Laporan Ujian Akhir Semester
Pemrograman Berorientasi Objek

Kelompok 11 :

Wahyudi 2211102441144

Nudziya Salma Fauziah 2211102441046

Tyara Andini 2211102441167

UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR

TAHUN 2023

❖ Codingan

GameOver

```
import greenfoot.*;

public class GameOver extends World
{
   Player1 player1 = new Player1();
   /**
    * Constructor for objects of class GameOver.
    *
    */
   public GameOver()
   {
      // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
      super(600, 400, 1);
      GreenfootImage go = getBackground();

      go.scale(go.getWidth()-400, go.getHeight()-600);
      setBackground(go);

      Greenfoot.playSound("GameOver.mp3");

      prepare();
   }

   public void Act()
   {
      showText("Score: " + player1.score,50,50);
   }

   private void prepare()
   {
      back Back = new back();
      addObject(Back,40,370);
   }

}
```

Penjelasan :
Kelas GameOver yang merupakan subkelas dari kelas World dalam lingkungan Greenfoot. Berikut adalah penjelasan untuk setiap bagian dari kode tersebut:
1. Kelas GameOver: Merupakan kelas yang mewarisi kelas World dalam lingkungan Greenfoot, yang digunakan untuk membuat dunia (world) dalam permainan.
2. Konstruktor GameOver(): Membuat objek GameOver, menginisialisasi dunia dengan ukuran 600x400 sel dengan ukuran 1x1 piksel. Mengatur latar belakang dan memainkan suara "GameOver.mp3”.
3. Method prepare(): Membuat dan menambahkan objek back ke dunia pada koordinat tertentu.

❖ Codingan

Game Over1

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

/**
 * Write a description of class GameOver1 here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class GameOver1 extends World
{

   Player1 player1 = new Player1();
   /**
    * Constructor for objects of class GameOver1.
    *
    */
   public GameOver1()
   {
      // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
      super(600, 400, 1);
       GreenfootImage go = getBackground();

      go.scale(go.getWidth()-400, go.getHeight()-600);
      setBackground(go);

      Greenfoot.playSound("GameOver.mp3");

      prepare();
   }

   public void Act()
   {
      showText("Score: " + player1.score,50,50);
   }

   private void prepare()
   {
      back Back = new back();
      addObject(Back,40,370);
   }
}
```

Penjelasan:
Kelas GameOver1 yang merupakan subkelas dari World dalam lingkungan Greenfoot.
1. Deklarasi Kelas GameOver1:
GameOver1 adalah sebuah kelas yang mewarisi kelas World di lingkungan Greenfoot. Ini menandakan bahwa GameOver1 akan menjadi dunia (world) dalam permainan.
2. Variabel dan Konstruktor:
- Variabel player1 bertipe Player1 dideklarasikan tetapi tidak digunakan dalam kode yang diberikan.
- Konstruktor GameOver1() digunakan untuk membuat objek GameOver1.
- Konstruktor ini membangun dunia dengan ukuran 600x400 sel dengan ukuran 1x1 piksel.
- Mengubah skala gambar latar belakang dan mengaturnya sebagai latar belakang dunia.
- Memainkan suara "GameOver.mp3".
- Memanggil method prepare().
3. Method Act():
- Metode ini mencoba menampilkan skor dari player1 di koordinat (50,50) dalam dunia permainan
4. Method prepare():
- Metode ini membuat objek back dan menambahkannya ke dunia pada koordinat (40,370).

❖ Codingan

MyWorld

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

/**
 * Write a description of class MyWorld here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class MyWorld extends World
{

   /**
    * Constructor for objects of class MyWorld.
    *
    */
   public MyWorld()
   {
      // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
      super(600, 400, 1);
      GreenfootImage image = getBackground();

      image.scale(image.getWidth()-1200, image.getHeight()-800);
      setBackground(image);

      Greenfoot.playSound("awal.mp3");
      prepare();
   }

   private void prepare()
   {
      start Start = new start();
      addObject(Start,230,250);
      info Info = new info();
      addObject(Info,367,250);
   }
}
```

Penjelasan:

