Programação Orientada por Objetos - Projeto 1

Introdução

No âmbito desta cadeira foi pedido um desenvolvimento de um jogo usando a plataforma Greenfoot, e conceitos de Programação Orientada por Objetos. O tema do jogo baseia-se nas alterações climáticas que se têm vindo a desenrolar nas últimas décadas, uma problemática cada vez mais pertinente a nível mundial. Com este tema de fundo, foi desenvolvido um jogo para dois jogadores que colaboram entre si.

Tema

Visto que o urso polar é um animal que tem sido bastante afetado pelos problemas causados pelo aquecimento global (degelo, falta de alimento, etc) decidimos fazer o projeto à volta desta temática. Dois ursos polares que se tentam alimentar, limpar o lixo do seu habitat, e fugir do aquecimento global, representado pelas chamas.

Objetivo

Conseguir o maior score possível antes de ficar sem vidas. O jogo é baseado no Pacman, partilhando vários dos seus conceitos.

Funcionamento

Jogadores

```
if (!wasInitialized) {
  // Sets the respawning positio
  is initialized (the position i
  spawnX = getX();
  spawnY = getY();

  // Player image
  if (player1MovementKeys != nul
    setImage("be_1.png");
    getImage().scale(17, 12);
  } else {
    setImage("ad_1.png");
    getImage().scale(17, 12);
  }

  wasInitialized = true;
}
```

Quando os jogadores executam o seu primeiro "act", o lugar onde eles estão é gravado, para no caso de eles morrerem, voltarem a essa posição. A sua imagem é também mudada, para a imagem correspondente a cada jogador.

Os jogadores 1 e 2 são distinguidos pelo parâmetro que

lhes é passado como argumento (char[] ou String[]) cumprindo o requisito de overloading.

Visto que o jogo é cooperativo, foi decidido manter as vidas partilhadas. Para cumprir o requisito de cada jogador ter algo que lhe é único, foi decidido que o jogador que apanhar o *power up* é o jogador que ficará com esse power up, verificável por imitir um som a rugir, e começar a piscar até o power up acabar. Quando o jogador está com o power up ativado, pode comer os inimigos e ganhar pontos.

```
private void blink() {···

private void powerUp() {···
```

Os ursos estão também animados de forma a cumprir outro requisito (algo que pode ser díficil de verificar num ecrã pequeno, devido à natureza do jogo necessitar o mesmo estar dividido em "tiles", e teletransportar o urso entre cada "tile").

O movimento dos jogadores foi programado de forma a ser contínuo, sendo apenas necessário um toque na tecla da direção em que ser quer mover. Foi programado também de forma a que os ursos parem quando colidirem com uma parede (classe Wall), parando também a sua animação.

```
// Movement
protected void setIsMoving(boolean newIsMoving) {···

private void bearMovementAnimation(···

private void move(char[] keyboardKeys) {···

private void move(String[] keyboardKeys) {···
```

Inimigos

Tal como os jogadores quando os inimigos executam o seu primeiro "act", o lugar onde eles estão é gravado, para no caso de eles morrerem, voltarem a essa posição. Após o countdown acabar (countdown este que será discutido à frente), os inimigos saem da caixa onde se encontram. A localização desta caixa varia de nível para nível. Os inimigos saem também da caixa se forem comidos por um player.

```
if (getWorld() instanceof Level1) {
  Level1 level1 = (Level1) getWorld();
  level1.leaveTheBox(this);
} else if (getWorld() instanceof Level2) {
  Level2 level2 = (Level2) getWorld();
  level2.leaveTheBox(this);
}
```

Os inimigos usam uma variedade de subclasses para se movimentar (Interception, Corner, TurnBack) da classe Actor, encontradas invísiveis em cada nível para se deslocar. Quando chegam a cada uma destas subclasses é executado o método correspondente. No caso de chegar a uma

interceção, é escolhida aleatóriamente uma direção para o inimigo se virar.

```
// Movement
protected void setMovementDirection(int newMovementDirection) {···

private void TurnBack() {···

private void moveWhenInCorner() {···

private void moveWhenInIntersection() {···

private void move() {···
```

Se os inimigos atingirem um jogador que não está com o power up ativado o jogo pára durante 1 segundo. Se o jogador não tiver o power up ativado e ficar sem vidas, então o jogo acaba, com a opção de voltar ao menu, ou reiniciar o nível.

Visto que a deteção de colisão com os jogadores não é feita por imagem, mas sim por coordenadas, esta função tem que ser chamada antes e depois de cada movimentação.

```
hitPlayers(playersHit);
move();
hitPlayers(playersHit);
```

Nível

Visto que os níveis partilham muita da suas respetivas implementações, foi criada a classe Level (classe pai de cada nível).

Quando um nível é inicializado, é executado um *countdown*, durante o qual nem os jogadores nem os inimigos se podem mover.

É mostrada a vida e o score dos jogadores em cada nível.

Pode pressionar a tecla ESC para voltar ao menu principal.

Quando todos os coins são comidos, estes voltam a ser repostos.

Nível 1

O nível 1 tem umas paredes que realizam o teletransporte dos inimigos e jogadores.

```
private void enemiesTeleportWalls() {···

private void playersTeleportWalls() {···
```

Nível 2

O nível 1 tem uns cantos sem fim, que obrigam os inimigos a voltar atrás.

Menu principal

No menu principal pode escolher ver as instruções, sair do jogo, ou jogar. Ao clicar no jogar, poderá escolher o nível que pretende jogar.

Overriding

Quando é que se utiliza overriding? O overriding é utilizado quando se tem uma classe pai que partilha um método com a maioria das suas subclasses, mas pretende-se ter um comportamento diferente numa minoria destas subclasse.

Exemplo:

Classe Ave - Método voar()

Subclasses – Papagaio, Pardal, Tucano, Galinha.

Todas as subclasses herdam o método voar da classe Ave. Visto que todas voam, menos a galinha, podemos fazer um overriding no método voar da classe galinha.

Algo do género:

```
Public class Galinha(){

@override

Voar(){tentaVoarECai();}

}
```

Devido à estrutura do projeto, não se justifica a implementação de um "override" para um comportamento correto, e fácil compreensão do programa. No entanto, esperamos que a justificação, e futuras perguntas na apresentação, sejam suficientes para não haver uma descida da nota, e demonstrar que tal conceito foi mais do que percebido.

Contudo, visto que todas as classes em Java herdam da classe objeto, e a classe objeto contém o método toString(), foi implementado um overriding do mesmo na subclasse Wall, de maneira a servir de exemplo de como seria implementado.

Conclusão

Em conclusão, o projeto não só foi concluído de forma a cumprir todos os requisitos, mas também teve o cuidado de integrar cada um deles de forma a respeitar o tema dado, de fazer sentido na sua integração, de facilitar a leitura do código fonte, e de permitir futuras mudanças (como por exemplo a implementação de novos níveis com novos comportamentos) da maneira mais fácil e rápida possível.

Para tais futuras mudanças serem implementadas facilmente, o encapsulamento dos vários métodos e variáveis foi também ele cuidadosamente pensado.

O código foi todo ele formatado usando o formatador de Java da Red Had, e a sua leitura é aconselhada num IDE que permite o colapso das várias classes e métodos nele presentes, como por exemplo o VsCode.

Todo o código, sem exceção, foi programado por nós.

Anexo

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


public class BigDot extends Actor {


  private long blinkingStartingTime = System.currentTimeMillis();


  public BigDot() {
    GreenfootImage image = getImage();
    image.scale(15, 15);
    setImage(image);
  }


  private void blink() {
    // Blinking


    long currentTime = System.currentTimeMillis();
```

```
      // 200 ms Opaque
      if (currentTime - blinkingStartingTime < 250) {
        getImage().setTransparency(255);
      }
      // 200 ms Transparent
      else if (
        (currentTime - blinkingStartingTime) > 250 &&
        (currentTime - blinkingStartingTime) < 500
      ) {
        getImage().setTransparency(0);
      }
      // Resets the cycle
      else {
        blinkingStartingTime = System.currentTimeMillis();
      }
    }

    private void hitPlayer() {
      List<Player> playerHit = getWorld()
        .getObjectsAt(getX(), getY(), Player.class);

      if (playerHit.isEmpty() == false) {
        Level level = getWorldOfType(Level.class);
        level.setScore(level.getScore() + 50);
        level.setNumberOfCoinsAte(level.getNumberOfCoinsAte() +
1);

        GreenfootSound roar = new GreenfootSound("roar.mp3");
        roar.play();
```

```
            // Power up the player that hit the big dot. If two
players hit it at the same time, only powers up the first in the
list.
            playerHit.get(0).setIsPoweredUp(true);
            playerHit
              .get(0)
              .setPoweredUpBlinkingStartingTime(System.currentTimeMil
lis());

playerHit.get(0).setPoweredUpStartingTime(System.currentTimeMillis()
);

            getWorld().removeObject(this);
          }
        }

        public void act() {
          blink();
          hitPlayer();
        }
      }
      import greenfoot.*;  // (World, Actor, GreenfootImage,
Greenfoot and MouseInfo)


      public class ButtonBackToMenu extends Actor
      {
          GreenfootSound backgroundMusic;


          public ButtonBackToMenu(GreenfootSound backgroundMusic){

              getImage().scale(225, 75);
          this.backgroundMusic = backgroundMusic;
```

```
        }

        public ButtonBackToMenu(){
           getImage().scale(225, 75);

        }



        public void act()
        {


            if (Greenfoot.mouseMoved(this))
getImage().setTransparency(100);
            if (Greenfoot.mouseMoved(null) && !
Greenfoot.mouseMoved(this)) getImage().setTransparency(255);



             if (Greenfoot.mouseClicked(this))
             {

                 if ( backgroundMusic != null){
                   backgroundMusic.stop();
                 }
                   Greenfoot.setWorld(new MainMenu());


           }
        }
    }
```

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



     public class ButtonExit extends Actor

     {


         public void act()

         {

            if (Greenfoot.mouseMoved(this))
getImage().setTransparency(100);

            if (Greenfoot.mouseMoved(null) && !
Greenfoot.mouseMoved(this)) getImage().setTransparency(255);




             if (Greenfoot.mouseClicked(this))

             {

                 System.exit(0);

         }

     }

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



     public class ButtonInstructions extends Actor

     {


         public void act()

         {
```

```
            if (Greenfoot.mouseMoved(this))
getImage().setTransparency(100);
            if (Greenfoot.mouseMoved(null) && !
Greenfoot.mouseMoved(this)) getImage().setTransparency(255);




             if (Greenfoot.mouseClicked(this))
             {
                 Greenfoot.setWorld(new Instructions());
         }
     }
     }
     import greenfoot.*;  // (World, Actor, GreenfootImage,
Greenfoot and MouseInfo)

     public class ButtonPlay extends Actor
     {

         public void act()
         {
            if (Greenfoot.mouseMoved(this))
getImage().setTransparency(100);
            if (Greenfoot.mouseMoved(null) && !
Greenfoot.mouseMoved(this)) getImage().setTransparency(255);




             if (Greenfoot.mouseClicked(this))
             {

                 Greenfoot.setWorld(new StageChooser());

         }
```

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



    public class ButtonRestartLevel extends Actor
    {

        GreenfootSound backgroundMusic;




        public ButtonRestartLevel(GreenfootSound backgroundMusic){

        this.backgroundMusic = backgroundMusic;
        getImage().scale(225, 75);

        }



        public void act()
        {
           if (Greenfoot.mouseMoved(this))
getImage().setTransparency(100);
           if (Greenfoot.mouseMoved(null) && !
Greenfoot.mouseMoved(this)) getImage().setTransparency(255);




            if (Greenfoot.mouseClicked(this))
            {
```

```
                    backgroundMusic.stop();


                        if (getWorld() instanceof Level1) {
      Greenfoot.setWorld(new Level1());
            } else if (getWorld() instanceof Level2) {
      Greenfoot.setWorld(new Level2());
            }



            }
          }
      }

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


      public class ButtonStage1 extends Actor
      {

          ButtonStage1(){

                      getImage().scale(200, 200);
          }


          public void act()
          {
              if (Greenfoot.mouseMoved(this))
getImage().setTransparency(100);
              if (Greenfoot.mouseMoved(null) && !
Greenfoot.mouseMoved(this)) getImage().setTransparency(255);
```

```
            if (Greenfoot.mouseClicked(this))
            {
                Greenfoot.setWorld(new Level1());
        }
    }
    }
    import greenfoot.*;  // (World, Actor, GreenfootImage,
Greenfoot and MouseInfo)

    public class ButtonStage2 extends Actor
    {

        public ButtonStage2(){

                getImage().scale(200, 200);
        }

        public void act()
        {
           if (Greenfoot.mouseMoved(this))
getImage().setTransparency(100);
            if (Greenfoot.mouseMoved(null) && !
Greenfoot.mouseMoved(this)) getImage().setTransparency(255);



