getOneIntersectingObject(cls);
      return ac!=null;
   }
   public void Hitghost(){
      Bkground world = (Bkground) getWorld();
      world.GameOver();
   }
}
```

- **Hantu**
  Karakter yang mengejar Pac-Man dalam permainan.

  Metode :
  - ghostMovement() : Mengatur pergerakan hantu dalam permainan.
  - checkPlayerCollision() : Memeriksa tabrakan antara hantu dan pemain.

```
import greenfoot.*;

public class hantu extends move
{
   private static GreenfootSound die = new GreenfootSound("die.wav");
   private String Arah = null;
   public void act()
   {
     if(bisagerak){
        gerak();
        int x = getX();
        int y = getY();
        if(x == 545 && y == 310)
            setLocation(545,308);
        else if(x == 545 && y == 308)
            setLocation(545,306);
        else if(x == 545 && y == 306)
            setLocation(545,304);
        else if(x == 545 && y == 304)
            setLocation(545,302);
        else if(x == 545 && y == 302)
            setLocation(545,300);
        else if(x == 545 && y == 300)
            setLocation(545,298);
        else if(x == 545 && y == 298)
            setLocation(545,296);
        else if(x == 545 && y == 296)
            setLocation(545,294);

        if(Hit(pacman.class)){
            die.play();
            Hitghost();}

       }
```

```
        }

        private void gerak(){
            if(Arah!=null && Hitwall(Arah))
                jalan(Arah);
            else{
                int acak = Greenfoot.getRandomNumber(4);
                setArah(acak);
                gerak();
            }
        }

        private void jalan(String arah){
            if(arah == "kanan")
                setLocation(getX()+2,getY());
            if(arah == "kiri")
                setLocation(getX()-2,getY());
            if(arah == "atas")
                setLocation(getX(),getY()-2);
            if(arah == "bawah")
                setLocation(getX(),getY()+2);
        }

        private String setArah(int random){
            if(random == 0)
                Arah = "kanan";
            else if(random == 1)
                Arah = "bawah";
            else if(random == 2)
                Arah = "kiri";
            else
                Arah = "atas";
            return Arah;
        }
}
```

- **Pacman**
  Karakter utama yang dikendalikan oleh pengguna dalam permainan.

  Metode :
  - movePacman() : Mengatur pergerakan Pacman berdasarkan input pengguna.
  - checkFoodCollision() : Memeriksa interaksi antara Pacman dan makanan.

```
import greenfoot.*;

public class pacman extends move
{
   private static GreenfootSound eat = new GreenfootSound("wakka.wav");
   public void act(){
      if(bisagerak){
         String arah = ArahGerak();
         Bkground ground = (Bkground) getWorld();
         Score Nilai = ground.Update();
         if(Hitwall(arah)){
            arahTerakhir = arah;
            gerak(arah);
         }else if(Hitwall(arahTerakhir)){
            gerak(arahTerakhir);
         }if(Hit(food.class)){
            eat.play();
            removeTouching(food.class);
            Nilai.update(10);}
      }
   }

   private String ArahGerak(){
      String arah = null;
      if(Greenfoot.isKeyDown("Left"))
         arah = "kiri";
      if(Greenfoot.isKeyDown("Right"))
         arah = "kanan";
      if(Greenfoot.isKeyDown("Up"))
         arah = "atas";
      if(Greenfoot.isKeyDown("Down"))
         arah = "bawah";
      if(arah!=null)
         return arah;
      else
         return arahTerakhir;
   }

   private void gerak (String arah){
      if(arah == "kanan"){
         setLocation(getX()+2,getY());
         setRotation(0);
      }else if(arah == "kiri"){
```

```
                setLocation(getX()-2,getY());
                setRotation(180);
            }else if(arah == "atas"){
                setLocation(getX(),getY()-2);
                setRotation(270);
            }else if(arah == "bawah"){
                setLocation(getX(),getY()+2);
                setRotation(90);}
        }

        private String arahTerakhir = null;
}
```

- **Mulailagi**
  Objek yang digunakan untuk memulai kembali permainan setelah game over atau menang.