Kelas MyWorld yang mewarisi kelas World dalam lingkungan Greenfoot. Berikut adalah penjelasannya :
1. Deklarasi Kelas MyWorld:
- MyWorld adalah kelas yang mewarisi kelas World di lingkungan Greenfoot. Ini menandakan bahwa - - MyWorld akan menjadi dunia (world) dalam permainan.
2. Konstruktor MyWorld():
- Konstruktor ini digunakan untuk membuat objek MyWorld.
- Membangun dunia dengan ukuran 600x400 sel dengan ukuran 1x1 piksel.
- Mengambil gambar latar belakang dan mengubahnya menjadi lebih kecil dengan mengubah skala gambar sebelum menetapkannya sebagai latar belakang dunia.
- Memainkan suara "awal.mp3".
- Memanggil method prepare().
3. Method prepare():
- Metode ini menyiapkan elemen-elemen awal di dalam dunia.
- Membuat objek start (mungkin merupakan tombol untuk memulai permainan) dan menambahkannya ke dunia pada koordinat (230, 250).
- Membuat objek info (mungkin merupakan tombol untuk menampilkan informasi permainan) dan menambahkannya ke dunia pada koordinat (367, 250).
Kode ini bertanggung jawab untuk membuat dunia awal permainan dengan menyiapkan elemen-elemen dasar seperti tombol untuk memulai permainan dan tombol informasi, serta menetapkan latar belakang dan suara awal permainan.

❖ Codingan

Forminfo

```
import greenfoot.*;

public class forminfo extends World
{
   public forminfo()
   {
      // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
      super(600, 400, 1);

      GreenfootImage image = getBackground();

      image.scale(image.getWidth()-1200, image.getHeight()-800);
      setBackground(image);

      prepare();
   }
   private void prepare()
   {
      back Back = new back();
      addObject(Back,40,40);
   }
}
```

Penjelasan:
kelas forminfo yang merupakan subkelas dari kelas World dalam lingkungan Greenfoot. Berikut adalah penjelasannya :
1. Deklarasi Kelas forminfo:
- forminfo adalah kelas yang mewarisi kelas World di lingkungan Greenfoot. Ini menandakan bahwa - - forminfo akan menjadi dunia (world) dalam permainan.
2. Konstruktor forminfo():
- Konstruktor ini digunakan untuk membuat objek forminfo.
- Membangun dunia dengan ukuran 600x400 sel dengan ukuran 1x1 piksel.
- Mengambil gambar latar belakang dan mengubahnya menjadi lebih kecil dengan mengubah skala gambar sebelum menetapkannya sebagai latar belakang dunia.
- Memanggil method prepare().
3. Method prepare():
- Metode ini menyiapkan elemen-elemen yang akan ditampilkan di dalam dunia.
- Membuat objek back (mungkin merupakan tombol kembali) dan menambahkannya ke dunia pada koordinat (40, 40).
Kelas forminfo bertanggung jawab untuk membuat dunia yang mungkin digunakan untuk menampilkan informasi atau formulir dalam permainan, dengan menyediakan tombol kembali dan menyesuaikan latar belakang dunia sesuai dengan kebutuhan formulir atau informasi yang ditampilkan.

❖ Codingan

Mulai

```
import greenfoot.*;

public class mulai extends World
{
   private int timer;
   Player1 player1 = new Player1();
   public mulai()
   {
      super(600, 400, 1);
      GreenfootImage go = getBackground();

      go.scale(go.getWidth()+440, go.getHeight()+240);
      setBackground(go);

      prepare();
      Greenfoot.playSound("mulai.mp3");
      timer = 0;
      showScore();
   }
   public void act()
   {
      countTime();
      randomSpawn();
      showScore();
   }
   private void prepare()
   {
      addObject(player1,102,320);
   }
   public void countTime()
   {
      timer = timer + 1;
      showTime();
      if(timer == 2000)
      {
         showText("Time's Up!", 300 ,150);
         player1.getScore();
         {
         showText("Score: " + player1.getScore(),550,50);
         Greenfoot.setWorld(new GameOver());
         }

         {
```

```
            showText("Score: " + player1.getScore(),550,50);
            Greenfoot.setWorld(new GameOver1());
            }
        }
    }

    public void showTime()
    {
        showText("Time: " + timer, 500, 25);
    }
    public void randomSpawn()
    {
        if (Greenfoot.getRandomNumber(500) < 7)
        {
            Cherry cherry = new Cherry();
            addObject(cherry, Greenfoot.getRandomNumber(600),
Greenfoot.getRandomNumber(1));

            Bananas bananas = new Bananas();
            addObject(bananas,Greenfoot.getRandomNumber(600),
Greenfoot.getRandomNumber(1));

            Apel apel = new Apel();
            addObject(apel,Greenfoot.getRandomNumber(600),
Greenfoot.getRandomNumber(1));
        }
        if(Greenfoot.getRandomNumber(500)< 3)
        {
            Bomb bomb = new Bomb();
            addObject(bomb,Greenfoot.getRandomNumber(600),
Greenfoot.getRandomNumber(1));

        }
    }
    public void showScore()
    {
        showText("Score: " + player1.getScore(),50,50);
    }
}
```