            if (Greenfoot.mouseClicked(this))
            {
                Greenfoot.setWorld(new Level2());
        }
    }
    }
    import greenfoot.*;  // (World, Actor, GreenfootImage,
Greenfoot and MouseInfo)
```

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

        public Corner() {

        GreenfootImage image = getImage();
        image.scale(15, 15);
        setImage(image);
        image.setTransparency(0);



    }

    public void act()
    {
        // Add your action code here.
    }
}
import greenfoot.*;
import java.util.ArrayList;
import java.util.List;

public class Enemy extends Actor {

  private int movementDirection; // 0 - Up, 1 - Right, 2 -
Down, 3 - Left
```

```
        // Spawn
        private int spawnX;
        private int spawnY;

        private boolean wasInitialized = false;

        public Enemy(int movementDirection) {
          this.movementDirection = movementDirection;

          GreenfootImage image = getImage();
          image.scale(15, 15);
          setImage(image);
        }

        // Movement
        protected void setMovementDirection(int newMovementDirection)
{
          movementDirection = newMovementDirection;
        }

        private void TurnBack() {
          if (movementDirection == 0) {
            movementDirection = 2;
          } else if (movementDirection == 1) {
            movementDirection = 3;
          } else if (movementDirection == 2) {
            movementDirection = 0;
          } else if (movementDirection == 3) {
            movementDirection = 1;
          }
        }
```

```
        private void moveWhenInCorner() {
          // Moving up when arriving at a corner, can't go up, can't
go back down.
          if (movementDirection == 0) {
            // Checks left
            if (
              (getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class)).isEmpty() ==
              false
            ) {
              movementDirection = 1;
            }
            // Checks right
            else if (
              (getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class)).isEmpty() ==
              false
            ) {
              movementDirection = 3;
            }
          }
          // Moving right when arriving at a corner, can't go right,
can't go back left
          else if (movementDirection == 1) {
            // Checks up
            if (
              (getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class)).isEmpty() ==
              false
            ) {
              movementDirection = 2;
            }
            // Checks down
            else if (
```

```
            (getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class)).isEmpty() ==

            false

          ) {

            movementDirection = 0;

          }

        }

        // Moving down when arriving at a corner, can't go down,
can't go back up

        else if (movementDirection == 2) {

          // Checks left

          if (

            (getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class)).isEmpty() ==

            false

          ) {

            movementDirection = 1;

          }

          // Checks right

          else if (

            (getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class)).isEmpty() ==

            false

          ) {

            movementDirection = 3;

          }

        }

        // Moving left when arriving at a corner, can't go left,
can't go back right

        else if (movementDirection == 3) {

          // Checks up

          if (

            (getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class)).isEmpty() ==

            false
```

```
            ) {
              movementDirection = 2;
            }
            // Checks down
            else if (
              (getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class)).isEmpty() ==
              false
            ) {
              movementDirection = 0;
            }
          }
        }

        private void moveWhenInIntersection() {
          // Wall only above. Can go right, down and left
          if (
            !getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class).isEmpty() && // Wall above
            getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class).isEmpty() && // No wall right
            getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class).isEmpty() && // No wall below
            getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class).isEmpty() // No wall left
          ) {
            // System.out.println("Wall only above");

            int randomNumber = Greenfoot.getRandomNumber(3);

            if (randomNumber == 0) {
              movementDirection = 1;
            } else if (randomNumber == 1) {
              movementDirection = 2;
```

```
            } else if (randomNumber == 2) {
              movementDirection = 3;
            }
          }
          // Wall only left. Can go up, right and down
          else if (
            getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class).isEmpty() && // No wall above
            getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class).isEmpty() && // No wall right
            getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class).isEmpty() && // No wall below
            !getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class).isEmpty() // Wall left
          ) {
            //System.out.println("Wall only left");

            int randomNumber = Greenfoot.getRandomNumber(3);

            if (randomNumber == 0) {
              movementDirection = 0;
            } else if (randomNumber == 1) {
              movementDirection = 1;
            } else if (randomNumber == 2) {
              movementDirection = 2;
            }
          }
          // No walls in any direction
          else if (
            getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class).isEmpty() && // No wall above
            getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class).isEmpty() && // No wall right
            getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class).isEmpty() && // No wall below
```

```
            getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class).isEmpty() // No wall left
          ) {
            //System.out.println("No walls in any direction");
            movementDirection = Greenfoot.getRandomNumber(4);
          }
          // Wall only below. Can go up, right and left
          else if (
            getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class).isEmpty() && // No wall above
            getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class).isEmpty() && // No wall right
            !getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class).isEmpty() && // Wall below
            getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class).isEmpty() // No wall left
          ) {
            //System.out.println("Wall only below");

            int randomNumber = Greenfoot.getRandomNumber(3);

            if (randomNumber == 0) {
              movementDirection = 0;
            } else if (randomNumber == 1) {
              movementDirection = 1;
            } else if (randomNumber == 2) {
              movementDirection = 3;
            }
          }
          // Wall only right. Can go up, right and down
          else if (
            getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class).isEmpty() && // No wall above
            !getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class).isEmpty() && // Wall right
```

```
            getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class).isEmpty() && // No wall below
            getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class).isEmpty() // No wall left
          ) {
            //System.out.println("Wall only to the right");

            int randomNumber = Greenfoot.getRandomNumber(3);

            if (randomNumber == 0) {
              movementDirection = 0;
            } else if (randomNumber == 1) {
              movementDirection = 2;
            } else if (randomNumber == 2) {
              movementDirection = 3;
            }
          }
        }

        private void move() {
          // Moving up
          if (movementDirection == 0) {
            setLocation(getX(), getY() - 1);
          }
          // Moving to the right
          else if (movementDirection == 1) {
            setLocation(getX() + 1, getY());
            // Moving down
          } else if (movementDirection == 2) {
            setLocation(getX(), getY() + 1);
            // Moving to the left
          } else if (movementDirection == 3) {
            setLocation(getX() - 1, getY());
```

```
          }


          // Reaches corner
          if (
            (getWorld().getObjectsAt(getX(), getY(),
Corner.class)).isEmpty() == false
          ) {
            moveWhenInCorner();
          }
          // Reaches intersection
          else if (
            (getWorld().getObjectsAt(getX(), getY(),
Intersection.class)).isEmpty() ==
            false
          ) {
            moveWhenInIntersection();
          }
          // Reaches turn back
          else if (
            (getWorld().getObjectsAt(getX(), getY(),
TurnBack.class)).isEmpty() ==
            false
          ) {
            TurnBack();
          }
        }


        // Hit detection


        private void gameOver() {
          getWorldOfType(Level.class).setGameOver(true);
          getWorld()
            .showText(
```

```
            "Game Over! Score: " +
            String.valueOf(getWorldOfType(Level.class).getScore())
+ ".",
            getWorld().getWidth() / 2,
            getWorld().getHeight() / 2
          );


                if (getWorld() instanceof Level1) {
                   Level1 level1 = (Level1) getWorld();


                   // Restart level button
          ButtonRestartLevel buttonRestartLevel = new
ButtonRestartLevel(level1.getBackgroundMusic());

getWorldOfType(Level.class).addObject(buttonRestartLevel, 10, 22);


                // Back to menu button
                ButtonBackToMenu buttonBackToMenu = new
ButtonBackToMenu(level1.getBackgroundMusic());

getWorldOfType(Level.class).addObject(buttonBackToMenu, 26, 22);


          } else if (getWorld() instanceof Level2) {
            Level2 level2 = (Level2) getWorld();
                           // Restart level button
          ButtonRestartLevel buttonRestartLevel = new
ButtonRestartLevel(level2.getBackgroundMusic());

getWorldOfType(Level.class).addObject(buttonRestartLevel, 10, 22);
                // Back to menu button
                ButtonBackToMenu buttonBackToMenu = new
ButtonBackToMenu(level2.getBackgroundMusic());

getWorldOfType(Level.class).addObject(buttonBackToMenu, 26, 22);
```

```
            }

        }

        private void stopTime(int milliseconds) {
          long startingTime = System.currentTimeMillis();

          while ((System.currentTimeMillis() - startingTime) <
milliseconds) {}
        }

        private List<Player> hitPlayers(List<Player>
playersAlreadyHit) {
          List<Player> playersHit = getWorld()
            .getObjectsAt(getX(), getY(), Player.class);

          // Hit one or more players
          if (playersHit.isEmpty() == false) {
            GreenfootSound roar = new GreenfootSound("roar.mp3");
            roar.play();

            // Players hit
            for (int i = 0; i < playersHit.size(); i++) {
              Player tempPlayerHit = playersHit.get(i);

              boolean playerAlreadyHit = false;

              // Players already hit
              for (int y = 0; y < playersAlreadyHit.size(); y++) {
```

```
                Player tempPlayerAlreadyHit =
playersAlreadyHit.get(y);

                if (tempPlayerHit == tempPlayerAlreadyHit) {
                  playerAlreadyHit = true;
                }
              }

              // If the player was already hit in a single act, don't
hit it again
              if (!playerAlreadyHit) {
                // The player isn't powered up
                if (!tempPlayerHit.isPoweredUp()) {
                  stopTime(1000);
                  getWorldOfType(Level.class)
                    .setPlayerHealth(
                      getWorldOfType(Level.class).getPlayerHealth() -
1
                    );

                  tempPlayerHit.setLocation(
                    tempPlayerHit.getSpawnX(),
                    tempPlayerHit.getSpawnY()
                  );

                  tempPlayerHit.setIsMoving(false);

                  if (getWorldOfType(Level.class).getPlayerHealth() <
1) {
                    gameOver();
                  }
                }
                // The player is powered up
                else if (tempPlayerHit.isPoweredUp()) {
```

```
                stopTime(500);

                getWorldOfType(Level.class)
                  .setScore(getWorldOfType(Level.class).getScore()
+ 200);

                // Moves the enemy back to the box
                setLocation(spawnX, spawnY);
              }
            }
          }
        }

        return playersHit;
      }

      public void act() {
        // Individual enemy initialization
        if (!wasInitialized) {
          // Sets the respawning position. The respawning position
is set equal to the position that the enemy actor is initialized
(the position in the first act).
          spawnX = getX();
          spawnY = getY();

          wasInitialized = true;
        }

        // After the countdown ends, if the game isn't over
        if (
          getWorldOfType(Level.class).wasCountdownAlreadyShown() &&
          (!getWorldOfType(Level.class).getGameOver())
        ) {
```

```
            List<Player> playersHit = new ArrayList<Player>();
            playersHit = hitPlayers(playersHit);

            if (getWorld() instanceof Level1) {
              Level1 level1 = (Level1) getWorld();
              level1.leaveTheBox(this);
            } else if (getWorld() instanceof Level2) {
              Level2 level2 = (Level2) getWorld();
              level2.leaveTheBox(this);
            }

            hitPlayers(playersHit);
            move();
            hitPlayers(playersHit);

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

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

          /**
           * Constructor for objects of class Instructions.
```

```
     *
     */
    public Instructions()
    {
        // Create a new world with 600x400 cells with a cell
size of 1x1 pixels.
        super(600, 400, 1);



        setBackground(new
GreenfootImage("instrucoes_background.png"));
        ButtonBackToMenu backmenuButton = new
ButtonBackToMenu();
        addObject(backmenuButton, 480, 355);
    }
}
import greenfoot.*;  // (World, Actor, GreenfootImage,
Greenfoot and MouseInfo)


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


        public Intersection() {


        GreenfootImage image = getImage();
        image.scale(15, 15);
        setImage(image);
        image.setTransparency(0);
```

```
    }

    public void act()
    {
        // Add your action code here.
    }
}
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)
import java.util.List;

public class Level extends World {

  private int score = 0;
  private int playerHealth = 3;
  private boolean countdownAlreadyShown = false; // Starts the movement after the coundown is shown
  private long startingTime; // The time at which the level was initialized
  private boolean gameOver = false; // Stops the movement after the game is over
  private int numberOfCoinsAte = 0; // Used to insert new coins after eating them all
  private int numberOfCoinsInLevel; // The total amount of coins in each level

  private Player[] players = new Player[2];
  private Enemy[] enemies = new Enemy[6];

  public Level(long startingTime) {
    super(36, 36, 17);
    this.startingTime = startingTime;
  }
```

```
// Actors
protected Player[] getPlayers() {
  return players;
}

protected void addPlayer(Player newPlayer, int i) {
  players[i] = newPlayer;
