

```
    } else {  
        showText("You Win!", getWidth() / 2, getHeight() / 2);  
        Greenfoot.stop();  
    }  
}  
  
public void prepare() {  
    addObject(new Character(), getRandomX(), getRandomY());  
    addObject(new ObjectSpecial(), getRandomX(), getRandomYAvoidEnemies());  
    addObject(new Enemy(), getRandomX(), getRandomYAvoidEnemies());  
}  
  
public void nextLevel() {  
    removeObjects(getObjects(ObjectSpecial.class));  
    removeObjects(getObjects(LevelComplete.class));  
    prepareNextLevel();  
}  
  
public void decreaseLives() {  
    lives--;  
    if (lives <= 0) {  
        showText("Game Over - You Lose!", getWidth() / 2, getHeight() / 2);  
        Greenfoot.stop();  
    }  
}
```

saved



```
public int getRandomX() {
    return Greenfoot.getRandomNumber(getWidth());
}

public int getRandomY() {
    return Greenfoot.getRandomNumber(getHeight());
}

public int getRandomYAvoidEnemies() {
    int newY;
    do {
        newY = getRandomY();
    } while (isTooCloseToEnemies(newY));
    return newY;
}

public boolean isTooCloseToEnemies(int newY) {
    List<Enemy> enemies = getObjects(Enemy.class);
    for (Enemy enemy : enemies) {
        if (Math.abs(newY - enemy.getY()) < 50) {
            return true; // Jarak terlalu dekat
        }
    }
    return false; // Jarak aman
}
```

saved

```
lives--;  
if (lives <= 0) {  
    showText("Game Over - You Lose!", getWidth() / 2, getHeight() / 2);  
    Greenfoot.stop();  
}  
  
public int getLives() {  
    return lives;  
}  
  
public void increaseScore() {  
    score += 10;  
    showText("Score: " + score, 50, 25);  
}  
  
public void prepareNextLevel() {  
    addObject(new ObjectSpecial(), getRandomX(), getRandomYAvoidEnemies());  
    addObject(new Enemy(), getRandomX(), getRandomYAvoidEnemies());  
}  
  
public int getRandomX() {  
    return Greenfoot.getRandomNumber(getWidth());  
}
```

```
return Greenfoot.getRandomNumber(getWidth());
}

public int getRandomY() {
    return Greenfoot.getRandomNumber(getHeight());
}

public int getRandomYAvoidEnemies() {
    int newY;
    do {
        newY = getRandomY();
    } while (isTooCloseToEnemies(newY));
    return newY;
}

public boolean isTooCloseToEnemies(int newY) {
    List<Enemy> enemies = getObjects(Enemy.class);
    for (Enemy enemy : enemies) {
        if (Math.abs(newY - enemy.getY()) < 50) {
            return true; // Jarak terlalu dekat
        }
    }
    return false; // Jarak aman
}
}
```



```
import greenfoot.*;

public class Character extends Actor {
    private GreenfootImage characterImage;
    private int speed = 5; // Kecepatan karakter
    private int lives = 3; // Jumlah nyawa karakter

    public Character() {
        characterImage = new GreenfootImage("character.png");
        setImage(characterImage);
    }

    public void act() {
        checkForCollision();
        handleKeyPress();
        addParticles(); // Panggil metode untuk menambahkan partikel
    }

    public void checkForCollision() {
        Actor object = getOneIntersectingObject(ObjectSpecial.class);
        if (object != null) {
            increaseScore();
            getWorld().removeObject(object);
        }
    }
}
```

saved

```
// Musuh menjadi diam dan mengakibatkan karakter kehilangan nyawa saat bersentuhan
Actor enemy = getOneIntersectingObject(Enemy.class);
if (enemy != null) {
    takeDamage();
}

}

public void increaseScore() {
    ((MyWorld) getWorld()).increaseScore();
}

public void handleKeyPress() {
    if (Greenfoot.isKeyDown("left")) {
        setLocation(getX() - speed, getY());
        setRotation(180); // Menghadap ke kiri
    }
    if (Greenfoot.isKeyDown("right")) {
        setLocation(getX() + speed, getY());
        setRotation(0); // Menghadap ke kanan
    }
    if (Greenfoot.isKeyDown("up")) {
        setLocation(getX(), getY() - speed);
        setRotation(270); // Menghadap ke atas
    }
}
```

```
        setLocation(getX() + speed, getY());
        setRotation(0); // Menghadap ke kanan
    }
    if (Greenfoot.isKeyDown("up")) {
        setLocation(getX(), getY() - speed);
        setRotation(270); // Menghadap ke atas
    }
    if (Greenfoot.isKeyDown("down")) {
        setLocation(getX(), getY() + speed);
        setRotation(90); // Menghadap ke bawah
    }
}

public void addParticles() {
    // Tambahkan partikel saat karakter bergerak
    if (Greenfoot.isKeyDown("left") || Greenfoot.isKeyDown("right") || Greenfoot.isKeyDown("up") || Greenfoot.isKeyDown("down")) {
        World world = getWorld();
        if (world != null) {
            int particleX = getX() + Greenfoot.getRandomNumber(20) - 10;
            int particleY = getY() + Greenfoot.getRandomNumber(20) - 10;
            world.addObject(new Particle(), particleX, particleY);
        }
    }
}
```