Penjelasan:

kelas mulai yang mewarisi kelas World dalam lingkungan Greenfoot. Berikut adalah penjelasan singkat tentang kode tersebut:

Deklarasi Kelas mulai:

mulai adalah kelas yang mewarisi kelas World dalam Greenfoot, yang menunjukkan bahwa mulai akan menjadi dunia (world) dalam permainan.

Variabel dan Konstruktor:

Variabel timer digunakan untuk melacak waktu dalam permainan.

Variabel player1 bertipe Player1 dideklarasikan dan diinisialisasi dalam kelas mulai.

Konstruktor mulai():

Membangun dunia dengan ukuran 600x400 sel dengan ukuran 1x1 piksel.

Mengubah skala gambar latar belakang dunia.

Memanggil method prepare().

Memainkan suara "mulai.mp3".

Inisialisasi variabel timer ke nilai 0.

Menampilkan skor pemain.

Method act():

Metode ini merupakan bagian penting yang dipanggil secara otomatis oleh Greenfoot pada setiap iterasi permainan.

Memanggil method countTime() untuk menghitung waktu permainan.

Memanggil method randomSpawn() untuk penempatan acak objek dalam dunia.

Memanggil method showScore() untuk menampilkan skor pemain.

Method prepare():

Metode ini menyiapkan elemen-elemen awal dalam dunia.

Menambahkan objek player1 ke dunia pada posisi tertentu.

Method countTime():

Menghitung waktu permainan dan menampilkan waktu.

Jika waktu mencapai 2000, menampilkan "Time's Up!" dan skor terakhir pemain.

Membuat objek GameOver atau GameOver1 sesuai kondisi tertentu berdasarkan skor pemain.

Method showTime():

Menampilkan waktu tersisa di layar permainan.

Method randomSpawn():

Menambahkan objek Cherry, Bananas, Apel, atau Bomb secara acak ke dunia berdasarkan nilai acak yang dihasilkan.

Method showScore():

Menampilkan skor pemain di layar permainan.

Kelas mulai bertanggung jawab untuk mengatur gameplay dalam permainan, menghitung waktu, menampilkan skor, dan menambahkan objek secara acak ke dunia berdasarkan kondisi tertentu. Selain itu, kelas ini juga menangani logika untuk menampilkan pesan "Time's Up!" dan memutuskan dunia mana yang harus dibuat berdasarkan skor pemain setelah waktu berakhir.

❖ Codingan

Apel

```
import greenfoot.*;

public class Apel extends Actor
{
   public void act()
   {
      setLocation(getX(), getY()+3);
      removeAtEdge();
   }
    public void removeAtEdge()
   {
      if( isAtEdge() )
      {
         getWorld().removeObject(this);
      }
   }
}
```

Penjelasan:
kelas Apel yang merupakan subkelas dari Actor dalam lingkungan Greenfoot. Berikut adalah penjelasannya :
1. Deklarasi Kelas Apel:
- Apel adalah kelas yang mewarisi kelas Actor dalam Greenfoot. Kelas Actor adalah kelas dasar untuk semua objek yang berinteraksi di dunia permainan.
2. Method act():
- Metode act() adalah metode yang dipanggil secara terus-menerus oleh Greenfoot pada setiap frame permainan.
- Dalam metode ini, posisi Apel diperbarui dengan memindahkan objek Apel ke bawah (sumbu Y) sejauh 3 piksel setiap kali metode act() dipanggil. Ini dilakukan dengan menggunakan setLocation(getX(), getY()+3) yang mengubah posisi Y Apel.
- Terdapat pemanggilan method removeAtEdge() untuk memeriksa apakah objek Apel telah mencapai tepi layar permainan.
3. Method removeAtEdge():
- Metode ini bertanggung jawab untuk menghapus objek Apel dari dunia jika objek tersebut berada di tepi layar permainan.
- Dengan menggunakan isAtEdge(), kondisi dicek untuk memverifikasi apakah objek Apel telah mencapai tepi layar permainan.
- Jika Apel berada di tepi layar, pemanggilan getWorld().removeObject(this) akan menghapus objek Apel dari dunia.
Dengan demikian, kelas Apel di sini mengontrol perilaku objek Apel dalam permainan. Saat act() dipanggil, objek Apel bergerak ke bawah setiap frame, dan jika objek tersebut mencapai tepi layar permainan, maka objek Apel akan dihapus dari dunia.