}

protected Enemy[] getEnemies() {
  return enemies;
}

protected void addEnemy(Enemy newEnemy, int i) {
  enemies[i] = newEnemy;
}

// Number of Coins
protected int getNumberOfCoinsAte() {
  return numberOfCoinsAte;
}

protected void setNumberOfCoinsAte(int newNumberOfCoinsAte) {
  numberOfCoinsAte = newNumberOfCoinsAte;
}

protected int getNumberOfCoinsInLevel() {
  return numberOfCoinsInLevel;
}
```

```
        protected void setNumberOfCoinsInLevel(int
newNumberOfCoinsInLevel) {
          numberOfCoinsInLevel = newNumberOfCoinsInLevel;
        }

        // Score
        protected int getScore() {
          return score;
        }

        protected void setScore(int newScore) {
          score = newScore;
        }

        protected void showScore(Level level) {
          showText("Score: " + String.valueOf(score), 2 *
getWidth() / 3, 1);
        }

        // Player Health
        protected int getPlayerHealth() {
          return playerHealth;
        }

        protected void setPlayerHealth(int newPlayerHealth) {
          playerHealth = newPlayerHealth;
        }

        protected void showHealthBar(Level level) {
          // Player Health
          level.showText("Health: " + String.valueOf(playerHealth),
getWidth() / 3, 1);
```

```
        }

        // Countdown
        protected boolean wasCountdownAlreadyShown() {
          return countdownAlreadyShown;
        }

        protected void setCountdownAlreadyShown(boolean
newCountdownAlreadyShown) {
          countdownAlreadyShown = newCountdownAlreadyShown;
        }

        protected void countdown(Level level) {
          // Starts a 3 second countdown

          if (!countdownAlreadyShown) {
            long currentTime = System.currentTimeMillis();

            if (
              (currentTime - startingTime) > 1000 &&
              (currentTime - startingTime) < 2000
            ) {
              level.showText("Get ready: 3", 18, 17);
            } else if (
              (currentTime - startingTime) > 2000 &&
              (currentTime - startingTime) < 3000
            ) {
              level.showText("Get ready: 2", 18, 17);
            } else if (
              (currentTime - startingTime) > 3000 &&
              (currentTime - startingTime) < 4000
            ) {
```

```
            level.showText("Get ready: 1", 18, 17);
          } else if ((currentTime - startingTime) > 4000) {
            countdownAlreadyShown = true;
            level.showText("", 18, 17);
          }
        }
      }

      // Game Over
      protected boolean getGameOver() {
        return gameOver;
      }

      protected void setGameOver(boolean newGameOver) {
        gameOver = newGameOver;
      }

      // Return to menu
      protected void returnToMenu(GreenfootSound backgroundMusic) {
        if (Greenfoot.isKeyDown("escape")) {
          backgroundMusic.stop();
          Greenfoot.setWorld(new MainMenu());
        }
      }


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

    public class Level1 extends Level {
```

```
    private GreenfootSound backgroundMusic = new GreenfootSound(
      "backgroundMusic.mp3"
    );

    public Level1() {
      super(System.currentTimeMillis());
      // Background
      setBackground("background.png");

      // Number of coins in level
      setNumberOfCoinsInLevel(244);

      // Players Initialization
      char[] player1MovementKeys = new char[] { 'W', 'D', 'S',
'A' };
      addPlayer(new Player(player1MovementKeys), 0);
      addObject(getPlayers()[0], 17, 26);

      String[] player2MovementKeys = new String[] {
        "up",
        "right",
        "down",
        "left",
      };
      addPlayer(new Player(player2MovementKeys), 1);
      addObject(getPlayers()[1], 18, 26);

      // Actors
      addWalls();
      addSmallDots();
      addBigDots();
      addIntersections();
      addCorners();
```

```
    // Game speed
    Greenfoot.setSpeed(35);

    // Game music
    backgroundMusic.setVolume(30);
    backgroundMusic.playLoop();

    // Enemies
    addEnemy(new Enemy(0), 0);
    addObject(getEnemies()[0], 17, 16);

    addEnemy(new Enemy(0), 1);
    addObject(getEnemies()[1], 18, 16);

    addEnemy(new Enemy(0), 2);
    addObject(getEnemies()[2], 17, 17);

    addEnemy(new Enemy(0), 3);
    addObject(getEnemies()[3], 18, 17);

    addEnemy(new Enemy(0), 4);
    addObject(getEnemies()[4], 17, 18);

    addEnemy(new Enemy(0), 5);
    addObject(getEnemies()[5], 18, 18);
  }

  // Background Music
  protected GreenfootSound getBackgroundMusic() {
    return backgroundMusic;
  }
```

```
        // Enemy movement
        private void addIntersections() {
          // Adds the intersection actors to the world (Used for the
enemies movement)

          Intersection inters1 = new Intersection();
          addObject(inters1, 10, 4);
          Intersection inters2 = new Intersection();
          addObject(inters2, 25, 4);
          Intersection inters3 = new Intersection();
          addObject(inters3, 5, 8);
          Intersection inters4 = new Intersection();
          addObject(inters4, 10, 8);
          Intersection inters5 = new Intersection();
          addObject(inters5, 13, 8);
          Intersection inters6 = new Intersection();
          addObject(inters6, 16, 8);
          Intersection inters7 = new Intersection();
          addObject(inters7, 19, 8);
          Intersection inters8 = new Intersection();
          addObject(inters8, 22, 8);
          Intersection inters9 = new Intersection();
          addObject(inters9, 25, 8);
          Intersection inters10 = new Intersection();
          addObject(inters10, 30, 8);
          Intersection inters11 = new Intersection();
          addObject(inters11, 10, 11);
          Intersection inters12 = new Intersection();
          addObject(inters12, 25, 11);
          Intersection inters13 = new Intersection();
          addObject(inters13, 16, 14);
          Intersection inters14 = new Intersection();
```

```
addObject(inters14, 19, 14);
Intersection inters15 = new Intersection();
addObject(inters15, 10, 17);
Intersection inters16 = new Intersection();
addObject(inters16, 13, 17);
Intersection inters17 = new Intersection();
addObject(inters17, 22, 17);
Intersection inters18 = new Intersection();
addObject(inters18, 25, 17);
Intersection inters19 = new Intersection();
addObject(inters19, 13, 20);
Intersection inters20 = new Intersection();
addObject(inters20, 22, 20);
Intersection inters21 = new Intersection();
addObject(inters21, 10, 23);
Intersection inters22 = new Intersection();
addObject(inters22, 10, 23);
Intersection inters23 = new Intersection();
addObject(inters23, 13, 23);
Intersection inters24 = new Intersection();
addObject(inters24, 22, 23);
Intersection inters25 = new Intersection();
addObject(inters25, 25, 23);
Intersection inters26 = new Intersection();
addObject(inters26, 10, 26);
Intersection inters27 = new Intersection();
addObject(inters27, 13, 26);
Intersection inters28 = new Intersection();
addObject(inters28, 16, 26);
Intersection inters29 = new Intersection();
addObject(inters29, 19, 26);
Intersection inters30 = new Intersection();
addObject(inters30, 22, 26);
```

```
        Intersection inters31 = new Intersection();
        addObject(inters31, 25, 26);
        Intersection inters32 = new Intersection();
        addObject(inters32, 7, 29);
        Intersection inters33 = new Intersection();
        addObject(inters33, 28, 29);
        Intersection inters34 = new Intersection();
        addObject(inters34, 16, 32);
        Intersection inters35 = new Intersection();
        addObject(inters35, 19, 32);
    }

    private void addCorners() {
        // Adds the corner actors to the world (Used for the
enemies movement)

        Corner corner1 = new Corner();
        addObject(corner1, 5, 4);
        Corner corner2 = new Corner();
        addObject(corner2, 16, 4);
        Corner corner3 = new Corner();
        addObject(corner3, 19, 4);
        Corner corner4 = new Corner();
        addObject(corner4, 30, 4);
        Corner corner5 = new Corner();
        addObject(corner5, 5, 11);
        Corner corner6 = new Corner();
        addObject(corner6, 13, 11);
        Corner corner7 = new Corner();
        addObject(corner7, 16, 11);
        Corner corner8 = new Corner();
        addObject(corner8, 19, 11);
        Corner corner9 = new Corner();
```

```
addObject(corner9, 22, 11);
Corner corner10 = new Corner();
addObject(corner10, 30, 11);
Corner corner11 = new Corner();
addObject(corner11, 13, 14);
Corner corner12 = new Corner();
addObject(corner12, 22, 14);
Corner corner13 = new Corner();
addObject(corner13, 5, 23);
Corner corner14 = new Corner();
addObject(corner14, 16, 23);
Corner corner15 = new Corner();
addObject(corner15, 19, 23);
Corner corner16 = new Corner();
addObject(corner16, 30, 23);
Corner corner17 = new Corner();
addObject(corner17, 5, 26);
Corner corner18 = new Corner();
addObject(corner18, 7, 26);
Corner corner19 = new Corner();
addObject(corner19, 28, 26);
Corner corner20 = new Corner();
addObject(corner20, 30, 26);
Corner corner21 = new Corner();
addObject(corner21, 5, 29);
Corner corner22 = new Corner();
addObject(corner22, 5, 29);
Corner corner23 = new Corner();
addObject(corner23, 10, 29);
Corner corner24 = new Corner();
addObject(corner24, 13, 29);
Corner corner25 = new Corner();
addObject(corner25, 16, 29);
```

```
        Corner corner26 = new Corner();
        addObject(corner26, 19, 29);
        Corner corner27 = new Corner();
        addObject(corner27, 22, 29);
        Corner corner28 = new Corner();
        addObject(corner28, 25, 29);
        Corner corner29 = new Corner();
        addObject(corner29, 30, 29);
        Corner corner30 = new Corner();
        addObject(corner30, 5, 32);
        Corner corner31 = new Corner();
        addObject(corner31, 30, 32);
      }

      protected void leaveTheBox(Enemy enemy) {
        // The enemies leave the spawning box at the beginning of a
level, and after they're eaten by a powered up player

        // Y = 18
        if ((enemy.getX() == 17) && (enemy.getY() == 18)) {
          enemy.setMovementDirection(0);
        } else if ((enemy.getX() == 18) && (enemy.getY() == 18)) {
          enemy.setMovementDirection(0);
        }
        // Y = 17
        else if ((enemy.getX() == 17) && (enemy.getY() == 17)) {
          enemy.setMovementDirection(0);
        } else if ((enemy.getX() == 18) && (enemy.getY() == 17)) {
          enemy.setMovementDirection(0);
        }
        // Y = 16
        else if ((enemy.getX() == 17) && (enemy.getY() == 16)) {
```

```
      enemy.setMovementDirection(0);
    } else if ((enemy.getX() == 18) && (enemy.getY() == 16)) {
      enemy.setMovementDirection(0);
    }
    // Y = 15 (Door)
    // Leaves the box and goes left or right randomly
    else if ((enemy.getX() == 17) && (enemy.getY() == 15)) {
      enemy.setLocation(enemy.getX(), enemy.getY() - 1);

      if (Greenfoot.getRandomNumber(2) == 0) {
        enemy.setMovementDirection(1);
      } else {
        enemy.setMovementDirection(3);
      }
    } else if ((enemy.getX() == 18) && (enemy.getY() == 15)) {
      enemy.setLocation(enemy.getX(), enemy.getY() - 1);
      if (Greenfoot.getRandomNumber(2) == 0) {
        enemy.setMovementDirection(1);
      } else {
        enemy.setMovementDirection(3);
      }
    }
  }

  // World building
  private void addSmallDots() {
    // Adds the small dot actors to the world

    SmallDot smallDot1 = new SmallDot();
    addObject(smallDot1, 19, 4);
    SmallDot smallDot2 = new SmallDot();
    addObject(smallDot2, 20, 4);
```

```
SmallDot smallDot3 = new SmallDot();
addObject(smallDot3, 21, 4);
SmallDot smallDot4 = new SmallDot();
addObject(smallDot4, 22, 4);
SmallDot smallDot5 = new SmallDot();
addObject(smallDot5, 23, 4);
SmallDot smallDot6 = new SmallDot();
addObject(smallDot6, 24, 4);
SmallDot smallDot7 = new SmallDot();
addObject(smallDot7, 25, 4);
SmallDot smallDot8 = new SmallDot();
addObject(smallDot8, 26, 4);
SmallDot smallDot9 = new SmallDot();
addObject(smallDot9, 27, 4);
SmallDot smallDot10 = new SmallDot();
addObject(smallDot10, 28, 4);
SmallDot smallDot11 = new SmallDot();
addObject(smallDot11, 29, 4);
SmallDot smallDot12 = new SmallDot();
addObject(smallDot12, 30, 4);
SmallDot smallDot13 = new SmallDot();
addObject(smallDot13, 19, 5);
SmallDot smallDot14 = new SmallDot();
addObject(smallDot14, 25, 5);
SmallDot smallDot15 = new SmallDot();
addObject(smallDot15, 30, 5);
SmallDot smallDot16 = new SmallDot();
addObject(smallDot16, 19, 6);
SmallDot smallDot17 = new SmallDot();
addObject(smallDot17, 25, 6);
SmallDot smallDot19 = new SmallDot();
addObject(smallDot19, 19, 7);
SmallDot smallDot20 = new SmallDot();
```

```
addObject(smallDot20, 25, 7);
SmallDot smallDot21 = new SmallDot();
addObject(smallDot21, 30, 7);
SmallDot smallDot22 = new SmallDot();
addObject(smallDot22, 18, 8);
SmallDot smallDot23 = new SmallDot();
addObject(smallDot23, 19, 8);
SmallDot smallDot24 = new SmallDot();
addObject(smallDot24, 20, 8);
SmallDot smallDot25 = new SmallDot();
addObject(smallDot25, 21, 8);
SmallDot smallDot26 = new SmallDot();
addObject(smallDot26, 22, 8);
SmallDot smallDot27 = new SmallDot();
addObject(smallDot27, 23, 8);
SmallDot smallDot28 = new SmallDot();
addObject(smallDot28, 24, 8);
SmallDot smallDot35 = new SmallDot();
addObject(smallDot35, 25, 8);
SmallDot smallDot36 = new SmallDot();
addObject(smallDot36, 26, 8);
SmallDot smallDot37 = new SmallDot();
addObject(smallDot37, 27, 8);
SmallDot smallDot38 = new SmallDot();
addObject(smallDot38, 28, 8);
SmallDot smallDot39 = new SmallDot();
addObject(smallDot39, 29, 8);
SmallDot smallDot40 = new SmallDot();
addObject(smallDot40, 30, 8);

//smallDots top left
```

```
SmallDot smallDot41 = new SmallDot();
addObject(smallDot41, 5, 4);
SmallDot smallDot42 = new SmallDot();
addObject(smallDot42, 6, 4);
SmallDot smallDot43 = new SmallDot();
addObject(smallDot43, 7, 4);
SmallDot smallDot44 = new SmallDot();
addObject(smallDot44, 8, 4);
SmallDot smallDot45 = new SmallDot();
addObject(smallDot45, 9, 4);
SmallDot smallDot46 = new SmallDot();
addObject(smallDot46, 10, 4);
SmallDot smallDot47 = new SmallDot();
addObject(smallDot47, 11, 4);
SmallDot smallDot48 = new SmallDot();
addObject(smallDot48, 12, 4);
SmallDot smallDot49 = new SmallDot();
addObject(smallDot49, 13, 4);
SmallDot smallDot50 = new SmallDot();
addObject(smallDot50, 14, 4);
SmallDot smallDot51 = new SmallDot();
addObject(smallDot51, 15, 4);
SmallDot smallDot52 = new SmallDot();
addObject(smallDot52, 16, 4);
SmallDot smallDot53 = new SmallDot();
addObject(smallDot53, 5, 5);
SmallDot smallDot54 = new SmallDot();
addObject(smallDot54, 10, 5);
SmallDot smallDot55 = new SmallDot();
addObject(smallDot55, 16, 5);
SmallDot smallDot57 = new SmallDot();
addObject(smallDot57, 10, 6);
SmallDot smallDot58 = new SmallDot();
```

```
addObject(smallDot58, 16, 6);
SmallDot smallDot59 = new SmallDot();
addObject(smallDot59, 5, 7);
SmallDot smallDot60 = new SmallDot();
addObject(smallDot60, 10, 7);
SmallDot smallDot61 = new SmallDot();
addObject(smallDot61, 16, 7);
SmallDot smallDot62 = new SmallDot();
addObject(smallDot62, 5, 8);
SmallDot smallDot63 = new SmallDot();
addObject(smallDot63, 6, 8);
SmallDot smallDot64 = new SmallDot();
addObject(smallDot64, 7, 8);
SmallDot smallDot65 = new SmallDot();
addObject(smallDot65, 8, 8);
SmallDot smallDot66 = new SmallDot();
addObject(smallDot66, 9, 8);
SmallDot smallDot67 = new SmallDot();
addObject(smallDot67, 10, 8);
SmallDot smallDot68 = new SmallDot();
addObject(smallDot68, 11, 8);
SmallDot smallDot69 = new SmallDot();
addObject(smallDot69, 12, 8);
SmallDot smallDot70 = new SmallDot();
addObject(smallDot70, 13, 8);
SmallDot smallDot71 = new SmallDot();
addObject(smallDot71, 14, 8);
SmallDot smallDot72 = new SmallDot();
addObject(smallDot72, 15, 8);
SmallDot smallDot73 = new SmallDot();
addObject(smallDot73, 16, 8);
SmallDot smallDot74 = new SmallDot();
addObject(smallDot74, 17, 8);
```

```
// smallDots middle

SmallDot smallDot75 = new SmallDot();
addObject(smallDot75, 5, 9);
SmallDot smallDot76 = new SmallDot();