```
    import greenfoot.*;

    public class mulailagi extends Actor
    {
        public void act()
        {
            // Add your action code here.
        }
}
```

Deskripsi Elemen Pada Program

1. Kelas dan Objek

Dalam pengembangan game ini, terdapat beberapa kelas yang mewakili berbagai elemen dalam permainan seperti Bkground, Opening, Level, Score, Start, Tembok, food, gameover, gerbang, keluar, move, hantu, dan pacman. Setiap kelas merepresentasikan objek tertentu dalam permainan dan memiliki perilaku serta atributnya sendiri.

Contoh Codingan Level Class :

```
public class Level extends Actor {
    public Level(int level){
        setImage(new GreenfootImage("Level " + level, 50, Color.WHITE, Color.BLACK));
    }
    // ...
}
```

Kelas Level bertanggung jawab untuk menampilkan level permainan pada layar. Constructor menerima parameter level dan menampilkan teks "Level [level]" dengan ukuran font 50, warna putih, dan latar belakang hitam.

2. Pewarisan (Inheritance)

Konsep pewarisan digunakan untuk membangun hubungan antara kelas yang terkait. Misalnya, kelas Level, Score, gameover, dan youwin mewarisi sifat-sifat dasar dari kelas Actor dalam Greenfoot.

Contoh codingan dari gameover class :

```
public class gameover extends Actor {
    // ...
}
```

Kelas gameover merupakan subkelas dari Actor dalam Greenfoot. Ini berarti gameover mewarisi properti dan metode dari kelas Actor, yang memungkinkannya untuk diaktifkan sebagai objek dalam permainan.

3. Polimorfisme

Dalam permainan, polimorfisme dapat diamati dalam berbagai perilaku objek. Contohnya, objek hantu dan pacman memiliki perilaku unik meskipun keduanya adalah turunan dari kelas move.

Contoh codingan dari hantu class :

```
public class hantu extends move {
    // ...
}
```

Kelas hantu merupakan turunan dari kelas move yang mengatur perilaku hantu dalam permainan. Meskipun menggunakan logika gerakan dari kelas induknya (move), kelas ini memiliki perilaku yang berbeda dalam interaksi dan jalannya.

4. **Enkapsulasi**

Enkapsulasi digunakan untuk melindungi data dan metode dari akses yang tidak diinginkan. Dalam kode, kita bisa menggunakan access modifiers seperti private, public, dan protected untuk mengatur akses terhadap atribut dan metode.

Contoh codingan dari Score class:

```
public class Score extends Actor {
    private int nilai = 0;
    // ...
}
```

Atribut nilai dalam kelas Score dienkapsulasi sebagai private untuk melindungi nilainya dari akses langsung. Hanya metode dalam kelas Score yang dapat mengubah nilai tersebut.

5. **Interaksi Antar Objek**

Dalam permainan ini, terdapat interaksi antara objek seperti pacman yang dapat memakan food atau bertabrakan dengan hantu, yang menghasilkan perubahan nilai atau kejadian tertentu dalam permainan.

Contoh codingan dari pacman class:

```
public class pacman extends move {
    // ...
}
```

Kelas pacman berinteraksi dengan objek food dan hantu. Ketika pacman menyentuh objek food, nilai akan bertambah, sementara jika bertabrakan dengan hantu, permainan akan berakhir.

6. Overriding dan Overloading

Overriding digunakan untuk mengubah perilaku metode dari superclass di subclass, sementara overloading memungkinkan pembuatan metode dengan nama yang sama tetapi parameter yang berbeda.

Contoh codingan dari move class:

```
public class move extends Actor {
    public boolean Hit(Class cls){
        // ...
    }
    public boolean Hit(move obj){
        // ...
    }
    // ...
}
```

Kelas move memiliki metode Hit yang memiliki dua versi yang berbeda. Salah satunya menerima parameter berupa kelas (Class cls), sementara yang lain menerima objek move. Hal ini menunjukkan overloading pada metode Hit.

Kesimpulan

Implementasi konsep PBO/OOP dalam pengembangan game menggunakan Greenfoot membantu dalam menyusun struktur kode yang terorganisir dan mempermudah pengelolaan serta pemeliharaan kode. Penggunaan konsep seperti pewarisan, polimorfisme, enkapsulasi, interaksi antar objek, overriding, dan overloading memberikan kerangka kerja yang kuat untuk mengembangkan game yang dinamis dan menarik.