Catatan: Codingan dan Penjelasan tersebut berlaku pada subclass Apel, Bananas, dan Chery

❖ Codingan

Bomb

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

/**
 * Write a description of class Bomb here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
on gets pressed in the environment.
     */
    public void act()
    {
        setLocation(getX(), getY()+3);
        removeAtEdge();// Add your action code here.
    }
    public void removeAtEdge()
    {
        if( isAtEdge() )
        {
            getWorld().removeObject(this);
        }
    }

    public void meledak(){
        World world = getWorld();
        world.addObject(new Duar(), getX(), getY());
    }
}
```

Penjelasan:
1. Deklarasi Kelas Bomb:
- Kelas Bomb adalah kelas yang mewarisi kelas Actor dalam lingkungan Greenfoot. Ini menunjukkan bahwa Bomb merupakan objek yang dapat berinteraksi dalam dunia permainan.
2. Method act():
- Metode act() adalah metode yang dipanggil secara teratur oleh Greenfoot pada setiap siklus permainan.
- Dalam potongan kode yang diberikan, terdapat instruksi setLocation(getX(), getY()+3), yang berarti bahwa objek Bomb akan bergerak ke bawah (sumbu Y) sejauh 3 piksel setiap kali metode act() dipanggil.
- Terdapat pemanggilan method removeAtEdge() untuk memeriksa apakah objek Bomb telah mencapai tepi layar permainan.
3. Method removeAtEdge():
- Metode ini bertanggung jawab untuk menghapus objek Bomb dari dunia jika objek tersebut berada di tepi layar permainan.
- Dengan menggunakan isAtEdge(), kondisi dicek untuk memverifikasi apakah objek Bomb telah mencapai tepi layar permainan.
- Jika Bomb berada di tepi layar, pemanggilan getWorld().removeObject(this) akan menghapus objek Bomb dari dunia.
4. Method meledak():
- Metode meledak() bertanggung jawab untuk menambahkan objek Duar ke dunia permainan pada posisi yang sama dengan objek Bomb.
- Menggunakan getWorld().addObject(new Duar(), getX(), getY()), objek Duar (mungkin sebagai visualisasi ledakan) ditambahkan ke dunia pada koordinat yang sama dengan objek Bomb.
- Keseluruhan kode ini, jika ditulis dengan benar, mengontrol perilaku dari objek Bomb dalam permainan, termasuk pergerakan, penghapusan saat mencapai tepi layar, dan menambahkan efek ledakan ke dunia permainan.

❖ Codingan

Duar

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

public class Duar extends Actor
{
   public static final int DELAY = 5;
   public int explodeCounter = 0;
   ArrayList<GreenfootImage> listImage = new ArrayList<GreenfootImage>();
   int currentIndex = 0;
   public Duar()
   {
      listImage.add(new GreenfootImage("tile001.png"));
      listImage.add(new GreenfootImage("tile003.png"));
      listImage.add(new GreenfootImage("tile005.png"));
      listImage.add(new GreenfootImage("tile007.png"));
      listImage.add(new GreenfootImage("tile009.png"));
      listImage.add(new GreenfootImage("tile011.png"));
      listImage.add(new GreenfootImage("tile013.png"));
      listImage.add(new GreenfootImage("tile015.png"));
      listImage.add(new GreenfootImage("tile017.png"));
      listImage.add(new GreenfootImage("tile019.png"));
      listImage.add(new GreenfootImage("tile021.png"));
      listImage.add(new GreenfootImage("tile023.png"));
      listImage.add(new GreenfootImage("tile024.png"));
   }

   public void act()
   {
      if(explodeCounter >= DELAY){
         setImage(listImage.get(currentIndex));
         currentIndex++;
         explodeCounter = 0;
      }
      explodeCounter++;

      if(currentIndex >= listImage.size()){
         World world = getWorld();
         world.removeObject(this);
      }
      // Add your action code here.
   }
}
```