addObject(smallDot76, 10, 9);
SmallDot smallDot77 = new SmallDot();
addObject(smallDot77, 13, 9);
SmallDot smallDot78 = new SmallDot();
addObject(smallDot78, 22, 9);
SmallDot smallDot79 = new SmallDot();
addObject(smallDot79, 25, 9);
SmallDot smallDot80 = new SmallDot();
addObject(smallDot80, 30, 9);
SmallDot smallDot81 = new SmallDot();
addObject(smallDot81, 5, 10);
SmallDot smallDot82 = new SmallDot();
addObject(smallDot82, 10, 10);
SmallDot smallDot83 = new SmallDot();
addObject(smallDot83, 13, 10);
SmallDot smallDot84 = new SmallDot();
addObject(smallDot84, 22, 10);
SmallDot smallDot85 = new SmallDot();
addObject(smallDot85, 25, 10);
SmallDot smallDot86 = new SmallDot();
addObject(smallDot86, 30, 10);
SmallDot smallDot87 = new SmallDot();
addObject(smallDot87, 5, 11);
SmallDot smallDot88 = new SmallDot();
addObject(smallDot88, 6, 11);
SmallDot smallDot89 = new SmallDot();
```

```
addObject(smallDot89, 7, 11);
SmallDot smallDot90 = new SmallDot();
addObject(smallDot90, 8, 11);
SmallDot smallDot91 = new SmallDot();
addObject(smallDot91, 9, 11);
SmallDot smallDot92 = new SmallDot();
addObject(smallDot92, 10, 11);
SmallDot smallDot93 = new SmallDot();
addObject(smallDot93, 13, 11);
SmallDot smallDot94 = new SmallDot();
addObject(smallDot94, 14, 11);
SmallDot smallDot95 = new SmallDot();
addObject(smallDot95, 15, 11);
SmallDot smallDot96 = new SmallDot();
addObject(smallDot96, 16, 11);
SmallDot smallDot97 = new SmallDot();
addObject(smallDot97, 19, 11);
SmallDot smallDot98 = new SmallDot();
addObject(smallDot98, 20, 11);
SmallDot smallDot99 = new SmallDot();
addObject(smallDot99, 21, 11);
SmallDot smallDot100 = new SmallDot();
addObject(smallDot100, 22, 11);
SmallDot smallDot101 = new SmallDot();
addObject(smallDot101, 25, 11);
SmallDot smallDot102 = new SmallDot();
addObject(smallDot102, 26, 11);
SmallDot smallDot103 = new SmallDot();
addObject(smallDot103, 27, 11);
SmallDot smallDot104 = new SmallDot();
addObject(smallDot104, 28, 11);
SmallDot smallDot105 = new SmallDot();
addObject(smallDot105, 29, 11);
```

```
SmallDot smallDot106 = new SmallDot();
addObject(smallDot106, 30, 11);
SmallDot smallDot107 = new SmallDot();
addObject(smallDot107, 10, 12);
//SmallDot smallDot108 = new SmallDot();
//addObject(smallDot108, 16, 12);
//SmallDot smallDot109 = new SmallDot();
//addObject(smallDot109, 19, 12);
SmallDot smallDot110 = new SmallDot();
addObject(smallDot110, 25, 12);
SmallDot smallDot111 = new SmallDot();
addObject(smallDot111, 10, 13);
//SmallDot smallDot112 = new SmallDot();
//addObject(smallDot112, 16, 13);
//SmallDot smallDot113 = new SmallDot();
//addObject(smallDot113, 19, 13);
SmallDot smallDot114 = new SmallDot();
addObject(smallDot114, 25, 13);
SmallDot smallDot115 = new SmallDot();
addObject(smallDot115, 10, 14);
//SmallDot smallDot116 = new SmallDot();
//addObject(smallDot116, 13, 14);
//SmallDot smallDot117 = new SmallDot();
//addObject(smallDot117, 14, 14);
//SmallDot smallDot118 = new SmallDot();
//addObject(smallDot118, 15, 14);
//SmallDot smallDot120 = new SmallDot();
//addObject(smallDot120, 16, 14);
//SmallDot smallDot121 = new SmallDot();
//addObject(smallDot121, 17, 14);
//SmallDot smallDot122 = new SmallDot();
//addObject(smallDot122, 18, 14);
//SmallDot smallDot123 = new SmallDot();
```

```
//addObject(smallDot123, 19, 14);
//SmallDot smallDot124 = new SmallDot();
//addObject(smallDot124, 20, 14);
//SmallDot smallDot125 = new SmallDot();
//addObject(smallDot125, 21, 14);
//SmallDot smallDot126 = new SmallDot();
//addObject(smallDot126, 22, 14);
SmallDot smallDot127 = new SmallDot();
addObject(smallDot127, 25, 14);
SmallDot smallDot128 = new SmallDot();
addObject(smallDot128, 10, 15);
//SmallDot smallDot129 = new SmallDot();
//addObject(smallDot129, 13, 15);
//SmallDot smallDot130 = new SmallDot();
//addObject(smallDot130, 22, 15);
SmallDot smallDot131 = new SmallDot();
addObject(smallDot131, 25, 15);
SmallDot smallDot132 = new SmallDot();
addObject(smallDot132, 10, 16);
//SmallDot smallDot133 = new SmallDot();
//addObject(smallDot133, 13, 16);
//SmallDot smallDot134 = new SmallDot();
//addObject(smallDot134, 22, 16);
SmallDot smallDot135 = new SmallDot();
addObject(smallDot135, 25, 16);
SmallDot smallDot136 = new SmallDot();
addObject(smallDot136, 10, 17);
//SmallDot smallDot137 = new SmallDot();
//addObject(smallDot137, 11, 17);
//SmallDot smallDot138 = new SmallDot();
//addObject(smallDot138, 12, 17);
//SmallDot smallDot139 = new SmallDot();
//addObject(smallDot139, 13, 17);
```

```
//SmallDot smallDot140 = new SmallDot();
//addObject(smallDot140, 22, 17);
//SmallDot smallDot141 = new SmallDot();
//addObject(smallDot141, 23, 17);
//SmallDot smallDot142 = new SmallDot();
//addObject(smallDot142, 24, 17);
SmallDot smallDot143 = new SmallDot();
addObject(smallDot143, 25, 17);
SmallDot smallDot144 = new SmallDot();
addObject(smallDot144, 10, 18);
//SmallDot smallDot145 = new SmallDot();
//addObject(smallDot145, 13, 18);
//SmallDot smallDot146 = new SmallDot();
//addObject(smallDot146, 22, 18);
SmallDot smallDot147 = new SmallDot();
addObject(smallDot147, 25, 18);
SmallDot smallDot148 = new SmallDot();
addObject(smallDot148, 10, 19);
//SmallDot smallDot149 = new SmallDot();
//addObject(smallDot149, 13, 19);
//SmallDot smallDot150 = new SmallDot();
//addObject(smallDot150, 22, 19);
SmallDot smallDot151 = new SmallDot();
addObject(smallDot151, 25, 19);
SmallDot smallDot152 = new SmallDot();
addObject(smallDot152, 10, 20);
//SmallDot smallDot153 = new SmallDot();
//addObject(smallDot153, 13, 20);
//SmallDot smallDot154 = new SmallDot();
//addObject(smallDot154, 14, 20);
//SmallDot smallDot155 = new SmallDot();
//addObject(smallDot155, 15, 20);
//SmallDot smallDot156 = new SmallDot();
```

```
//addObject(smallDot156, 16, 20);
//SmallDot smallDot157 = new SmallDot();
//addObject(smallDot157, 17, 20);
//SmallDot smallDot158 = new SmallDot();
//addObject(smallDot158, 18, 20);
//SmallDot smallDot159 = new SmallDot();
//addObject(smallDot159, 19, 20);
//SmallDot smallDot160 = new SmallDot();
//addObject(smallDot160, 20, 20);
//SmallDot smallDot161 = new SmallDot();
//addObject(smallDot161, 21, 20);
//SmallDot smallDot162 = new SmallDot();
//addObject(smallDot162, 22, 20);
SmallDot smallDot163 = new SmallDot();
addObject(smallDot163, 25, 20);
SmallDot smallDot164 = new SmallDot();
addObject(smallDot164, 10, 21);
//SmallDot smallDot165 = new SmallDot();
//addObject(smallDot165, 13, 21);
//SmallDot smallDot166 = new SmallDot();
//addObject(smallDot166, 22, 21);
SmallDot smallDot167 = new SmallDot();
addObject(smallDot167, 25, 21);
SmallDot smallDot168 = new SmallDot();
addObject(smallDot168, 10, 22);
//SmallDot smallDot169 = new SmallDot();
//addObject(smallDot169, 13, 22);
//SmallDot smallDot170 = new SmallDot();
//addObject(smallDot170, 22, 22);
SmallDot smallDot171 = new SmallDot();
addObject(smallDot171, 25, 22);
```

```
//smallDots down up

SmallDot smallDot172 = new SmallDot();
addObject(smallDot172, 5, 23);
SmallDot smallDot173 = new SmallDot();
addObject(smallDot173, 6, 23);
SmallDot smallDot174 = new SmallDot();
addObject(smallDot174, 7, 23);
SmallDot smallDot175 = new SmallDot();
addObject(smallDot175, 8, 23);
SmallDot smallDot176 = new SmallDot();
addObject(smallDot176, 9, 23);
SmallDot smallDot177 = new SmallDot();
addObject(smallDot177, 10, 23);
SmallDot smallDot178 = new SmallDot();
addObject(smallDot178, 11, 23);
SmallDot smallDot179 = new SmallDot();
addObject(smallDot179, 12, 23);
SmallDot smallDot180 = new SmallDot();
addObject(smallDot180, 13, 23);
SmallDot smallDot181 = new SmallDot();
addObject(smallDot181, 14, 23);
SmallDot smallDot182 = new SmallDot();
addObject(smallDot182, 15, 23);
SmallDot smallDot183 = new SmallDot();
addObject(smallDot183, 16, 23);
SmallDot smallDot184 = new SmallDot();
addObject(smallDot184, 19, 23);
SmallDot smallDot185 = new SmallDot();
addObject(smallDot185, 20, 23);
SmallDot smallDot186 = new SmallDot();
addObject(smallDot186, 21, 23);
```

```
SmallDot smallDot187 = new SmallDot();
addObject(smallDot187, 22, 23);
SmallDot smallDot188 = new SmallDot();
addObject(smallDot188, 23, 23);
SmallDot smallDot189 = new SmallDot();
addObject(smallDot189, 24, 23);
SmallDot smallDot190 = new SmallDot();
addObject(smallDot190, 25, 23);
SmallDot smallDot191 = new SmallDot();
addObject(smallDot191, 26, 23);
SmallDot smallDot192 = new SmallDot();
addObject(smallDot192, 27, 23);
SmallDot smallDot193 = new SmallDot();
addObject(smallDot193, 28, 23);
SmallDot smallDot194 = new SmallDot();
addObject(smallDot194, 29, 23);
SmallDot smallDot195 = new SmallDot();
addObject(smallDot195, 30, 23);
SmallDot smallDot196 = new SmallDot();
addObject(smallDot196, 5, 24);
SmallDot smallDot197 = new SmallDot();
addObject(smallDot197, 10, 24);
SmallDot smallDot198 = new SmallDot();
addObject(smallDot198, 16, 24);
SmallDot smallDot199 = new SmallDot();
addObject(smallDot199, 19, 24);
SmallDot smallDot200 = new SmallDot();
addObject(smallDot200, 25, 24);
SmallDot smallDot201 = new SmallDot();
addObject(smallDot201, 30, 24);
SmallDot smallDot202 = new SmallDot();
addObject(smallDot202, 5, 25);
SmallDot smallDot203 = new SmallDot();
```

```
addObject(smallDot203, 10, 25);
SmallDot smallDot204 = new SmallDot();
addObject(smallDot204, 16, 25);
SmallDot smallDot205 = new SmallDot();
addObject(smallDot205, 19, 25);
SmallDot smallDot206 = new SmallDot();
addObject(smallDot206, 25, 25);
SmallDot smallDot207 = new SmallDot();
addObject(smallDot207, 30, 25);
SmallDot smallDot209 = new SmallDot();
addObject(smallDot209, 6, 26);
SmallDot smallDot210 = new SmallDot();
addObject(smallDot210, 7, 26);
SmallDot smallDot211 = new SmallDot();
addObject(smallDot211, 10, 26);
SmallDot smallDot212 = new SmallDot();
addObject(smallDot212, 11, 26);
SmallDot smallDot213 = new SmallDot();
addObject(smallDot213, 12, 26);
SmallDot smallDot214 = new SmallDot();
addObject(smallDot214, 13, 26);
SmallDot smallDot215 = new SmallDot();
addObject(smallDot215, 14, 26);
SmallDot smallDot216 = new SmallDot();
addObject(smallDot216, 15, 26);
SmallDot smallDot217 = new SmallDot();
addObject(smallDot217, 16, 26);
SmallDot smallDot220 = new SmallDot();
addObject(smallDot220, 19, 26);
SmallDot smallDot221 = new SmallDot();
addObject(smallDot221, 20, 26);
SmallDot smallDot222 = new SmallDot();
addObject(smallDot222, 21, 26);
```

```
SmallDot smallDot223 = new SmallDot();
addObject(smallDot223, 22, 26);
SmallDot smallDot224 = new SmallDot();
addObject(smallDot224, 23, 26);
SmallDot smallDot225 = new SmallDot();
addObject(smallDot225, 24, 26);
SmallDot smallDot226 = new SmallDot();
addObject(smallDot226, 25, 26);
SmallDot smallDot229 = new SmallDot();
addObject(smallDot229, 28, 26);
SmallDot smallDot230 = new SmallDot();
addObject(smallDot230, 29, 26);
SmallDot smallDot233 = new SmallDot();
addObject(smallDot233, 7, 27);
SmallDot smallDot234 = new SmallDot();
addObject(smallDot234, 10, 27);
SmallDot smallDot235 = new SmallDot();
addObject(smallDot235, 13, 27);
SmallDot smallDot236 = new SmallDot();
addObject(smallDot236, 22, 27);
SmallDot smallDot237 = new SmallDot();
addObject(smallDot237, 25, 27);
SmallDot smallDot238 = new SmallDot();
addObject(smallDot238, 28, 27);

// smallDots down down

SmallDot smallDot239 = new SmallDot();
addObject(smallDot239, 7, 28);
SmallDot smallDot240 = new SmallDot();
addObject(smallDot240, 10, 28);
SmallDot smallDot241 = new SmallDot();
```

```
addObject(smallDot241, 13, 28);
SmallDot smallDot242 = new SmallDot();
addObject(smallDot242, 22, 28);
SmallDot smallDot243 = new SmallDot();
addObject(smallDot243, 25, 28);
SmallDot smallDot244 = new SmallDot();
addObject(smallDot244, 28, 28);
SmallDot smallDot245 = new SmallDot();
addObject(smallDot245, 5, 29);
SmallDot smallDot246 = new SmallDot();
addObject(smallDot246, 6, 29);
SmallDot smallDot247 = new SmallDot();
addObject(smallDot247, 7, 29);
SmallDot smallDot248 = new SmallDot();
addObject(smallDot248, 8, 29);
SmallDot smallDot249 = new SmallDot();
addObject(smallDot249, 9, 29);
SmallDot smallDot250 = new SmallDot();
addObject(smallDot250, 10, 29);
SmallDot smallDot251 = new SmallDot();
addObject(smallDot251, 13, 29);
SmallDot smallDot252 = new SmallDot();
addObject(smallDot252, 14, 29);
SmallDot smallDot253 = new SmallDot();
addObject(smallDot253, 15, 29);
SmallDot smallDot254 = new SmallDot();
addObject(smallDot254, 16, 29);
SmallDot smallDot255 = new SmallDot();
addObject(smallDot255, 19, 29);
SmallDot smallDot256 = new SmallDot();
addObject(smallDot256, 20, 29);
SmallDot smallDot257 = new SmallDot();
addObject(smallDot257, 21, 29);
```

```
SmallDot smallDot258 = new SmallDot();
addObject(smallDot258, 22, 29);
SmallDot smallDot259 = new SmallDot();
addObject(smallDot259, 25, 29);
SmallDot smallDot260 = new SmallDot();
addObject(smallDot260, 26, 29);
SmallDot smallDot261 = new SmallDot();
addObject(smallDot261, 27, 29);
SmallDot smallDot262 = new SmallDot();
addObject(smallDot262, 28, 29);
SmallDot smallDot263 = new SmallDot();
addObject(smallDot263, 29, 29);
SmallDot smallDot264 = new SmallDot();
addObject(smallDot264, 30, 29);
SmallDot smallDot265 = new SmallDot();
addObject(smallDot265, 5, 30);
SmallDot smallDot266 = new SmallDot();
addObject(smallDot266, 16, 30);
SmallDot smallDot267 = new SmallDot();
addObject(smallDot267, 19, 30);
SmallDot smallDot268 = new SmallDot();
addObject(smallDot268, 30, 30);
SmallDot smallDot269 = new SmallDot();
addObject(smallDot269, 5, 31);
SmallDot smallDot270 = new SmallDot();
addObject(smallDot270, 16, 31);
SmallDot smallDot271 = new SmallDot();
addObject(smallDot271, 19, 31);
SmallDot smallDot272 = new SmallDot();
addObject(smallDot272, 30, 31);
SmallDot smallDot273 = new SmallDot();
addObject(smallDot273, 5, 32);
SmallDot smallDot274 = new SmallDot();
```

```
addObject(smallDot274, 6, 32);
SmallDot smallDot275 = new SmallDot();
addObject(smallDot275, 7, 32);
SmallDot smallDot276 = new SmallDot();
addObject(smallDot276, 8, 32);
SmallDot smallDot277 = new SmallDot();
addObject(smallDot277, 9, 32);
SmallDot smallDot278 = new SmallDot();
addObject(smallDot278, 10, 32);
SmallDot smallDot279 = new SmallDot();
addObject(smallDot279, 11, 32);
SmallDot smallDot280 = new SmallDot();
addObject(smallDot280, 12, 32);
SmallDot smallDot281 = new SmallDot();
addObject(smallDot281, 13, 32);
SmallDot smallDot282 = new SmallDot();
addObject(smallDot282, 14, 32);
SmallDot smallDot283 = new SmallDot();
addObject(smallDot283, 15, 32);
SmallDot smallDot284 = new SmallDot();
addObject(smallDot284, 16, 32);
SmallDot smallDot285 = new SmallDot();
addObject(smallDot285, 17, 32);
SmallDot smallDot286 = new SmallDot();
addObject(smallDot286, 18, 32);
SmallDot smallDot287 = new SmallDot();
addObject(smallDot287, 19, 32);
SmallDot smallDot288 = new SmallDot();
addObject(smallDot288, 20, 32);
SmallDot smallDot289 = new SmallDot();
addObject(smallDot289, 21, 32);
SmallDot smallDot290 = new SmallDot();
addObject(smallDot290, 22, 32);
```

```
    SmallDot smallDot291 = new SmallDot();
    addObject(smallDot291, 23, 32);
    SmallDot smallDot292 = new SmallDot();
    addObject(smallDot292, 24, 32);
    SmallDot smallDot293 = new SmallDot();
    addObject(smallDot293, 25, 32);
    SmallDot smallDot294 = new SmallDot();
    addObject(smallDot294, 26, 32);
    SmallDot smallDot295 = new SmallDot();
    addObject(smallDot295, 27, 32);
    SmallDot smallDot296 = new SmallDot();
    addObject(smallDot296, 28, 32);
    SmallDot smallDot297 = new SmallDot();
    addObject(smallDot297, 29, 32);
    SmallDot smallDot298 = new SmallDot();
    addObject(smallDot298, 30, 32);
  }

  private void addBigDots() {
    // Adds the big dot actors to the world

    BigDot BigDot1 = new BigDot();
    addObject(BigDot1, 30, 6);
    BigDot BigDot2 = new BigDot();
    addObject(BigDot2, 5, 6);
    BigDot BigDot3 = new BigDot();
    addObject(BigDot3, 30, 26);
    BigDot BigDot4 = new BigDot();
    addObject(BigDot4, 5, 26);
  }

  private void addWalls() {
```

```
// Adds the level walls to the world

// Upper/middle walls (Square 1)
Wall wallq1 = new Wall();
addObject(wallq1, 6, 5);
wallq1.setLocation(6, 5);
Wall wallq11 = new Wall();
addObject(wallq11, 7, 5);