saved


```
public void addParticles() {
    // Tambahkan partikel saat karakter bergerak
    if (Greenfoot.isKeyDown("left") || Greenfoot.isKeyDown("right") || Greenfoot.isKeyDown("up") || Greenfoot.isKeyDown("down")) {
        World world = getWorld();
        if (world != null) {
            int particleX = getX() + Greenfoot.getRandomNumber(20) - 10;
            int particleY = getY() + Greenfoot.getRandomNumber(20) - 10;
            world.addObject(new Particle(), particleX, particleY);
        }
    }
}

// Metode untuk mengurangi nyawa karakter
public void takeDamage() {
    lives--;
    if (lives <= 0) {
        World world = getWorld();
        if (world != null) {
            world.showText("Game Over - You Lose!", world.getWidth() / 2, world.getHeight() / 2);
            Greenfoot.stop();
        }
    }
}
```



```
import greenfoot.*;

public class Enemy extends Actor {
    private int moveDistance = 5; // Jarak pergerakan musuh
    private int moveDirection = 1; // 1 berarti bergerak ke kanan, -1 berarti bergerak ke kiri

    public Enemy() {
        setImage("enemy.png");
    }

    public void act() {
        moveVertically();
        moveHorizontally();
        checkEdge();
    }

    public void moveVertically() {
        setLocation(getX() + (moveDirection * moveDistance), getY());
    }

    public void moveHorizontally() {
        setLocation(getX() + (moveDirection * moveDistance), getY());
    }

    public void checkEdge() {
        if (getY() == 0) {
            moveDirection = -1;
        }
        if (getY() == 400) {
            moveDirection = 1;
        }
    }
}
```

saved

```
public Enemy() {  
    setImage("enemy.png");  
}  
  
public void act() {  
    moveVertically();  
    moveHorizontally();  
    checkEdge();  
}  
  
public void moveVertically() {  
    setLocation(getX() + (moveDirection * moveDistance), getY());  
}  
  
public void moveHorizontally() {  
    setLocation(getX() + (moveDirection * moveDistance), getY());  
}  
  
public void checkEdge() {  
    if (isAtEdge()) {  
        // Jika musuh mencapai tepi layar, balik arahnya  
        moveDirection *= -1;  
    }  
}  
}
```

saved

```
|import greenfoot.*;
```

```
public class LevelComplete extends Actor {
```

```
    public LevelComplete(int level) {
```

```
        setImage(new GreenfootImage("Level " + level + " Complete", 36, Color.WHITE, null));
```

```
    }
```

```
    public void act() {
```

```
        getWorld().removeObject(this);
```

```
    }
```

```
import greenfoot.*;

public class ObjectSpecial extends Actor {
    private GreenfootSound collectSound = new GreenfootSound("collect.mp3"); // Ganti dengan nama file suara Anda

    public ObjectSpecial() {
        setImage("object.png");
    }

    public void act() {
        checkForCollision();
    }

    public void checkForCollision() {
        Actor character = getOneIntersectingObject(Character.class);
        if (character != null) {
            ((Character) character).increaseScore();
            getWorld().removeObject(this);
            playCollectSound(); // Memainkan efek suara
        }
    }

    public void playCollectSound() {
        collectSound.play();
    }
}
```



```
import greenfoot.*;
```

```
public class Particle extends Actor {
```

```
    public Particle() {
```

```
        GreenfootImage particleImage = new GreenfootImage(5, 5);
```

```
        particleImage.setColor(Color.YELLOW); // Ubah warna partikel sesuai keinginan
```

```
        particleImage.fillOval(0, 0, 5, 5);
```

```
        setImage(particleImage);
```

```
    }
```

```
    public void act() {
```

```
        if (getImage().getTransparency() > 5) {
```

```
            getImage().setTransparency(getImage().getTransparency() - 5);
```

```
        } else {
```

```
            getWorld().removeObject(this);
```

```
        }
```

```
    }
```

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

public class MyWorld extends World {
    private int currentLevel = 1;
    private int score = 0;
    private int lives = 3;
    private int requiredSpecialObjects = 10;

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

    public void act() {
        if (getObjects(ObjectSpecial.class).isEmpty()) {
            if (currentLevel < 3) {
                currentLevel++;
                requiredSpecialObjects += 5;
                addObject(new LevelComplete(currentLevel), getWidth() / 2, getHeight() / 2);
                Greenfoot.delay(100);
                nextLevel();
            } else {
                showText("You Win!", getWidth() / 2, getHeight() / 2);
                Greenfoot.stop();
            }
        }
    }
}
```

saved