Penjelasan;
kelas Duar yang merupakan subkelas dari Actor dalam lingkungan Greenfoot. Ini bertanggung jawab untuk menampilkan efek ledakan dengan menggunakan serangkaian gambar yang berbeda secara berurutan.
Berikut adalah penjelasan singkat tentang kode tersebut:
1. Deklarasi Kelas Duar:
- Duar adalah kelas yang mewarisi kelas Actor dalam Greenfoot. Ini berarti bahwa Duar adalah objek yang dapat berinteraksi di dunia permainan.
2. Variabel dan Konstruktor:
- DELAY merupakan konstanta yang menentukan kecepatan perubahan gambar dalam efek ledakan.
- explodeCounter digunakan untuk menghitung waktu sebelum beralih ke gambar ledakan berikutnya.
- listImage adalah ArrayList yang berisi serangkaian objek GreenfootImage yang digunakan untuk efek ledakan.
- currentIndex menyimpan indeks gambar saat ini dalam ArrayList listImage.
3. Konstruktor Duar():
- Membuat serangkaian GreenfootImage yang berisi gambar-gambar untuk efek ledakan.
4. Method act():
- Metode act() dipanggil secara teratur oleh Greenfoot pada setiap siklus permainan.
- Pada setiap siklus, metode ini memeriksa apakah waktunya untuk beralih ke gambar ledakan berikutnya (explodeCounter >= DELAY). Jika iya, gambar pada currentIndex diganti dengan gambar berikutnya dalam listImage.
- explodeCounter diinkrementasi untuk memperbarui hitungan waktu.
- Jika semua gambar ledakan sudah ditampilkan (currentIndex >= listImage.size()), objek Duar dihapus dari dunia.
Jadi, kelas Duar bertanggung jawab untuk menampilkan efek ledakan dengan menggunakan serangkaian gambar yang tersimpan dalam ArrayList listImage. Setiap gambar dalam listImage ditampilkan secara berurutan sesuai dengan waktu yang ditentukan oleh DELAY, dan setelah semua gambar ditampilkan, objek Duar akan dihapus dari dunia permainan.

❖ Codingan

Player

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

public class Players extends Actor implements iniinterface
{
   public static final int DELAY = 10;
   private boolean markForDeletion = false;

   public void act()
   {
      // Add your action code here.
      lookingForCherry();
      lookingForBomb();
      lookingForBananas();
      lookingForApel();
   }

   public void checkKeyPress (String left, String right, String down, String up)
   {

      if(Greenfoot.isKeyDown(left))
      {
         setLocation(getX()-3, getY());
      }
      if(Greenfoot.isKeyDown(right))
      {
         setLocation(getX()+3, getY());
      }
      if(Greenfoot.isKeyDown(down))
      {
         setLocation(getX(), getY()+3);
      }
      if(Greenfoot.isKeyDown(up))
      {
         setLocation(getX(), getY()-3);
      }
   }

   public int score=0;
   public int getScore()
   {
      return score;
   }
```

```
    public void lookingForCherry()
    {
        if( isTouching(Cherry.class) )
        {
            removeTouching(Cherry.class);
            score = score +25;

        }
    }

    public void lookingForBomb()
    {
        if( isTouching(Bomb.class) )
        {
            removeTouching(Bomb.class);
            score = score -25;
            Greenfoot.playSound("Explosion.wav");
            meledak();
        }
    }

    public void lookingForBananas()
    {
        if( isTouching(Bananas.class) )
        {
            removeTouching(Bananas.class);
            score = score +50;
        }
    }

    public void lookingForApel()
    {
        if( isTouching(Apel.class) )
        {
            removeTouching(Apel.class);
            score = score +30;
        }
    }

    public void meledak(){
        World world = getWorld();
        world.addObject(new Duar(), getX(), getY());
    }