wallq11.setLocation(7, 5);
Wall wallq12 = new Wall();
addObject(wallq12, 8, 5);
wallq12.setLocation(8, 5);
Wall wallq13 = new Wall();
addObject(wallq13, 9, 5);
wallq13.setLocation(9, 5);
Wall wallq14 = new Wall();
addObject(wallq14, 9, 6);
wallq14.setLocation(9, 6);
Wall wallq15 = new Wall();
addObject(wallq15, 9, 7);
wallq15.setLocation(9, 7);
Wall wallq16 = new Wall();
addObject(wallq16, 8, 7);
wallq16.setLocation(8, 7);
Wall wallq17 = new Wall();
addObject(wallq17, 7, 7);
wallq17.setLocation(7, 7);
Wall wallq18 = new Wall();
addObject(wallq18, 6, 7);
wallq18.setLocation(6, 7);
Wall wallq19 = new Wall();
addObject(wallq19, 6, 6);
```

```
wallq19.setLocation(6, 6);

// Upper/middle walls (Square 2)
Wall wallq2 = new Wall();
addObject(wallq2, 11, 5);
wallq2.setLocation(11, 5);
Wall wallq21 = new Wall();
addObject(wallq21, 12, 5);
wallq21.setLocation(12, 5);
Wall wallq22 = new Wall();
addObject(wallq22, 13, 5);
wallq22.setLocation(13, 5);
Wall wallq23 = new Wall();
addObject(wallq23, 14, 5);
wallq23.setLocation(14, 5);
Wall wallq24 = new Wall();
addObject(wallq24, 15, 5);
wallq24.setLocation(15, 5);
Wall wallq25 = new Wall();
addObject(wallq25, 15, 6);
wallq25.setLocation(15, 6);
Wall wallq26 = new Wall();
addObject(wallq26, 15, 7);
wallq26.setLocation(15, 7);
Wall wallq27 = new Wall();
addObject(wallq27, 14, 7);
wallq27.setLocation(14, 7);
Wall wallq28 = new Wall();
addObject(wallq28, 13, 7);
wallq28.setLocation(13, 7);
Wall wallq29 = new Wall();
addObject(wallq29, 12, 7);
```

```
wallq29.setLocation(12, 7);
Wall wallq2q1 = new Wall();
addObject(wallq2q1, 11, 7);
wallq2q1.setLocation(11, 7);
Wall wallq2q2 = new Wall();
addObject(wallq2q2, 11, 6);
wallq2q2.setLocation(11, 6);

// Upper/middle walls (Square 3)
Wall wallq3 = new Wall();
addObject(wallq3, 20, 5);
wallq3.setLocation(20, 5);
Wall wallq31 = new Wall();
addObject(wallq31, 21, 5);
wallq31.setLocation(21, 5);
Wall wallq32 = new Wall();
addObject(wallq32, 22, 5);
wallq32.setLocation(22, 5);
Wall wallq33 = new Wall();
addObject(wallq33, 23, 5);
wallq33.setLocation(23, 5);
Wall wallq34 = new Wall();
addObject(wallq34, 24, 5);
wallq34.setLocation(24, 5);
Wall wallq35 = new Wall();
addObject(wallq35, 24, 6);
wallq35.setLocation(24, 6);
Wall wallq36 = new Wall();
addObject(wallq36, 24, 7);
wallq36.setLocation(24, 7);
Wall wallq37 = new Wall();
addObject(wallq37, 23, 7);
```

```
wallq37.setLocation(23, 7);
Wall wallq38 = new Wall();
addObject(wallq38, 22, 7);
wallq38.setLocation(22, 7);
Wall wallq39 = new Wall();
addObject(wallq39, 21, 7);
wallq39.setLocation(21, 7);
Wall wallq3q1 = new Wall();
addObject(wallq3q1, 20, 7);
wallq3q1.setLocation(20, 7);
Wall wallq3q2 = new Wall();
addObject(wallq3q2, 20, 6);
wallq3q2.setLocation(20, 6);

// Upper/middle walls (Square 4)
Wall wallq4 = new Wall();
addObject(wallq4, 26, 5);
wallq4.setLocation(26, 5);
Wall wallq41 = new Wall();
addObject(wallq41, 27, 5);
wallq41.setLocation(27, 5);
Wall wallq42 = new Wall();
addObject(wallq42, 28, 5);
wallq42.setLocation(28, 5);
Wall wallq43 = new Wall();
addObject(wallq43, 29, 5);
wallq43.setLocation(29, 5);
Wall wallq44 = new Wall();
addObject(wallq44, 29, 6);
wallq44.setLocation(29, 6);
Wall wallq45 = new Wall();
addObject(wallq45, 29, 7);
```

```
wallq45.setLocation(29, 7);
Wall wallq46 = new Wall();
addObject(wallq46, 28, 7);
wallq46.setLocation(28, 7);
Wall wallq47 = new Wall();
addObject(wallq47, 27, 7);
wallq47.setLocation(27, 7);
Wall wallq48 = new Wall();
addObject(wallq48, 26, 7);
wallq48.setLocation(26, 7);
Wall wallq49 = new Wall();
addObject(wallq49, 26, 6);
wallq49.setLocation(26, 6);

// Middle walls (Square 5)
Wall wallq5 = new Wall();
addObject(wallq5, 6, 9);
wallq5.setLocation(6, 9);
Wall wallq51 = new Wall();
addObject(wallq51, 7, 9);
wallq51.setLocation(7, 9);
Wall wallq52 = new Wall();
addObject(wallq52, 8, 9);
wallq52.setLocation(8, 9);
Wall wallq53 = new Wall();
addObject(wallq53, 9, 9);
wallq53.setLocation(9, 9);
Wall wallq54 = new Wall();
addObject(wallq54, 9, 10);
wallq54.setLocation(9, 10);
Wall wallq55 = new Wall();
addObject(wallq55, 8, 10);
```

```
wallq55.setLocation(8, 10);
Wall wallq56 = new Wall();
addObject(wallq56, 7, 10);
wallq56.setLocation(7, 10);
Wall wallq57 = new Wall();
addObject(wallq57, 6, 10);
wallq57.setLocation(6, 10);

// Middle walls (Square 6)
Wall wallq6 = new Wall();
addObject(wallq6, 11, 9);
wallq6.setLocation(11, 9);
Wall wallq61 = new Wall();
addObject(wallq61, 12, 9);
wallq61.setLocation(12, 9);
Wall wallq62 = new Wall();
addObject(wallq62, 12, 10);
wallq62.setLocation(12, 10);
Wall wallq63 = new Wall();
addObject(wallq63, 12, 11);
wallq63.setLocation(12, 11);
Wall wallq64 = new Wall();
addObject(wallq64, 12, 12);
wallq64.setLocation(12, 12);
Wall wallq65 = new Wall();
addObject(wallq65, 13, 12);
wallq65.setLocation(13, 12);
Wall wallq66 = new Wall();
addObject(wallq66, 14, 12);
wallq66.setLocation(14, 12);
Wall wallq67 = new Wall();
addObject(wallq67, 15, 12);
```

```
wallq67.setLocation(15, 12);
Wall wallq68 = new Wall();
addObject(wallq68, 15, 13);
wallq68.setLocation(15, 13);
Wall wallq69 = new Wall();
addObject(wallq69, 14, 13);
wallq69.setLocation(14, 13);
Wall wallq6q1 = new Wall();
addObject(wallq6q1, 13, 13);
wallq6q1.setLocation(13, 13);
Wall wallq6q2 = new Wall();
addObject(wallq6q2, 12, 13);
wallq6q2.setLocation(12, 13);
Wall wallq6q3 = new Wall();
addObject(wallq6q3, 12, 14);
wallq6q3.setLocation(12, 14);
Wall wallq6q4 = new Wall();
addObject(wallq6q4, 12, 15);
wallq6q4.setLocation(12, 15);
Wall wallq6q5 = new Wall();
addObject(wallq6q5, 12, 16);
wallq6q5.setLocation(12, 16);
Wall wallq6q6 = new Wall();
addObject(wallq6q6, 11, 16);
wallq6q6.setLocation(11, 16);
Wall wallq6q7 = new Wall();
addObject(wallq6q7, 11, 15);
wallq6q7.setLocation(11, 15);
Wall wallq6q8 = new Wall();
addObject(wallq6q8, 11, 14);
wallq6q8.setLocation(11, 14);
Wall wallq6q9 = new Wall();
addObject(wallq6q9, 11, 13);
```

```
wallq6q9.setLocation(11, 13);
Wall wallq6qq1 = new Wall();
addObject(wallq6qq1, 11, 12);
wallq6qq1.setLocation(11, 12);
Wall wallq6qq2 = new Wall();
addObject(wallq6qq2, 11, 11);
wallq6qq2.setLocation(11, 11);
Wall wallq6qq3 = new Wall();
addObject(wallq6qq3, 11, 10);
wallq6qq3.setLocation(11, 10);

// Middle walls (Square 7)
Wall wallq7 = new Wall();
addObject(wallq7, 14, 9);
wallq7.setLocation(14, 9);
Wall wallq71 = new Wall();
addObject(wallq71, 15, 9);
wallq71.setLocation(15, 9);
Wall wallq72 = new Wall();
addObject(wallq72, 16, 9);
wallq72.setLocation(16, 9);
Wall wallq73 = new Wall();
addObject(wallq73, 17, 9);
wallq73.setLocation(17, 9);
Wall wallq74 = new Wall();
addObject(wallq74, 18, 9);
wallq74.setLocation(18, 9);
Wall wallq75 = new Wall();
addObject(wallq75, 19, 9);
wallq75.setLocation(19, 9);
Wall wallq76 = new Wall();
addObject(wallq76, 20, 9);
```

```
wallq76.setLocation(20, 9);
Wall wallq77 = new Wall();
addObject(wallq77, 21, 9);
wallq77.setLocation(21, 9);
Wall wallq78 = new Wall();
addObject(wallq78, 21, 10);
wallq78.setLocation(21, 10);
Wall wallq79 = new Wall();
addObject(wallq79, 20, 10);
wallq79.setLocation(20, 10);
Wall wallq7q = new Wall();
addObject(wallq7q, 19, 10);
wallq7q.setLocation(19, 10);
Wall wallq7q1 = new Wall();
addObject(wallq7q1, 18, 10);
wallq7q1.setLocation(18, 10);
Wall wallq7q2 = new Wall();
addObject(wallq7q2, 18, 11);
wallq7q2.setLocation(18, 11);
Wall wallq7q3 = new Wall();
addObject(wallq7q3, 18, 12);
wallq7q3.setLocation(18, 12);
Wall wallq7q4 = new Wall();
addObject(wallq7q4, 18, 13);
wallq7q4.setLocation(18, 13);
Wall wallq7q5 = new Wall();
addObject(wallq7q5, 17, 13);
wallq7q5.setLocation(17, 13);
Wall wallq7q6 = new Wall();
addObject(wallq7q6, 17, 12);
wallq7q6.setLocation(17, 12);
Wall wallq7q7 = new Wall();
addObject(wallq7q7, 17, 11);
```

```
wallq7q7.setLocation(17, 11);
Wall wallq7q8 = new Wall();
addObject(wallq7q8, 17, 10);
wallq7q8.setLocation(17, 10);
Wall wallq7q9 = new Wall();
addObject(wallq7q9, 16, 10);
wallq7q9.setLocation(16, 10);
Wall wallq7qq = new Wall();
addObject(wallq7qq, 15, 10);
wallq7qq.setLocation(15, 10);
Wall wallq7qq1 = new Wall();
addObject(wallq7qq1, 14, 10);
wallq7qq1.setLocation(14, 10);

// Middle walls (Square 8)
Wall wallq8 = new Wall();
addObject(wallq8, 23, 9);
wallq8.setLocation(23, 9);
Wall wallq81 = new Wall();
addObject(wallq81, 24, 9);
wallq81.setLocation(24, 9);
Wall wallq82 = new Wall();
addObject(wallq82, 24, 10);
wallq82.setLocation(24, 10);
Wall wallq83 = new Wall();
addObject(wallq83, 24, 11);
wallq83.setLocation(24, 11);
Wall wallq84 = new Wall();
addObject(wallq84, 24, 12);
wallq84.setLocation(24, 12);
Wall wallq85 = new Wall();
addObject(wallq85, 24, 13);
```

```
wallq85.setLocation(24, 13);
Wall wallq86 = new Wall();
addObject(wallq86, 24, 14);
wallq86.setLocation(24, 14);
Wall wallq87 = new Wall();
addObject(wallq87, 24, 15);
wallq87.setLocation(24, 15);
Wall wallq88 = new Wall();
addObject(wallq88, 24, 16);
wallq88.setLocation(24, 16);
Wall wallq89 = new Wall();
addObject(wallq89, 23, 16);
wallq89.setLocation(23, 16);
Wall wallq8q = new Wall();
addObject(wallq8q, 23, 15);
wallq8q.setLocation(23, 15);
Wall wallq8q1 = new Wall();
addObject(wallq8q1, 23, 14);
wallq8q1.setLocation(23, 14);
Wall wallq8q2 = new Wall();
addObject(wallq8q2, 23, 13);
wallq8q2.setLocation(23, 13);
Wall wallq8q3 = new Wall();
addObject(wallq8q3, 22, 13);
wallq8q3.setLocation(22, 13);
Wall wallq8q4 = new Wall();
addObject(wallq8q4, 21, 13);
wallq8q4.setLocation(21, 13);
Wall wallq8q5 = new Wall();
addObject(wallq8q5, 20, 13);
wallq8q5.setLocation(20, 13);
Wall wallq8q6 = new Wall();
addObject(wallq8q6, 20, 12);
```

```
wallq8q6.setLocation(20, 12);
Wall wallq8q7 = new Wall();
addObject(wallq8q7, 21, 12);
wallq8q7.setLocation(21, 12);
Wall wallq8q8 = new Wall();
addObject(wallq8q8, 22, 12);
wallq8q8.setLocation(22, 12);
Wall wallq8q9 = new Wall();
addObject(wallq8q9, 23, 12);
wallq8q9.setLocation(23, 12);
Wall wallq8qq = new Wall();
addObject(wallq8qq, 23, 11);
wallq8qq.setLocation(23, 11);
Wall wallq8qq1 = new Wall();
addObject(wallq8qq1, 23, 10);
wallq8qq1.setLocation(23, 10);

// Middle walls (Square 9)
Wall wallq9 = new Wall();
addObject(wallq9, 26, 9);
wallq9.setLocation(26, 9);
Wall wallq91 = new Wall();
addObject(wallq91, 27, 9);
wallq91.setLocation(27, 9);
Wall wallq92 = new Wall();
addObject(wallq92, 28, 9);
wallq92.setLocation(28, 9);
Wall wallq93 = new Wall();
addObject(wallq93, 29, 9);
wallq93.setLocation(29, 9);
Wall wallq94 = new Wall();
addObject(wallq94, 29, 10);
```

```
wallq94.setLocation(29, 10);
Wall wallq95 = new Wall();
addObject(wallq95, 28, 10);
wallq95.setLocation(28, 10);
Wall wallq96 = new Wall();
addObject(wallq96, 27, 10);
wallq96.setLocation(27, 10);
Wall wallq97 = new Wall();
addObject(wallq97, 26, 10);
wallq97.setLocation(26, 10);

// Middle walls (Square 10)
Wall wallqq10 = new Wall();
addObject(wallqq10, 11, 18);
wallqq10.setLocation(11, 18);
Wall wallqq101 = new Wall();
addObject(wallqq101, 12, 18);
wallqq101.setLocation(12, 18);
Wall wallqq102 = new Wall();
addObject(wallqq102, 12, 19);
wallqq102.setLocation(12, 19);
Wall wallqq103 = new Wall();
addObject(wallqq103, 12, 20);
wallqq103.setLocation(12, 20);
Wall wallqq104 = new Wall();
addObject(wallqq104, 12, 21);
wallqq104.setLocation(12, 21);
Wall wallqq105 = new Wall();
addObject(wallqq105, 12, 22);
wallqq105.setLocation(12, 22);
Wall wallqq106 = new Wall();
addObject(wallqq106, 11, 22);
```

```
wallqq106.setLocation(11, 22);
Wall wallqq107 = new Wall();
addObject(wallqq107, 11, 21);
wallqq107.setLocation(11, 21);
Wall wallqq108 = new Wall();
addObject(wallqq108, 11, 20);
wallqq108.setLocation(11, 20);
Wall wallqq109 = new Wall();
addObject(wallqq109, 11, 19);
wallqq109.setLocation(11, 19);

// Centre walls (Square 11)
Wall wallqq11 = new Wall();
addObject(wallqq11, 14, 15);
wallqq11.setLocation(14, 15);
Wall wallqq111 = new Wall();
addObject(wallqq111, 15, 15);
wallqq111.setLocation(15, 15);
Wall wallqq112 = new Wall();
addObject(wallqq112, 16, 15);
wallqq112.setLocation(16, 15);
Wall wallqq113 = new Wall();

// Door
addObject(wallqq113, 19, 15);
wallqq113.setLocation(19, 15);
Wall wallqq114 = new Wall();
addObject(wallqq114, 20, 15);
wallqq114.setLocation(20, 15);
Wall wallqq115 = new Wall();
addObject(wallqq115, 21, 15);
wallqq115.setLocation(21, 15);
```

```
Wall wallqq116 = new Wall();
addObject(wallqq116, 21, 16);
wallqq116.setLocation(21, 16);
Wall wallqq117 = new Wall();
addObject(wallqq117, 21, 17);
wallqq117.setLocation(21, 17);
Wall wallqq118 = new Wall();
addObject(wallqq118, 21, 18);
wallqq118.setLocation(21, 18);
Wall wallqq119 = new Wall();
addObject(wallqq119, 21, 19);
wallqq119.setLocation(21, 19);
Wall wallqq11q = new Wall();
addObject(wallqq11q, 20, 19);
wallqq11q.setLocation(20, 19);
Wall wallqq11q1 = new Wall();
addObject(wallqq11q1, 19, 19);
wallqq11q1.setLocation(19, 19);
Wall wallqq11q2 = new Wall();
addObject(wallqq11q2, 18, 19);
wallqq11q2.setLocation(18, 19);
Wall wallqq11q3 = new Wall();
addObject(wallqq11q3, 17, 19);
wallqq11q3.setLocation(17, 19);
Wall wallqq11q4 = new Wall();
addObject(wallqq11q4, 16, 19);
wallqq11q4.setLocation(16, 19);
Wall wallqq11q5 = new Wall();
addObject(wallqq11q5, 15, 19);
wallqq11q5.setLocation(15, 19);
Wall wallqq11q6 = new Wall();
addObject(wallqq11q6, 14, 19);
wallqq11q6.setLocation(14, 19);
```

```
Wall wallqq11q7 = new Wall();
addObject(wallqq11q7, 14, 18);
wallqq11q7.setLocation(14, 18);
Wall wallqq11q8 = new Wall();
addObject(wallqq11q8, 14, 17);
wallqq11q8.setLocation(14, 17);
Wall wallqq11q9 = new Wall();
addObject(wallqq11q9, 14, 16);
wallqq11q9.setLocation(14, 16);

// Centre walls (Square 12)
Wall wallqq12 = new Wall();
addObject(wallqq12, 23, 18);
wallqq12.setLocation(23, 18);
Wall wallqq121 = new Wall();
addObject(wallqq121, 24, 18);
wallqq121.setLocation(24, 18);
Wall wallqq122 = new Wall();
addObject(wallqq122, 24, 19);
wallqq122.setLocation(24, 19);
Wall wallqq123 = new Wall();
addObject(wallqq123, 24, 20);
wallqq123.setLocation(24, 20);
Wall wallqq124 = new Wall();
addObject(wallqq124, 24, 21);
wallqq124.setLocation(24, 21);
Wall wallqq125 = new Wall();
addObject(wallqq125, 24, 22);
wallqq125.setLocation(24, 22);
Wall wallqq126 = new Wall();
addObject(wallqq126, 23, 22);
wallqq126.setLocation(23, 22);
```

```
Wall wallqq127 = new Wall();
addObject(wallqq127, 23, 21);
wallqq127.setLocation(23, 21);
Wall wallqq128 = new Wall();
addObject(wallqq128, 23, 20);
wallqq128.setLocation(23, 20);
Wall wallqq129 = new Wall();
addObject(wallqq129, 23, 19);
wallqq129.setLocation(23, 19);

// Centre walls (Square 13)
Wall wallqq13 = new Wall();
addObject(wallqq13, 14, 21);
wallqq13.setLocation(14, 21);
Wall wallqq131 = new Wall();
addObject(wallqq131, 15, 21);
wallqq131.setLocation(15, 21);
Wall wallqq132 = new Wall();
addObject(wallqq132, 16, 21);
wallqq132.setLocation(16, 21);
Wall wallqq133 = new Wall();
addObject(wallqq133, 17, 21);
wallqq133.setLocation(17, 21);
Wall wallqq134 = new Wall();
addObject(wallqq134, 18, 21);
wallqq134.setLocation(18, 21);
Wall wallqq135 = new Wall();
addObject(wallqq135, 19, 21);
wallqq135.setLocation(19, 21);
Wall wallqq136 = new Wall();
addObject(wallqq136, 20, 21);
wallqq136.setLocation(20, 21);
```

```
Wall wallqq137 = new Wall();
addObject(wallqq137, 21, 21);
wallqq137.setLocation(21, 21);
Wall wallqq138 = new Wall();
addObject(wallqq138, 21, 22);
wallqq138.setLocation(21, 22);
Wall wallqq139 = new Wall();
addObject(wallqq139, 20, 22);
wallqq139.setLocation(20, 22);
Wall wallqq13q = new Wall();
addObject(wallqq13q, 19, 22);
wallqq13q.setLocation(19, 22);
Wall wallqq13q1 = new Wall();
addObject(wallqq13q1, 18, 22);
wallqq13q1.setLocation(18, 22);
Wall wallqq13q2 = new Wall();
addObject(wallqq13q2, 18, 23);
wallqq13q2.setLocation(18, 23);
Wall wallqq13q3 = new Wall();
addObject(wallqq13q3, 18, 24);
wallqq13q3.setLocation(18, 24);
Wall wallqq13q4 = new Wall();
addObject(wallqq13q4, 18, 25);
wallqq13q4.setLocation(18, 25);
Wall wallqq13q5 = new Wall();
addObject(wallqq13q5, 17, 25);
wallqq13q5.setLocation(17, 25);
Wall wallqq13q6 = new Wall();
addObject(wallqq13q6, 17, 24);
wallqq13q6.setLocation(17, 24);
Wall wallqq13q7 = new Wall();
addObject(wallqq13q7, 17, 23);