}
```

Penjelasan:
kelas Players yang merupakan subkelas dari Actor dan mengimplementasikan sebuah interface bernama iniinterface (yang tidak terlihat dalam potongan kode yang Anda berikan).
Berikut adalah penjelasannya :
1. Deklarasi Kelas Players:
- Players adalah kelas yang mewarisi kelas Actor dalam Greenfoot. Kelas ini mewakili objek pemain dalam permainan.
- Variabel dan Konstanta:
- DELAY adalah konstanta yang menentukan kecepatan aksi dalam metode act().
- markForDeletion adalah variabel boolean yang menandakan apakah objek Players ditandai untuk dihapus.
2. Method act():
- Metode act() dipanggil secara teratur oleh Greenfoot pada setiap siklus permainan.
- Metode ini mengatur aksi yang dilakukan oleh objek Players. Pada bagian ini, terdapat pemanggilan beberapa metode seperti lookingForCherry(), lookingForBomb(), lookingForBananas(), dan lookingForApel().
3. Method checkKeyPress():
- Metode ini memeriksa penekanan tombol kontrol pada keyboard. Jika tombol tertentu ditekan, posisi objek Players akan bergerak sesuai dengan tombol yang ditekan.
- Variabel score:
- Variabel ini menyimpan skor dari pemain.
4. Method getScore():
- Metode ini mengembalikan nilai dari variabel score.
- Metode lookingForCherry(), lookingForBomb(), lookingForBananas(), dan lookingForApel():
- Metode-metode ini memeriksa apakah objek Players menyentuh objek Cherry, Bomb, Bananas, atau Apel. Jika iya, maka aksi tertentu akan diambil seperti menambah atau mengurangi skor serta menghapus objek yang bersentuhan.
5. Metode meledak():
- Metode ini bertanggung jawab untuk menambahkan objek Duar ke dunia permainan pada posisi yang sama dengan objek Players.
Kelas Players ini mengatur perilaku dari objek pemain dalam permainan, termasuk mekanisme untuk mengumpulkan atau menghindari objek tertentu serta mengatur skor pemain. Juga, kelas ini memiliki kemampuan untuk menambahkan efek ledakan ke dunia permainan jika objek Players menyentuh objek Bomb.

❖ Codingan

Player1

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

/**
 * Write a description of class Player1 here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Player1 extends Players
{
   /**
    * Act - do whatever the Player1 wants to do. This method is called whenever
    * the 'Act' or 'Run' button gets pressed in the environment.
    */
   public Player1()
   {
      GreenfootImage info = getImage();
      int tinggi = (int)info.getHeight()*1/6;
      int lebar = (int)info.getWidth()*1/6;

      info.scale(lebar,tinggi);
   }
   public void act()
   {
      checkKeyPress ("left", "right", "s", "w");
      lookingForCherry();
      lookingForBananas();
      lookingForBomb();
      lookingForApel();
      getScore();
     // Add your action code here.
   }
}
```

Penjelasan:
Kelas Player1, yang merupakan subkelas dari kelas Players. Kelas ini mewarisi fitur dan perilaku dari kelas Players dan menambahkan perilaku khusus untuk pemain pertama dalam permainan.
Berikut adalah penjelasannya :
1. Deklarasi Kelas Player1:
- Player1 adalah subkelas dari Players, yang berarti Player1 mewarisi atribut dan metode yang ada dalam kelas Players.
2. Konstruktor Player1():
- Konstruktor ini menyesuaikan gambar untuk Player1 dengan melakukan penyesuaian ukuran gambar (GreenfootImage) sesuai dengan 1/6 dari tinggi dan lebar gambar asli.
3. Method act():
- Metode act() dipanggil secara teratur oleh Greenfoot pada setiap siklus permainan.
- Metode ini menangani aksi yang dilakukan oleh objek Player1. Di sini, metode memanggil beberapa metode yang diwarisi dari Players seperti checkKeyPress(), lookingForCherry(), lookingForBananas(), lookingForBomb(), dan lookingForApel(). Selain itu, metode getScore() juga dipanggil, meskipun nilai yang dikembalikan tidak disimpan atau digunakan dalam kode ini.
Jadi, kelas Player1 ini mengatur perilaku khusus untuk pemain pertama dalam permainan, dengan menggunakan fitur dan perilaku yang telah didefinisikan dalam kelas induk Players dan menyesuaikan atribut gambar untuk tampilan pemain pertama dalam permainan.

❖ Codingan

Back

```
import greenfoot.*;

public class back extends Actor
{
   public back()
   {
       GreenfootImage info = getImage();
      int tinggi = (int)info.getHeight()*1/3;
      int lebar = (int)info.getWidth()*1/3;

      info.scale(lebar,tinggi);
   }

   public void act()
   {
      if(Greenfoot.mouseClicked(this))
      {
         Greenfoot.setWorld(new MyWorld());
      }
   }
}
```

Penjelasan:
Kelas back, yang merupakan subkelas dari Actor dalam lingkungan Greenfoot. Kelas ini digunakan untuk menangani tindakan saat objek back diklik dalam lingkungan permainan.
Berikut adalah penjelasannya :
1. Deklarasi Kelas back:
- back adalah kelas yang mewarisi kelas Actor dalam Greenfoot. Objek dari kelas ini dapat ditempatkan dalam dunia permainan dan memiliki perilaku tertentu.
2. Konstruktor back():
- Konstruktor ini mengatur gambar untuk objek back dengan mengubah ukuran gambar (GreenfootImage) sesuai dengan 1/3 dari tinggi dan lebar gambar asli.
3. Method act():
- Metode act() dipanggil secara teratur oleh Greenfoot pada setiap siklus permainan.
- Dalam metode ini, ada pengondisian: jika objek back diklik dengan mouse (Greenfoot.mouseClicked(this)), maka perintah Greenfoot.setWorld(new MyWorld()) akan menjalankan aksi untuk mengubah dunia permainan menjadi sebuah dunia baru yang direpresentasikan oleh kelas MyWorld.
Jadi, kelas back ini bertindak sebagai tombol atau objek yang, ketika diklik, akan mengubah dunia permainan menjadi dunia baru yang diwakili oleh kelas MyWorld. Hal ini memungkinkan pemain untuk kembali ke atau memulai ulang permainan dari awal, tergantung pada cara pengembangannya dalam kelas MyWorld.

❖ Codingan

Info

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

public class info extends Actor
{

   public info()
   {
       GreenfootImage info = getImage();
      int tinggi = (int)info.getHeight()*1/3;
      int lebar = (int)info.getWidth()*1/3;

      info.scale(lebar,tinggi);
   }
   public void act()
   {
      if(Greenfoot.mouseClicked(this))
      {
         Greenfoot.setWorld(new forminfo());
      }
   }
}
```

Penjelasan:
kelas info, yang merupakan subkelas dari Actor dalam lingkungan Greenfoot. Kelas ini bertanggung jawab untuk menangani tindakan saat objek info diklik dalam lingkungan permainan.
Berikut adalah penjelasannya :
1. Deklarasi Kelas info:
- info adalah kelas yang mewarisi kelas Actor dalam Greenfoot. Objek dari kelas ini dapat ditempatkan dalam dunia permainan dan memiliki perilaku tertentu.
2. Konstruktor info():
- Konstruktor ini mengatur gambar untuk objek info dengan mengubah ukuran gambar (GreenfootImage) sesuai dengan 1/3 dari tinggi dan lebar gambar asli.
3. Method act():
- Metode act() dipanggil secara teratur oleh Greenfoot pada setiap siklus permainan.
- Dalam metode ini, terdapat kondisi: jika objek info diklik dengan mouse (Greenfoot.mouseClicked(this)), maka perintah Greenfoot.setWorld(new forminfo()) akan menjalankan aksi untuk mengubah dunia permainan menjadi sebuah dunia baru yang direpresentasikan oleh kelas forminfo.
Jadi, kelas info ini bertindak sebagai objek dalam permainan yang, ketika diklik, akan mengubah dunia permainan menjadi dunia baru yang diwakili oleh kelas forminfo. Hal ini memungkinkan pemain untuk melihat atau mengakses informasi tambahan atau tampilan lain dalam permainan, yang dapat ditampilkan oleh kelas forminfo.

❖ Codingan

Start

```
import greenfoot.*;

public class start extends Actor
{
   public start()
   {
      GreenfootImage info = getImage();
      int tinggi = (int)info.getHeight()*1/3;
      int lebar = (int)info.getWidth()*1/3;

      info.scale(lebar,tinggi);
   }
   public void act()
   {
      if(Greenfoot.mouseClicked(this))
      {
         Greenfoot.setWorld(new mulai());
      }
   }
}
```

Penjelasan:
kelas start, yang merupakan subkelas dari Actor dalam lingkungan Greenfoot. Kelas ini bertanggung jawab untuk menangani tindakan saat objek start diklik dalam lingkungan permainan.
Berikut adalah penjelasannya :
1. Deklarasi Kelas start:
- start adalah kelas yang mewarisi kelas Actor dalam Greenfoot. Objek dari kelas ini dapat ditempatkan dalam dunia permainan dan memiliki perilaku tertentu.
2. Konstruktor start():
- Konstruktor ini mengatur gambar untuk objek start dengan mengubah ukuran gambar (GreenfootImage) sesuai dengan 1/3 dari tinggi dan lebar gambar asli.
3. Method act():
- Metode act() dipanggil secara teratur oleh Greenfoot pada setiap siklus permainan.
- Dalam metode ini, terdapat kondisi: jika objek start diklik dengan mouse (Greenfoot.mouseClicked(this)), maka perintah Greenfoot.setWorld(new mulai()) akan menjalankan aksi untuk mengubah dunia permainan menjadi sebuah dunia baru yang direpresentasikan oleh kelas mulai.
Jadi, kelas start ini bertindak sebagai objek dalam permainan yang, ketika diklik, akan memulai atau menginisialisasi permainan dengan mengubah dunia permainan menjadi dunia baru yang diwakili oleh kelas mulai. Ini memungkinkan pemain untuk memulai atau memulai ulang permainan dari awal, tergantung pada cara pengembangannya dalam kelas mulai.

❖ Codingan

Interface