wallqq13q7.setLocation(17, 23);
```

```
Wall wallqq13q8 = new Wall();
addObject(wallqq13q8, 17, 22);
wallqq13q8.setLocation(17, 22);
Wall wallqq13q9 = new Wall();
addObject(wallqq13q9, 16, 22);
wallqq13q9.setLocation(16, 22);
Wall wallqq13qq = new Wall();
addObject(wallqq13qq, 15, 22);
wallqq13qq.setLocation(15, 22);
Wall wallqq13qq1 = new Wall();
addObject(wallqq13qq1, 14, 22);
wallqq13qq1.setLocation(14, 22);

// Bottom/middle walls (Square 14)

Wall wallqq14 = new Wall();
addObject(wallqq14, 6, 24);
wallqq14.setLocation(6, 24);
Wall wallqq141 = new Wall();
addObject(wallqq141, 7, 24);
wallqq141.setLocation(7, 24);
Wall wallqq142 = new Wall();
addObject(wallqq142, 8, 24);
wallqq142.setLocation(8, 24);
Wall wallqq143 = new Wall();
addObject(wallqq143, 9, 24);
wallqq143.setLocation(9, 24);
Wall wallqq144 = new Wall();
addObject(wallqq144, 9, 25);
wallqq144.setLocation(9, 25);
Wall wallqq145 = new Wall();
addObject(wallqq145, 9, 26);
```

```
wallqq145.setLocation(9, 26);
Wall wallqq146 = new Wall();
addObject(wallqq146, 9, 27);
wallqq146.setLocation(9, 27);
Wall wallqq147 = new Wall();
addObject(wallqq147, 9, 28);
wallqq147.setLocation(9, 28);
Wall wallqq148 = new Wall();
addObject(wallqq148, 8, 28);
wallqq148.setLocation(8, 28);
Wall wallqq149 = new Wall();
addObject(wallqq149, 8, 27);
wallqq149.setLocation(8, 27);
Wall wallqq14q = new Wall();
addObject(wallqq14q, 8, 26);
wallqq14q.setLocation(8, 26);
Wall wallqq14q1 = new Wall();
addObject(wallqq14q1, 8, 25);
wallqq14q1.setLocation(8, 25);
Wall wallqq14q2 = new Wall();
addObject(wallqq14q2, 7, 25);
wallqq14q2.setLocation(7, 25);
Wall wallqq14q3 = new Wall();
addObject(wallqq14q3, 6, 25);
wallqq14q3.setLocation(6, 25);

// Bottom/middle walls (Square 15)
Wall wallqq15 = new Wall();
addObject(wallqq15, 11, 24);
wallqq15.setLocation(11, 24);
Wall wallqq151 = new Wall();
addObject(wallqq151, 12, 24);
```

```
wallqq151.setLocation(12, 24);
Wall wallqq152 = new Wall();
addObject(wallqq152, 13, 24);
wallqq152.setLocation(13, 24);
Wall wallqq153 = new Wall();
addObject(wallqq153, 14, 24);
wallqq153.setLocation(14, 24);
Wall wallqq154 = new Wall();
addObject(wallqq154, 15, 24);
wallqq154.setLocation(15, 24);
Wall wallqq155 = new Wall();
addObject(wallqq155, 15, 25);
wallqq155.setLocation(15, 25);
Wall wallqq156 = new Wall();
addObject(wallqq156, 14, 25);
wallqq156.setLocation(14, 25);
Wall wallqq157 = new Wall();
addObject(wallqq157, 13, 25);
wallqq157.setLocation(13, 25);
Wall wallqq158 = new Wall();
addObject(wallqq158, 12, 25);
wallqq158.setLocation(12, 25);
Wall wallqq159 = new Wall();
addObject(wallqq159, 11, 25);
wallqq159.setLocation(11, 25);

// Bottom/middle walls (Square 16)
Wall wallqq16 = new Wall();
addObject(wallqq16, 20, 24);
wallqq16.setLocation(20, 24);
Wall wallqq161 = new Wall();
addObject(wallqq161, 21, 24);
```

```
wallqq161.setLocation(21, 24);
Wall wallqq162 = new Wall();
addObject(wallqq162, 22, 24);
wallqq162.setLocation(22, 24);
Wall wallqq163 = new Wall();
addObject(wallqq163, 23, 24);
wallqq163.setLocation(23, 24);
Wall wallqq164 = new Wall();
addObject(wallqq164, 24, 24);
wallqq164.setLocation(24, 24);
Wall wallqq165 = new Wall();
addObject(wallqq165, 24, 25);
wallqq165.setLocation(24, 25);
Wall wallqq166 = new Wall();
addObject(wallqq166, 23, 25);
wallqq166.setLocation(23, 25);
Wall wallqq167 = new Wall();
addObject(wallqq167, 22, 25);
wallqq167.setLocation(22, 25);
Wall wallqq168 = new Wall();
addObject(wallqq168, 21, 25);
wallqq168.setLocation(21, 25);
Wall wallqq169 = new Wall();
addObject(wallqq169, 20, 25);
wallqq169.setLocation(20, 25);

// Bottom/middle walls (Square 17)
Wall wallqq17 = new Wall();
addObject(wallqq17, 26, 24);
wallqq17.setLocation(26, 24);
Wall wallqq171 = new Wall();
addObject(wallqq171, 27, 24);
```

```
wallqq171.setLocation(27, 24);
Wall wallqq172 = new Wall();
addObject(wallqq172, 28, 24);
wallqq172.setLocation(28, 24);
Wall wallqq173 = new Wall();
addObject(wallqq173, 29, 24);
wallqq173.setLocation(29, 24);
Wall wallqq174 = new Wall();
addObject(wallqq174, 29, 25);
wallqq174.setLocation(29, 25);
Wall wallqq175 = new Wall();
addObject(wallqq175, 28, 25);
wallqq175.setLocation(28, 25);
Wall wallqq176 = new Wall();
addObject(wallqq176, 27, 25);
wallqq176.setLocation(27, 25);
Wall wallqq177 = new Wall();
addObject(wallqq177, 27, 26);
wallqq177.setLocation(27, 26);
Wall wallqq178 = new Wall();
addObject(wallqq178, 27, 27);
wallqq178.setLocation(27, 27);
Wall wallqq179 = new Wall();
addObject(wallqq179, 27, 28);
wallqq179.setLocation(27, 28);
Wall wallqq17q = new Wall();
addObject(wallqq17q, 26, 28);
wallqq17q.setLocation(26, 28);
Wall wallqq17q1 = new Wall();
addObject(wallqq17q1, 26, 27);
wallqq17q1.setLocation(26, 27);
Wall wallqq17q2 = new Wall();
addObject(wallqq17q2, 26, 26);
```

```
wallqq17q2.setLocation(26, 26);
Wall wallqq17q3 = new Wall();
addObject(wallqq17q3, 26, 25);
wallqq17q3.setLocation(26, 25);

// Bottom/middle walls (Square 18)
Wall wallqq18 = new Wall();
addObject(wallqq18, 6, 30);
wallqq18.setLocation(6, 30);
Wall wallqq181 = new Wall();
addObject(wallqq181, 7, 30);
wallqq181.setLocation(7, 30);
Wall wallqq182 = new Wall();
addObject(wallqq182, 8, 30);
wallqq182.setLocation(8, 30);
Wall wallqq183 = new Wall();
addObject(wallqq183, 9, 30);
wallqq183.setLocation(9, 30);
Wall wallqq184 = new Wall();
addObject(wallqq184, 10, 30);
wallqq184.setLocation(10, 30);
Wall wallqq185 = new Wall();
addObject(wallqq185, 11, 30);
wallqq185.setLocation(11, 30);
Wall wallqq186 = new Wall();
addObject(wallqq186, 11, 29);
wallqq186.setLocation(11, 29);
Wall wallqq187 = new Wall();
addObject(wallqq187, 11, 28);
wallqq187.setLocation(11, 28);
Wall wallqq188 = new Wall();
addObject(wallqq188, 11, 27);
```

```
wallqq188.setLocation(11, 27);
Wall wallqq189 = new Wall();
addObject(wallqq189, 12, 27);
wallqq189.setLocation(12, 27);
Wall wallqq18q = new Wall();
addObject(wallqq18q, 12, 28);
wallqq18q.setLocation(12, 28);
Wall wallqq18q1 = new Wall();
addObject(wallqq18q1, 12, 29);
wallqq18q1.setLocation(12, 29);
Wall wallqq18q2 = new Wall();
addObject(wallqq18q2, 12, 30);
wallqq18q2.setLocation(12, 30);
Wall wallqq18q3 = new Wall();
addObject(wallqq18q3, 13, 30);
wallqq18q3.setLocation(13, 30);
Wall wallqq18q4 = new Wall();
addObject(wallqq18q4, 14, 30);
wallqq18q4.setLocation(14, 30);
Wall wallqq18q5 = new Wall();
addObject(wallqq18q5, 15, 30);
wallqq18q5.setLocation(15, 30);
Wall wallqq18q6 = new Wall();
addObject(wallqq18q6, 15, 31);
wallqq18q6.setLocation(15, 31);
Wall wallqq18q7 = new Wall();
addObject(wallqq18q7, 14, 31);
wallqq18q7.setLocation(14, 31);
Wall wallqq18q8 = new Wall();
addObject(wallqq18q8, 13, 31);
wallqq18q8.setLocation(13, 31);
Wall wallqq18q9 = new Wall();
addObject(wallqq18q9, 12, 31);
```

```
wallqq18q9.setLocation(12, 31);
Wall wallqq18qq = new Wall();
addObject(wallqq18qq, 11, 31);
wallqq18qq.setLocation(11, 31);
Wall wallqq18qq1 = new Wall();
addObject(wallqq18qq1, 10, 31);
wallqq18qq1.setLocation(10, 31);
Wall wallqq18qq2 = new Wall();
addObject(wallqq18qq2, 9, 31);
wallqq18qq2.setLocation(9, 31);
Wall wallqq18qq3 = new Wall();
addObject(wallqq18qq3, 8, 31);
wallqq18qq3.setLocation(8, 31);
Wall wallqq18qq4 = new Wall();
addObject(wallqq18qq4, 7, 31);
wallqq18qq4.setLocation(7, 31);
Wall wallqq18qq5 = new Wall();
addObject(wallqq18qq5, 6, 31);
wallqq18qq5.setLocation(6, 31);

// Bottom/middle walls (Square 19)
Wall wallqq19 = new Wall();
addObject(wallqq19, 14, 27);
wallqq19.setLocation(14, 27);
Wall wallqq191 = new Wall();
addObject(wallqq191, 15, 27);
wallqq191.setLocation(15, 27);
Wall wallqq192 = new Wall();
addObject(wallqq192, 16, 27);
wallqq192.setLocation(16, 27);
Wall wallqq193 = new Wall();
addObject(wallqq193, 17, 27);
```

```
wallqq193.setLocation(17, 27);
Wall wallqq194 = new Wall();
addObject(wallqq194, 18, 27);
wallqq194.setLocation(18, 27);
Wall wallqq195 = new Wall();
addObject(wallqq195, 19, 27);
wallqq195.setLocation(19, 27);
Wall wallqq196 = new Wall();
addObject(wallqq196, 20, 27);
wallqq196.setLocation(20, 27);
Wall wallqq197 = new Wall();
addObject(wallqq197, 21, 27);
wallqq197.setLocation(21, 27);
Wall wallqq198 = new Wall();
addObject(wallqq198, 21, 28);
wallqq198.setLocation(21, 28);
Wall wallqq199 = new Wall();
addObject(wallqq199, 20, 28);
wallqq199.setLocation(20, 28);
Wall wallqq19q = new Wall();
addObject(wallqq19q, 19, 28);
wallqq19q.setLocation(19, 28);
Wall wallqq19q1 = new Wall();
addObject(wallqq19q1, 18, 28);
wallqq19q1.setLocation(18, 28);
Wall wallqq19q2 = new Wall();
addObject(wallqq19q2, 18, 29);
wallqq19q2.setLocation(18, 29);
Wall wallqq19q3 = new Wall();
addObject(wallqq19q3, 18, 30);
wallqq19q3.setLocation(18, 30);
Wall wallqq19q4 = new Wall();
addObject(wallqq19q4, 18, 31);
```

```
wallqq19q4.setLocation(18, 31);
Wall wallqq19q5 = new Wall();
addObject(wallqq19q5, 17, 31);
wallqq19q5.setLocation(17, 31);
Wall wallqq19q6 = new Wall();
addObject(wallqq19q6, 17, 30);
wallqq19q6.setLocation(17, 30);
Wall wallqq19q7 = new Wall();
addObject(wallqq19q7, 17, 29);
wallqq19q7.setLocation(17, 29);
Wall wallqq19q8 = new Wall();
addObject(wallqq19q8, 17, 28);
wallqq19q8.setLocation(17, 28);
Wall wallqq19q9 = new Wall();
addObject(wallqq19q9, 16, 28);
wallqq19q9.setLocation(16, 28);
Wall wallqq19qq = new Wall();
addObject(wallqq19qq, 15, 28);
wallqq19qq.setLocation(15, 28);
Wall wallqq19qq1 = new Wall();
addObject(wallqq19qq1, 14, 28);
wallqq19qq1.setLocation(14, 28);

// Bottom/middle walls (Square 20)
Wall wallqq20 = new Wall();
addObject(wallqq20, 20, 30);
wallqq20.setLocation(20, 30);
Wall wallqq201 = new Wall();
addObject(wallqq201, 21, 30);
wallqq201.setLocation(21, 30);
Wall wallqq202 = new Wall();
addObject(wallqq202, 22, 30);
```

```
wallqq202.setLocation(22, 30);
Wall wallqq203 = new Wall();
addObject(wallqq203, 23, 30);
wallqq203.setLocation(23, 30);
Wall wallqq204 = new Wall();
addObject(wallqq204, 23, 29);
wallqq204.setLocation(23, 29);
Wall wallqq205 = new Wall();
addObject(wallqq205, 23, 28);
wallqq205.setLocation(23, 28);
Wall wallqq206 = new Wall();
addObject(wallqq206, 23, 27);
wallqq206.setLocation(23, 27);
Wall wallqq207 = new Wall();
addObject(wallqq207, 24, 27);
wallqq207.setLocation(24, 27);
Wall wallqq208 = new Wall();
addObject(wallqq208, 24, 28);
wallqq208.setLocation(24, 28);
Wall wallqq209 = new Wall();
addObject(wallqq209, 24, 29);
wallqq209.setLocation(24, 29);
Wall wallqq20q = new Wall();
addObject(wallqq20q, 24, 30);
wallqq20q.setLocation(24, 30);
Wall wallqq20q1 = new Wall();
addObject(wallqq20q1, 25, 30);
wallqq20q1.setLocation(25, 30);
Wall wallqq20q2 = new Wall();
addObject(wallqq20q2, 26, 30);
wallqq20q2.setLocation(26, 30);
Wall wallqq20q3 = new Wall();
addObject(wallqq20q3, 27, 30);
```

```
wallqq20q3.setLocation(27, 30);
Wall wallqq20q4 = new Wall();
addObject(wallqq20q4, 28, 30);
wallqq20q4.setLocation(28, 30);
Wall wallqq20q5 = new Wall();
addObject(wallqq20q5, 29, 30);
wallqq20q5.setLocation(29, 30);
Wall wallqq20q6 = new Wall();
addObject(wallqq20q6, 29, 31);
wallqq20q6.setLocation(29, 31);
Wall wallqq20q7 = new Wall();
addObject(wallqq20q7, 28, 31);
wallqq20q7.setLocation(28, 31);
Wall wallqq20q8 = new Wall();
addObject(wallqq20q8, 27, 31);
wallqq20q8.setLocation(27, 31);
Wall wallqq20q9 = new Wall();
addObject(wallqq20q9, 26, 31);
wallqq20q9.setLocation(26, 31);
Wall wallqq20qq = new Wall();
addObject(wallqq20qq, 25, 31);
wallqq20qq.setLocation(25, 31);
Wall wallqq20qq1 = new Wall();
addObject(wallqq20qq1, 24, 31);
wallqq20qq1.setLocation(24, 31);
Wall wallqq20qq2 = new Wall();
addObject(wallqq20qq2, 23, 31);
wallqq20qq2.setLocation(23, 31);
Wall wallqq20qq3 = new Wall();
addObject(wallqq20qq3, 22, 31);
wallqq20qq3.setLocation(22, 31);
Wall wallqq20qq4 = new Wall();
addObject(wallqq20qq4, 21, 31);
```

```
wallqq20qq4.setLocation(21, 31);
Wall wallqq20qq5 = new Wall();
addObject(wallqq20qq5, 20, 31);
wallqq20qq5.setLocation(20, 31);

// Left walls
Wall walle = new Wall();
addObject(walle, 4, 32);
walle.setLocation(4, 32);
Wall walle1 = new Wall();
addObject(walle1, 4, 31);
walle1.setLocation(4, 31);
Wall walle2 = new Wall();
addObject(walle2, 4, 30);
walle2.setLocation(4, 30);
Wall walle3 = new Wall();
addObject(walle3, 4, 29);
walle3.setLocation(4, 29);
Wall walle4 = new Wall();
addObject(walle4, 4, 28);
walle4.setLocation(4, 28);
Wall walle5 = new Wall();
addObject(walle5, 5, 28);
walle5.setLocation(5, 28);
Wall walle6 = new Wall();
addObject(walle6, 6, 28);
walle6.setLocation(6, 28);
Wall walle7 = new Wall();
addObject(walle7, 6, 27);
walle7.setLocation(6, 27);
Wall walle8 = new Wall();
addObject(walle8, 5, 27);
```

```
walle8.setLocation(5, 27);
Wall walle9 = new Wall();
addObject(walle9, 4, 27);
walle9.setLocation(4, 27);
Wall walle10 = new Wall();
addObject(walle10, 4, 26);
walle10.setLocation(4, 26);
Wall walle11 = new Wall();
addObject(walle11, 4, 25);
walle11.setLocation(4, 25);
Wall walle12 = new Wall();
addObject(walle12, 4, 24);
walle12.setLocation(4, 24);
Wall walle13 = new Wall();
addObject(walle13, 4, 23);
walle13.setLocation(4, 23);
Wall walle14 = new Wall();
addObject(walle14, 4, 24);
walle14.setLocation(4, 24);
Wall walle15 = new Wall();
addObject(walle15, 4, 23);
walle15.setLocation(4, 23);
Wall walle16 = new Wall();
addObject(walle16, 4, 22);
walle16.setLocation(4, 22);
Wall walle17 = new Wall();
addObject(walle17, 5, 22);
walle17.setLocation(5, 22);
Wall walle18 = new Wall();
addObject(walle18, 6, 22);
walle18.setLocation(6, 22);
Wall walle19 = new Wall();
addObject(walle19, 7, 22);
```

```
walle19.setLocation(7, 22);
Wall walle20 = new Wall();
addObject(walle20, 8, 22);
walle20.setLocation(8, 22);
Wall walle21 = new Wall();
addObject(walle21, 9, 22);
walle21.setLocation(9, 22);
Wall walle22 = new Wall();
addObject(walle22, 9, 21);
walle22.setLocation(9, 21);
Wall walle23 = new Wall();
addObject(walle23, 9, 20);
walle23.setLocation(9, 20);
Wall walle24 = new Wall();
addObject(walle24, 9, 19);
walle24.setLocation(9, 19);
Wall walle25 = new Wall();
addObject(walle25, 9, 18);
walle25.setLocation(9, 18);
Wall walle26 = new Wall();
addObject(walle26, 8, 18);
walle26.setLocation(8, 18);
Wall walle27 = new Wall();
addObject(walle27, 7, 18);
walle27.setLocation(7, 18);
Wall walle28 = new Wall();
addObject(walle28, 6, 18);
walle28.setLocation(6, 18);
Wall walle29 = new Wall();
addObject(walle29, 5, 18);
walle29.setLocation(5, 18);
Wall walle30 = new Wall();
addObject(walle30, 4, 18);
```

```
walle30.setLocation(4, 18);
Wall walle31 = new Wall();
addObject(walle31, 4, 16);
walle31.setLocation(4, 16);
Wall walle32 = new Wall();
addObject(walle32, 5, 16);
walle32.setLocation(5, 16);
Wall walle33 = new Wall();
addObject(walle33, 6, 16);
walle33.setLocation(6, 16);
Wall walle34 = new Wall();
addObject(walle34, 7, 16);
walle34.setLocation(7, 16);
Wall walle35 = new Wall();
addObject(walle35, 8, 16);
walle35.setLocation(8, 16);
Wall walle36 = new Wall();
addObject(walle36, 9, 16);
walle36.setLocation(9, 16);
Wall walle37 = new Wall();
addObject(walle37, 9, 15);
walle37.setLocation(9, 15);
Wall walle38 = new Wall();
addObject(walle38, 9, 14);
walle38.setLocation(9, 14);
Wall walle39 = new Wall();
addObject(walle39, 9, 13);
walle39.setLocation(9, 13);
Wall walle40 = new Wall();
addObject(walle40, 9, 12);
walle40.setLocation(9, 12);
Wall walle41 = new Wall();
addObject(walle41, 8, 12);
```

```
walle41.setLocation(8, 12);
Wall walle42 = new Wall();
addObject(walle42, 7, 12);
walle42.setLocation(7, 12);
Wall walle43 = new Wall();
addObject(walle43, 6, 12);
walle43.setLocation(6, 12);
Wall walle44 = new Wall();
addObject(walle44, 5, 12);
walle44.setLocation(5, 12);
Wall walle45 = new Wall();
addObject(walle45, 4, 12);
walle45.setLocation(4, 12);
Wall walle46 = new Wall();
addObject(walle46, 4, 11);
walle46.setLocation(4, 11);
Wall walle47 = new Wall();
addObject(walle47, 4, 10);
walle47.setLocation(4, 10);
Wall walle48 = new Wall();
addObject(walle48, 4, 9);
walle48.setLocation(4, 9);
Wall walle49 = new Wall();
addObject(walle49, 4, 8);
walle49.setLocation(4, 8);
Wall walle50 = new Wall();
addObject(walle50, 4, 7);
walle50.setLocation(4, 7);
Wall walle51 = new Wall();
addObject(walle51, 4, 6);
walle51.setLocation(4, 6);
Wall walle52 = new Wall();
addObject(walle52, 4, 5);
```

```
walle52.setLocation(4, 5);
Wall walle53 = new Wall();
addObject(walle53, 4, 4);
walle53.setLocation(4, 4);

// Bottom walls
Wall wallb4 = new Wall();
addObject(wallb4, 4, 33);
wallb4.setLocation(4, 33);
Wall wallb5 = new Wall();
addObject(wallb5, 5, 33);
wallb5.setLocation(5, 33);
Wall wallb6 = new Wall();
addObject(wallb6, 6, 33);
wallb6.setLocation(6, 33);
Wall wallb7 = new Wall();
addObject(wallb7, 7, 33);
wallb7.setLocation(7, 33);
Wall wallb8 = new Wall();
addObject(wallb8, 8, 33);
wallb8.setLocation(8, 33);
Wall wallb9 = new Wall();
addObject(wallb9, 9, 33);
wallb9.setLocation(9, 33);
Wall wallb10 = new Wall();
addObject(wallb10, 10, 33);
wallb10.setLocation(10, 33);
Wall wallb11 = new Wall();
addObject(wallb11, 11, 33);
wallb11.setLocation(11, 33);
Wall wallb12 = new Wall();
addObject(wallb12, 12, 33);
```

```
wallb12.setLocation(12, 33);
Wall wallb13 = new Wall();
addObject(wallb13, 13, 33);
wallb13.setLocation(13, 33);
Wall wallb14 = new Wall();
addObject(wallb14, 14, 33);
wallb14.setLocation(14, 33);
Wall wallb15 = new Wall();
addObject(wallb15, 15, 33);
wallb15.setLocation(15, 33);
Wall wallb16 = new Wall();
addObject(wallb16, 16, 33);
wallb16.setLocation(16, 33);
Wall wallb17 = new Wall();
addObject(wallb17, 17, 33);
wallb17.setLocation(17, 33);
Wall wallb18 = new Wall();
addObject(wallb18, 18, 33);
wallb18.setLocation(18, 33);
Wall wallb19 = new Wall();
addObject(wallb19, 19, 33);
wallb19.setLocation(19, 33);
Wall wallb20 = new Wall();
addObject(wallb20, 20, 33);
wallb20.setLocation(20, 33);
Wall wallb21 = new Wall();
addObject(wallb21, 21, 33);
wallb21.setLocation(21, 33);
Wall wallb22 = new Wall();
addObject(wallb22, 22, 33);
wallb22.setLocation(22, 33);
Wall wallb23 = new Wall();
addObject(wallb23, 23, 33);
```

```
wallb23.setLocation(23, 33);
Wall wallb24 = new Wall();
addObject(wallb24, 24, 33);
wallb24.setLocation(24, 33);
Wall wallb25 = new Wall();
addObject(wallb25, 25, 33);
wallb25.setLocation(25, 33);
Wall wallb26 = new Wall();
addObject(wallb26, 26, 33);
wallb26.setLocation(26, 33);
Wall wallb27 = new Wall();
addObject(wallb27, 27, 33);
wallb27.setLocation(27, 33);
Wall wallb28 = new Wall();
addObject(wallb28, 28, 33);
wallb28.setLocation(28, 33);
Wall wallb29 = new Wall();
addObject(wallb29, 29, 33);
wallb29.setLocation(29, 33);
Wall wallb30 = new Wall();
addObject(wallb30, 30, 33);
wallb30.setLocation(30, 33);
Wall wallb31 = new Wall();
addObject(wallb31, 31, 33);
wallb31.setLocation(31, 33);

// Right walls
Wall walld = new Wall();
addObject(walld, 31, 32);
walld.setLocation(31, 32);
Wall walld1 = new Wall();
addObject(walld1, 31, 31);
```

```
walld1.setLocation(31, 31);
Wall walld2 = new Wall();
addObject(walld2, 31, 30);
walld2.setLocation(31, 30);
Wall walld3 = new Wall();
addObject(walld3, 31, 29);
walld3.setLocation(31, 29);
Wall walld4 = new Wall();
addObject(walld4, 31, 28);
walld4.setLocation(31, 28);
Wall walld5 = new Wall();
addObject(walld5, 30, 28);
walld5.setLocation(30, 28);
Wall walld6 = new Wall();
addObject(walld6, 29, 28);
walld6.setLocation(29, 28);
Wall walld7 = new Wall();
addObject(walld7, 29, 27);
walld7.setLocation(29, 27);
Wall walld8 = new Wall();
addObject(walld8, 30, 27);
walld8.setLocation(30, 27);
Wall walld9 = new Wall();
addObject(walld9, 31, 27);
walld9.setLocation(31, 27);
Wall walld10 = new Wall();
addObject(walld10, 31, 26);
walld10.setLocation(31, 26);
Wall walld11 = new Wall();
addObject(walld11, 31, 25);
walld11.setLocation(31, 25);
Wall walld12 = new Wall();
addObject(walld12, 31, 24);
```

```
walld12.setLocation(31, 24);
Wall walld13 = new Wall();
addObject(walld13, 31, 23);
walld13.setLocation(31, 23);
Wall walld14 = new Wall();
addObject(walld14, 31, 24);
walld14.setLocation(31, 24);
Wall walld15 = new Wall();
addObject(walld15, 31, 23);
walld15.setLocation(31, 23);
Wall walld16 = new Wall();
addObject(walld16, 31, 22);
walld16.setLocation(31, 22);
Wall walld17 = new Wall();
addObject(walld17, 30, 22);
walld17.setLocation(30, 22);
Wall walld18 = new Wall();
addObject(walld18, 29, 22);
walld18.setLocation(29, 22);
Wall walld19 = new Wall();
addObject(walld19, 28, 22);
walld19.setLocation(28, 22);
Wall walld20 = new Wall();
addObject(walld20, 27, 22);
walld20.setLocation(27, 22);
Wall walld21 = new Wall();
addObject(walld21, 26, 22);
walld21.setLocation(26, 22);
Wall walld22 = new Wall();
addObject(walld22, 26, 21);
walld22.setLocation(26, 21);
Wall walld23 = new Wall();
addObject(walld23, 26, 20);
```

```
walld23.setLocation(26, 20);
Wall walld24 = new Wall();
addObject(walld24, 26, 19);
walld24.setLocation(26, 19);
Wall walld25 = new Wall();
addObject(walld25, 26, 18);
walld25.setLocation(26, 18);
Wall walld26 = new Wall();
addObject(walld26, 27, 18);
walld26.setLocation(27, 18);
Wall walld27 = new Wall();
addObject(walld27, 28, 18);
walld27.setLocation(28, 18);
Wall walld28 = new Wall();
addObject(walld28, 29, 18);
walld28.setLocation(29, 18);
Wall walld29 = new Wall();
addObject(walld29, 30, 18);
walld29.setLocation(30, 18);
Wall walld30 = new Wall();
addObject(walld30, 31, 18);
walld30.setLocation(31, 18);
Wall walld31 = new Wall();
addObject(walld31, 31, 16);
walld31.setLocation(31, 16);
Wall walld32 = new Wall();
addObject(walld32, 30, 16);
walld32.setLocation(30, 16);
Wall walld33 = new Wall();
addObject(walld33, 29, 16);
walld33.setLocation(29, 16);
Wall walld34 = new Wall();
addObject(walld34, 28, 16);
```

```
walld34.setLocation(28, 16);
Wall walld35 = new Wall();
addObject(walld35, 27, 16);
walld35.setLocation(27, 16);
Wall walld36 = new Wall();
addObject(walld36, 26, 16);
walld36.setLocation(26, 16);
Wall walld37 = new Wall();
addObject(walld37, 26, 15);
walld37.setLocation(26, 15);
Wall walld38 = new Wall();
addObject(walld38, 26, 14);
walld38.setLocation(26, 14);
Wall walld39 = new Wall();
addObject(walld39, 26, 13);
walld39.setLocation(26, 13);
Wall walld40 = new Wall();
addObject(walld40, 26, 12);
walld40.setLocation(26, 12);
Wall walld41 = new Wall();
addObject(walld41, 27, 12);
walld41.setLocation(27, 12);
Wall walld42 = new Wall();
addObject(walld42, 28, 12);
walld42.setLocation(28, 12);
Wall walld43 = new Wall();
addObject(walld43, 29, 12);
walld43.setLocation(29, 12);
Wall walld44 = new Wall();
addObject(walld44, 30, 12);
walld44.setLocation(30, 12);
Wall walld45 = new Wall();
addObject(walld45, 31, 12);
```

```
walld45.setLocation(31, 12);
Wall walld46 = new Wall();
addObject(walld46, 31, 11);
walld46.setLocation(31, 11);
Wall walld47 = new Wall();
addObject(walld47, 31, 10);
walld47.setLocation(31, 10);
Wall walld48 = new Wall();
addObject(walld48, 31, 9);
walld48.setLocation(31, 9);
Wall walld49 = new Wall();
addObject(walld49, 31, 8);
walld49.setLocation(31, 8);
Wall walld50 = new Wall();
addObject(walld50, 31, 7);
walld50.setLocation(31, 7);
Wall walld51 = new Wall();
addObject(walld51, 31, 6);
walld51.setLocation(31, 6);
Wall walld52 = new Wall();
addObject(walld52, 31, 5);
walld52.setLocation(31, 5);
Wall walld53 = new Wall();
addObject(walld53, 31, 4);
walld53.setLocation(31, 4);
Wall walld54 = new Wall();
addObject(walld54, 31, 3);
walld54.setLocation(31, 3);

// Upper walls
Wall wallc3 = new Wall();
addObject(wallc3, 4, 3);
```

```
wallc3.setLocation(4, 3);
Wall wallc4 = new Wall();
addObject(wallc4, 5, 3);
wallc4.setLocation(5, 3);
Wall wallc5 = new Wall();
addObject(wallc5, 6, 3);
wallc5.setLocation(6, 3);
Wall wallc6 = new Wall();
addObject(wallc6, 7, 3);
wallc6.setLocation(7, 3);
Wall wallc7 = new Wall();
addObject(wallc7, 8, 3);
wallc7.setLocation(8, 3);
Wall wallc8 = new Wall();
addObject(wallc8, 9, 3);
wallc8.setLocation(9, 3);
Wall wallc9 = new Wall();
addObject(wallc9, 10, 3);
wallc9.setLocation(10, 3);
Wall wallc10 = new Wall();
addObject(wallc10, 11, 3);
wallc10.setLocation(11, 3);
Wall wallc11 = new Wall();
addObject(wallc11, 12, 3);
wallc11.setLocation(12, 3);
Wall wallc12 = new Wall();
addObject(wallc12, 13, 3);
wallc12.setLocation(13, 3);
Wall wallc13 = new Wall();
addObject(wallc13, 14, 3);
wallc13.setLocation(14, 3);
Wall wallc14 = new Wall();
addObject(wallc14, 15, 3);
```

```
wallc14.setLocation(15, 3);
Wall wallc15 = new Wall();
addObject(wallc15, 16, 3);
wallc15.setLocation(16, 3);
Wall wallc17 = new Wall();
addObject(wallc17, 17, 3);
wallc17.setLocation(17, 3);
Wall wallc18 = new Wall();
addObject(wallc18, 17, 4);
wallc18.setLocation(17, 4);
Wall wallc19 = new Wall();
addObject(wallc19, 17, 5);
wallc19.setLocation(17, 5);
Wall wallc20 = new Wall();
addObject(wallc20, 17, 6);
wallc20.setLocation(17, 6);
Wall wallc21 = new Wall();
addObject(wallc21, 17, 7);
wallc21.setLocation(17, 7);
Wall wallc22 = new Wall();
addObject(wallc22, 18, 7);
wallc22.setLocation(18, 7);
Wall wallc23 = new Wall();
addObject(wallc23, 18, 6);
wallc23.setLocation(18, 6);
Wall wallc24 = new Wall();
addObject(wallc24, 18, 5);
wallc24.setLocation(18, 5);
Wall wallc25 = new Wall();
addObject(wallc25, 18, 4);
wallc25.setLocation(18, 4);
Wall wallc26 = new Wall();
addObject(wallc26, 18, 3);
```

```
wallc26.setLocation(18, 3);
Wall wallc27 = new Wall();
addObject(wallc27, 19, 3);
wallc27.setLocation(19, 3);
Wall wallc28 = new Wall();
addObject(wallc28, 20, 3);
wallc28.setLocation(20, 3);
Wall wallc29 = new Wall();
addObject(wallc29, 21, 3);
wallc29.setLocation(21, 3);
Wall wallc30 = new Wall();
addObject(wallc30, 22, 3);
wallc30.setLocation(22, 3);
Wall wallc31 = new Wall();
addObject(wallc31, 23, 3);
wallc31.setLocation(23, 3);
Wall wallc32 = new Wall();
addObject(wallc32, 24, 3);
wallc32.setLocation(24, 3);
Wall wallc33 = new Wall();
addObject(wallc33, 25, 3);
wallc33.setLocation(25, 3);
Wall wallc34 = new Wall();
addObject(wallc34, 26, 3);
wallc34.setLocation(26, 3);
Wall wallc35 = new Wall();
addObject(wallc35, 27, 3);
wallc35.setLocation(27, 3);
Wall wallc36 = new Wall();
addObject(wallc36, 28, 3);
wallc36.setLocation(28, 3);
Wall wallc37 = new Wall();
addObject(wallc37, 29, 3);
```

```
        //wallc37.setLocation(29, 3);
        Wall wallc38 = new Wall();
        addObject(wallc38, 30, 3);
        //wallc38.setLocation(30, 3);
    }

    // World behaviour
    private void resetCoins() {
        // Resets the coins after they've all been ate
        if (getNumberOfCoinsAte() == getNumberOfCoinsInLevel()) {
            addSmallDots();
            addBigDots();
        }
    }

    private void enemiesTeleportWalls() {
        Enemy[] enemies = getEnemies();

        for (int i = 0; i < 6; i++) {
            Enemy tempEnemy = enemies[i];

            // Left teleport
            if ((tempEnemy.getX() == 3) && (tempEnemy.getY() == 17))
{
                tempEnemy.setLocation(36 - 5, tempEnemy.getY());
            }
            // Right teleport
            else if ((tempEnemy.getX() == 36 - 4) &&
(tempEnemy.getY() == 17)) {
                tempEnemy.setLocation(4, tempEnemy.getY());
            }
        }
    }
```

```
        private void playersTeleportWalls() {
          Player[] players = getPlayers();

          for (int i = 0; i < 2; i++) {
            Player tempPlayer = players[i];

            // Left teleport
            if ((tempPlayer.getX() == 3) && (tempPlayer.getY() ==
17)) {
              tempPlayer.setLocation(36 - 5, tempPlayer.getY());
              Greenfoot.playSound("teleport.mp3");
            }
            // Right teleport
            else if ((tempPlayer.getX() == 36 - 4) &&
(tempPlayer.getY() == 17)) {
              tempPlayer.setLocation(4, tempPlayer.getY());
              Greenfoot.playSound("teleport.mp3");
            }
          }
        }

        public void act() {
          countdown(this);


          showHealthBar(this);

          showScore(this);

          returnToMenu(backgroundMusic);

          playersTeleportWalls();
```

```
        enemiesTeleportWalls();

        resetCoins();
      }
    }
    import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot
and MouseInfo)
    import java.util.List;

    public class Level2 extends Level {

      private GreenfootSound backgroundMusic = new GreenfootSound(
        "backgroundMusic.mp3"
      );

      public Level2() {
        super(System.currentTimeMillis());
        // Background
        setBackground("background.png");

        // Number of coins in level
        setNumberOfCoinsInLevel(263);

        // Players Initialization
        char[] player1MovementKeys = new char[] { 'W', 'D', 'S',
'A' };
        addPlayer(new Player(player1MovementKeys), 0);
        addObject(getPlayers()[0], 17, 21);

        String[] player2MovementKeys = new String[] {
          "up",
          "right",
```

```
  "down",
  "left",
};
addPlayer(new Player(player2MovementKeys), 1);
addObject(getPlayers()[1], 18, 21);

// Game speed
Greenfoot.setSpeed(35);

// Game music
backgroundMusic.setVolume(30);
backgroundMusic.playLoop();

//Actors
addSmallDots();
addWalls();
addIntersections();
addCorners();
addBigDots();
addTurnback();

// Enemies
addEnemy(new Enemy(0), 0);
addObject(getEnemies()[0], 17, 17);

addEnemy(new Enemy(0), 1);
addObject(getEnemies()[1], 18, 17);

addEnemy(new Enemy(0), 2);
addObject(getEnemies()[2], 17, 18);

addEnemy(new Enemy(0), 3);
```

```
        addObject(getEnemies()[3], 18, 18);

        addEnemy(new Enemy(0), 4);
        addObject(getEnemies()[4], 17, 19);

        addEnemy(new Enemy(0), 5);
        addObject(getEnemies()[5], 18, 19);
      }

      // Background Music
      protected GreenfootSound getBackgroundMusic() {
        return backgroundMusic;
      }

      // Enemy Movement
      private void addIntersections() {
        // Adds the intersection actors to the world (Used for the
enemies movement)

        Intersection inters1 = new Intersection();
        addObject(inters1, 7, 8);
        Intersection inters2 = new Intersection();
        addObject(inters2, 28, 8);
        Intersection inters3 = new Intersection();
        addObject(inters3, 16, 9);
        Intersection inters4 = new Intersection();
        addObject(inters4, 19, 9);
        Intersection inters5 = new Intersection();
        addObject(inters5, 7, 11);
        Intersection inters6 = new Intersection();
        addObject(inters6, 28, 11);
        Intersection inters7 = new Intersection();
        addObject(inters7, 13, 15);
```

```
Intersection inters8 = new Intersection();
addObject(inters8, 16, 15);
Intersection inters9 = new Intersection();
addObject(inters9, 19, 15);
Intersection inters10 = new Intersection();
addObject(inters10, 24, 15);
Intersection inters11 = new Intersection();
addObject(inters11, 8, 18);
Intersection inters12 = new Intersection();
addObject(inters12, 13, 18);
Intersection inters13 = new Intersection();
addObject(inters13, 22, 18);
Intersection inters14 = new Intersection();
addObject(inters14, 27, 18);
Intersection inters15 = new Intersection();
addObject(inters15, 5, 20);
Intersection inters16 = new Intersection();
addObject(inters16, 8, 20);
Intersection inters17 = new Intersection();
addObject(inters17, 27, 20);
Intersection inters18 = new Intersection();
addObject(inters18, 30, 20);
Intersection inters19 = new Intersection();
addObject(inters19, 13, 21);
Intersection inters20 = new Intersection();
addObject(inters20, 22, 21);
Intersection inters21 = new Intersection();
addObject(inters21, 8, 27);
Intersection inters22 = new Intersection();
addObject(inters22, 16, 27);
Intersection inters23 = new Intersection();
addObject(inters23, 19, 27);
Intersection inters24 = new Intersection();
```

```
        addObject(inters24, 27, 27);
        Intersection inters25 = new Intersection();
        addObject(inters25, 14, 4);
        Intersection inters26 = new Intersection();
        addObject(inters26, 21, 4);
        Intersection inters27 = new Intersection();
        addObject(inters27, 22, 15);
        //Intersection inters27 = new Intersection();
        //addObject(inters27, 13, 26);
        //Intersection inters28 = new Intersection();
        //addObject(inters28, 16, 26);
        //Intersection inters29 = new Intersection();
        //addObject(inters29, 19, 26);
        //Intersection inters30 = new Intersection();
        //addObject(inters30, 22, 26);