```
public interface iniinterface
{
   // instance variables - replace the example below with your own
   public void lookingForCherry();
   public void lookingForApel();
   public void lookingForBananas();
   public void lookingForBomb();
}
```

Penjelasan:
sebuah interface yang dinamai iniinterface. Interface adalah sekumpulan metode yang dideklarasikan tanpa implementasi yang spesifik. Interface ini menetapkan kontrak atau kesepakatan bahwa kelas-kelas yang mengimplementasikannya akan menyediakan implementasi (kode) untuk metode-metode yang dideklarasikan di dalam interface tersebut.
Berikut adalah penjelasannya :
1. Deklarasi Interface iniinterface:
- Interface ini memiliki empat metode tanpa tubuh (body) yaitu lookingForCherry(), lookingForApel(), lookingForBananas(), dan lookingForBomb().
- Interface hanya mendeklarasikan metode-metode ini tanpa memberikan implementasi konkret. Implementasi metode-metode ini akan disediakan oleh kelas-kelas yang mengimplementasikan (menggunakan) interface ini.
2. Tujuan Interface:
- Interface digunakan sebagai kontrak yang menetapkan bahwa kelas-kelas yang mengimplementasikannya (menggunakan interface ini) harus menyediakan implementasi untuk semua metode yang dideklarasikan di dalam interface tersebut.
- Ketika sebuah kelas menggunakan interface ini (dengan menggunakan kata kunci implements), kelas tersebut harus memberikan implementasi konkret (kode) untuk semua metode yang dideklarasikan di dalam interface tersebut.
- Jadi, interface iniinterface ini memberikan panduan tentang metode-metode yang harus diberikan implementasinya oleh kelas-kelas yang menggunakannya. Setiap kelas yang menggunakan interface ini harus memberikan implementasi konkret untuk semua metode yang dideklarasikan di dalamnya. Hal ini membantu dalam menciptakan konsistensi dan fleksibilitas dalam kode, memungkinkan untuk adopsi pola desain yang lebih baik, seperti pola polimorfisme dan abstraksi dalam pemrograman berorientasi objek.

Penerapan Inheritance:
Inheritance terjadi di baris kode pertama dari kelas MyWorld, yaitu: public class MyWorld extends World.

Penerapan Polimorfisme:
Terdapat pada subclass bomb
Polimorfisme terjadi pada baris kode: world.addObject(new Duar(), getX(), getY());
Dalam kode di atas, method addObject adalah method dari kelas GreenfootWorld. Method ini menerima tiga parameter, yaitu sebuah actor, x-coordinate, dan y-coordinate. Di sini, objek new Duar() digunakan sebagai argumen pertama untuk method addObject.

Penerapan Overriding:
Terdapat Pada subclass Apel Pada baris ke delapan
'removeAtEdge()'
Terdapat overriding pada method 'removeAtEdge()' di dalam kelas Apel. Hal ini dapat dilihat dari nama method yang sama tetapi memiliki implementasi atau body code yang berbeda dengan method yang ada di kelas induknya, yaitu Actor.

Penerapan Overloading:
Terdapat pada subclass bomb
Terdapat overloading pada method removeTouching di dalam method:
lookingForCherry() pada baris ke-24 dengan parameter Cherry.class
lookingForBomb() pada baris ke-36 dengan parameter Bomb.class
lookingForBananas() pada baris ke-48 dengan parameter Bananas.class
lookingForApel() pada baris ke-60 dengan parameter Apel.class

Overloading adalah kemampuan memiliki lebih dari satu method yang memiliki nama sama tetapi memiliki jumlah atau tipe parameter yang berbeda. Di sini, masing-masing method removeTouching memiliki parameter yang berbeda sehingga menjadi contoh overloading.