        //Intersection inters31 = new Intersection();
        //addObject(inters31, 25, 26);
        //Intersection inters32 = new Intersection();
        //addObject(inters32, 7, 29);
        //Intersection inters33 = new Intersection();
        //addObject(inters33, 28, 29);
        //Intersection inters34 = new Intersection();
        //addObject(inters34, 16, 32);
        //Intersection inters35 = new Intersection();
        //addObject(inters35, 19, 32);
    }

    private void addCorners() {
        // Adds the corner actors to the world (Used for the
enemies movement)

        Corner corner1 = new Corner();
        addObject(corner1, 11, 4);
```

```
Corner corner2 = new Corner();
addObject(corner2, 24, 4);
Corner corner3 = new Corner();
addObject(corner3, 14, 6);
Corner corner4 = new Corner();
addObject(corner4, 16, 6);
Corner corner5 = new Corner();
addObject(corner5, 19, 6);
Corner corner6 = new Corner();
addObject(corner6, 21, 6);
//Corner corner7 = new Corner();
//addObject(corner7, 5, 8);
Corner corner8 = new Corner();
addObject(corner8, 11, 8);
Corner corner9 = new Corner();
addObject(corner9, 24, 8);
//Corner corner10 = new Corner();
//addObject(corner10, 30, 8);
Corner corner11 = new Corner();
addObject(corner11, 14, 9);
Corner corner12 = new Corner();
addObject(corner12, 21, 9);

Corner corner13 = new Corner();
addObject(corner13, 5, 11);
Corner corner14 = new Corner();
addObject(corner14, 11, 11);
Corner corner15 = new Corner();
addObject(corner15, 24, 11);
Corner corner16 = new Corner();
addObject(corner16, 30, 11);
```

```
Corner corner17 = new Corner();
addObject(corner17, 14, 12);
Corner corner18 = new Corner();
addObject(corner18, 21, 12);
Corner corner37 = new Corner();
addObject(corner37, 16, 12);
Corner corner38 = new Corner();
addObject(corner38, 19, 12);

Corner corner19 = new Corner();
addObject(corner19, 5, 14);
Corner corner20 = new Corner();
addObject(corner20, 8, 14);
Corner corner21 = new Corner();
addObject(corner21, 27, 14);
Corner corner22 = new Corner();
addObject(corner22, 30, 14);

Corner corner23 = new Corner();
addObject(corner23, 11, 15);
Corner corner24 = new Corner();
addObject(corner24, 24, 15);

Corner corner25 = new Corner();
addObject(corner25, 11, 21);
Corner corner26 = new Corner();
addObject(corner26, 24, 21);

Corner corner27 = new Corner();
addObject(corner27, 11, 24);
Corner corner28 = new Corner();
addObject(corner28, 16, 24);
```

```
        Corner corner29 = new Corner();
        addObject(corner29, 19, 24);
        Corner corner30 = new Corner();
        addObject(corner30, 24, 24);

        Corner corner31 = new Corner();
        addObject(corner31, 5, 27);
        Corner corner32 = new Corner();
        addObject(corner32, 30, 27);
        Corner corner33 = new Corner();
        addObject(corner33, 5, 32);
        Corner corner34 = new Corner();
        addObject(corner34, 8, 32);
        Corner corner35 = new Corner();
        addObject(corner35, 27, 32);
        Corner corner36 = new Corner();
        addObject(corner36, 30, 32);
    }

    private void addTurnback() {
        // Adds the corner actors to the world (Used for the
enemies movement)

        TurnBack tback = new TurnBack();
        addObject(tback, 5, 8);
        TurnBack tback2 = new TurnBack();
        addObject(tback2, 30, 8);
        TurnBack tback3 = new TurnBack();
        addObject(tback3, 5, 17);
        TurnBack tback4 = new TurnBack();
        addObject(tback4, 30, 17);
        TurnBack tback5 = new TurnBack();
        addObject(tback5, 5, 24);
```

```
    TurnBack tback6 = new TurnBack();
    addObject(tback6, 30, 24);
  }

  protected void leaveTheBox(Enemy enemy) {
    // Y = 19
    if ((enemy.getX() == 17) && (enemy.getY() == 19)) {
      enemy.setMovementDirection(0);
    } else if ((enemy.getX() == 18) && (enemy.getY() == 19)) {
      enemy.setMovementDirection(0);
    }
    // Y = 18
    else if ((enemy.getX() == 17) && (enemy.getY() == 18)) {
      enemy.setMovementDirection(0);
    } else if ((enemy.getX() == 18) && (enemy.getY() == 18)) {
      enemy.setMovementDirection(0);
    }
    // Y = 17
    else if ((enemy.getX() == 17) && (enemy.getY() == 17)) {
      enemy.setMovementDirection(0);
    } else if ((enemy.getX() == 18) && (enemy.getY() == 17)) {
      enemy.setMovementDirection(0);
    }
    // Y = 16 (Door)
    // Leaves the box and goes left or right randomly
    else if ((enemy.getX() == 17) && (enemy.getY() == 16)) {
      enemy.setLocation(enemy.getX(), enemy.getY() - 1);

      if (Greenfoot.getRandomNumber(2) == 0) {
        enemy.setMovementDirection(1);
      } else {
        enemy.setMovementDirection(3);
```

```
    }
  } else if ((enemy.getX() == 18) && (enemy.getY() == 16)) {
    enemy.setLocation(enemy.getX(), enemy.getY() - 1);
    if (Greenfoot.getRandomNumber(2) == 0) {
      enemy.setMovementDirection(1);
    } else {
      enemy.setMovementDirection(3);
    }
  }
}

// World building
private void addBigDots() {
  // Adds the big dot actors to the world

  BigDot BigDot1 = new BigDot();
  addObject(BigDot1, 5, 14);
  BigDot BigDot2 = new BigDot();
  addObject(BigDot2, 30, 14);
  BigDot BigDot3 = new BigDot();
  addObject(BigDot3, 30, 27);
  BigDot BigDot4 = new BigDot();
  addObject(BigDot4, 5, 27);
}

private void addSmallDots() {
  // Adds the small dot actors to the world

  //Top

  SmallDot smallDot1 = new SmallDot();
  addObject(smallDot1, 11, 4);
```

```
SmallDot smallDot2 = new SmallDot();
addObject(smallDot2, 12, 4);
SmallDot smallDot3 = new SmallDot();
addObject(smallDot3, 13, 4);
SmallDot smallDot5 = new SmallDot();
addObject(smallDot5, 14, 4);
SmallDot smallDot6 = new SmallDot();
addObject(smallDot6, 15, 4);
SmallDot smallDot7 = new SmallDot();
addObject(smallDot7, 16, 4);
SmallDot smallDot8 = new SmallDot();
addObject(smallDot8, 17, 4);
SmallDot smallDot9 = new SmallDot();
addObject(smallDot9, 18, 4);
SmallDot smallDot10 = new SmallDot();
addObject(smallDot10, 19, 4);
SmallDot smallDot11 = new SmallDot();
addObject(smallDot11, 20, 4);
SmallDot smallDot12 = new SmallDot();
addObject(smallDot12, 21, 4);
SmallDot smallDot13 = new SmallDot();
addObject(smallDot13, 22, 4);
SmallDot smallDot14 = new SmallDot();
addObject(smallDot14, 23, 4);
SmallDot smallDot15 = new SmallDot();
addObject(smallDot15, 24, 4);
SmallDot smallDot16 = new SmallDot();
addObject(smallDot16, 11, 5);
SmallDot smallDot17 = new SmallDot();
addObject(smallDot17, 14, 5);
SmallDot smallDot19 = new SmallDot();
addObject(smallDot19, 21, 5);
SmallDot smallDot20 = new SmallDot();
```

```
addObject(smallDot20, 24, 5);
SmallDot smallDot21 = new SmallDot();
addObject(smallDot21, 11, 6);
SmallDot smallDot22 = new SmallDot();
addObject(smallDot22, 14, 6);
SmallDot smallDot23 = new SmallDot();
addObject(smallDot23, 15, 6);
SmallDot smallDot24 = new SmallDot();
addObject(smallDot24, 16, 6);
SmallDot smallDot25 = new SmallDot();
addObject(smallDot25, 19, 6);
SmallDot smallDot26 = new SmallDot();
addObject(smallDot26, 20, 6);
SmallDot smallDot27 = new SmallDot();
addObject(smallDot27, 21, 6);
SmallDot smallDot28 = new SmallDot();
addObject(smallDot28, 24, 6);
SmallDot smallDot35 = new SmallDot();
addObject(smallDot35, 11, 7);
SmallDot smallDot36 = new SmallDot();
addObject(smallDot36, 16, 7);
SmallDot smallDot37 = new SmallDot();
addObject(smallDot37, 19, 7);
SmallDot smallDot38 = new SmallDot();
addObject(smallDot38, 24, 7);
SmallDot smallDot39 = new SmallDot();
addObject(smallDot39, 5, 8);
SmallDot smallDot40 = new SmallDot();
addObject(smallDot40, 6, 8);
SmallDot smallDot57 = new SmallDot();
addObject(smallDot57, 7, 8);
SmallDot smallDot58 = new SmallDot();
addObject(smallDot58, 8, 8);
```

```
SmallDot smallDot59 = new SmallDot();
addObject(smallDot59, 9, 8);
SmallDot smallDot60 = new SmallDot();
addObject(smallDot60, 10, 8);
SmallDot smallDot61 = new SmallDot();
addObject(smallDot61, 11, 8);
SmallDot smallDot62 = new SmallDot();
addObject(smallDot62, 16, 8);
SmallDot smallDot63 = new SmallDot();
addObject(smallDot63, 19, 8);
SmallDot smallDot64 = new SmallDot();
addObject(smallDot64, 24, 8);
SmallDot smallDot65 = new SmallDot();
addObject(smallDot65, 25, 8);
SmallDot smallDot66 = new SmallDot();
addObject(smallDot66, 26, 8);
SmallDot smallDot67 = new SmallDot();
addObject(smallDot67, 27, 8);
SmallDot smallDot68 = new SmallDot();
addObject(smallDot68, 28, 8);
SmallDot smallDot69 = new SmallDot();
addObject(smallDot69, 29, 8);
SmallDot smallDot70 = new SmallDot();
addObject(smallDot70, 30, 8);
SmallDot smallDot71 = new SmallDot();
addObject(smallDot71, 7, 9);
SmallDot smallDot72 = new SmallDot();
addObject(smallDot72, 14, 9);
SmallDot smallDot73 = new SmallDot();
addObject(smallDot73, 15, 9);
SmallDot smallDot74 = new SmallDot();
addObject(smallDot74, 16, 9);
SmallDot smallDot75 = new SmallDot();
```

```
addObject(smallDot75, 17, 9);
SmallDot smallDot76 = new SmallDot();
addObject(smallDot76, 18, 9);
SmallDot smallDot77 = new SmallDot();
addObject(smallDot77, 19, 9);
SmallDot smallDot78 = new SmallDot();
addObject(smallDot78, 20, 9);
SmallDot smallDot79 = new SmallDot();
addObject(smallDot79, 21, 9);
SmallDot smallDot80 = new SmallDot();
addObject(smallDot80, 28, 9);
SmallDot smallDot81 = new SmallDot();
addObject(smallDot81, 7, 10);
SmallDot smallDot82 = new SmallDot();
addObject(smallDot82, 10, 10);
SmallDot smallDot83 = new SmallDot();
addObject(smallDot83, 14, 10);
SmallDot smallDot84 = new SmallDot();
addObject(smallDot84, 21, 10);
SmallDot smallDot85 = new SmallDot();
addObject(smallDot85, 28, 10);
SmallDot smallDot86 = new SmallDot();
addObject(smallDot86, 5, 11);
SmallDot smallDot87 = new SmallDot();
addObject(smallDot87, 6, 11);
SmallDot smallDot88 = new SmallDot();
addObject(smallDot88, 7, 11);
SmallDot smallDot89 = new SmallDot();
addObject(smallDot89, 8, 11);
SmallDot smallDot90 = new SmallDot();
addObject(smallDot90, 9, 11);
SmallDot smallDot91 = new SmallDot();
addObject(smallDot91, 10, 11);
```

```
SmallDot smallDot92 = new SmallDot();
addObject(smallDot92, 10, 11);
SmallDot smallDot93 = new SmallDot();
addObject(smallDot93, 11, 11);
SmallDot smallDot94 = new SmallDot();
addObject(smallDot94, 14, 11);
SmallDot smallDot95 = new SmallDot();
addObject(smallDot95, 21, 11);
SmallDot smallDot96 = new SmallDot();
addObject(smallDot96, 24, 11);
SmallDot smallDot97 = new SmallDot();
addObject(smallDot97, 25, 11);
SmallDot smallDot98 = new SmallDot();
addObject(smallDot98, 26, 11);
SmallDot smallDot99 = new SmallDot();
addObject(smallDot99, 27, 11);
SmallDot smallDot100 = new SmallDot();
addObject(smallDot100, 28, 11);
SmallDot smallDot101 = new SmallDot();
addObject(smallDot101, 29, 11);
SmallDot smallDot102 = new SmallDot();
addObject(smallDot102, 30, 11);
SmallDot smallDot103 = new SmallDot();
addObject(smallDot103, 5, 12);
SmallDot smallDot104 = new SmallDot();
addObject(smallDot104, 11, 12);
SmallDot smallDot105 = new SmallDot();
addObject(smallDot105, 14, 12);
SmallDot smallDot106 = new SmallDot();
addObject(smallDot106, 15, 12);
SmallDot smallDot107 = new SmallDot();
addObject(smallDot107, 16, 12);
SmallDot smallDot108 = new SmallDot();
```

```
addObject(smallDot108, 19, 12);
SmallDot smallDot109 = new SmallDot();
addObject(smallDot109, 20, 12);
SmallDot smallDot110 = new SmallDot();
addObject(smallDot110, 21, 12);
SmallDot smallDot111 = new SmallDot();
addObject(smallDot111, 24, 12);
SmallDot smallDot112 = new SmallDot();
addObject(smallDot112, 30, 12);
SmallDot smallDot113 = new SmallDot();
addObject(smallDot113, 5, 13);
SmallDot smallDot114 = new SmallDot();
addObject(smallDot114, 11, 13);
SmallDot smallDot115 = new SmallDot();
addObject(smallDot115, 10, 13);
SmallDot smallDot116 = new SmallDot();
addObject(smallDot116, 13, 13);
SmallDot smallDot117 = new SmallDot();
addObject(smallDot117, 16, 13);
SmallDot smallDot118 = new SmallDot();
addObject(smallDot118, 19, 13);
SmallDot smallDot120 = new SmallDot();
addObject(smallDot120, 24, 13);
SmallDot smallDot121 = new SmallDot();
addObject(smallDot121, 30, 13);
//SmallDot smallDot122 = new SmallDot();
//addObject(smallDot122, 5, 14);
SmallDot smallDot123 = new SmallDot();
addObject(smallDot123, 6, 14);
SmallDot smallDot124 = new SmallDot();
addObject(smallDot124, 7, 14);
SmallDot smallDot125 = new SmallDot();
addObject(smallDot125, 8, 14);
```

```
SmallDot smallDot126 = new SmallDot();
addObject(smallDot126, 11, 14);
SmallDot smallDot127 = new SmallDot();
addObject(smallDot127, 16, 14);
SmallDot smallDot128 = new SmallDot();
addObject(smallDot128, 19, 14);
SmallDot smallDot129 = new SmallDot();
addObject(smallDot129, 24, 14);
SmallDot smallDot130 = new SmallDot();
addObject(smallDot130, 27, 14);
SmallDot smallDot131 = new SmallDot();
addObject(smallDot131, 28, 14);
SmallDot smallDot132 = new SmallDot();
addObject(smallDot132, 29, 14);
//SmallDot smallDot133 = new SmallDot();
//addObject(smallDot133, 30, 14);
SmallDot smallDot134 = new SmallDot();
addObject(smallDot134, 8, 15);
SmallDot smallDot135 = new SmallDot();
addObject(smallDot135, 11, 15);
SmallDot smallDot136 = new SmallDot();
addObject(smallDot136, 12, 15);
SmallDot smallDot137 = new SmallDot();
addObject(smallDot137, 13, 15);
SmallDot smallDot138 = new SmallDot();
addObject(smallDot138, 14, 15);
SmallDot smallDot139 = new SmallDot();
addObject(smallDot139, 15, 15);
SmallDot smallDot140 = new SmallDot();
addObject(smallDot140, 16, 15);
SmallDot smallDot141 = new SmallDot();
addObject(smallDot141, 17, 15);
SmallDot smallDot142 = new SmallDot();
```

```
addObject(smallDot142, 18, 15);
SmallDot smallDot143 = new SmallDot();
addObject(smallDot143, 19, 15);
SmallDot smallDot144 = new SmallDot();
addObject(smallDot144, 20, 15);
SmallDot smallDot145 = new SmallDot();
addObject(smallDot145, 21, 15);
SmallDot smallDot146 = new SmallDot();
addObject(smallDot146, 22, 15);
SmallDot smallDot147 = new SmallDot();
addObject(smallDot147, 23, 15);
SmallDot smallDot148 = new SmallDot();
addObject(smallDot148, 24, 15);
SmallDot smallDot149 = new SmallDot();
addObject(smallDot149, 27, 15);
SmallDot smallDot150 = new SmallDot();
addObject(smallDot150, 8, 16);
SmallDot smallDot151 = new SmallDot();
addObject(smallDot151, 13, 16);
SmallDot smallDot152 = new SmallDot();
addObject(smallDot152, 22, 16);
SmallDot smallDot153 = new SmallDot();
addObject(smallDot153, 27, 16);
SmallDot smallDot154 = new SmallDot();
addObject(smallDot154, 5, 17);
SmallDot smallDot155 = new SmallDot();
addObject(smallDot155, 8, 17);
SmallDot smallDot156 = new SmallDot();
addObject(smallDot156, 13, 17);
SmallDot smallDot157 = new SmallDot();
addObject(smallDot157, 22, 17);
SmallDot smallDot158 = new SmallDot();
addObject(smallDot158, 27, 17);
```

```
SmallDot smallDot159 = new SmallDot();
addObject(smallDot159, 30, 17);
SmallDot smallDot160 = new SmallDot();
addObject(smallDot160, 5, 18);
SmallDot smallDot161 = new SmallDot();
addObject(smallDot161, 8, 18);
SmallDot smallDot162 = new SmallDot();
addObject(smallDot162, 9, 18);
SmallDot smallDot163 = new SmallDot();
addObject(smallDot163, 10, 18);
SmallDot smallDot164 = new SmallDot();
addObject(smallDot164, 11, 18);
SmallDot smallDot165 = new SmallDot();
addObject(smallDot165, 12, 18);
SmallDot smallDot166 = new SmallDot();
addObject(smallDot166, 13, 18);
SmallDot smallDot167 = new SmallDot();
addObject(smallDot167, 22, 18);
SmallDot smallDot168 = new SmallDot();
addObject(smallDot168, 23, 18);
SmallDot smallDot169 = new SmallDot();
addObject(smallDot169, 24, 18);
SmallDot smallDot170 = new SmallDot();
addObject(smallDot170, 25, 18);
SmallDot smallDot171 = new SmallDot();
addObject(smallDot171, 26, 18);
SmallDot smallDot172 = new SmallDot();
addObject(smallDot172, 27, 18);
SmallDot smallDot173 = new SmallDot();
addObject(smallDot173, 30, 18);
SmallDot smallDot174 = new SmallDot();
addObject(smallDot174, 5, 19);
SmallDot smallDot175 = new SmallDot();
```

```
addObject(smallDot175, 8, 19);
SmallDot smallDot176 = new SmallDot();
addObject(smallDot176, 13, 19);
SmallDot smallDot177 = new SmallDot();
addObject(smallDot177, 22, 19);
SmallDot smallDot178 = new SmallDot();
addObject(smallDot178, 27, 19);
SmallDot smallDot179 = new SmallDot();
addObject(smallDot179, 30, 19);
SmallDot smallDot180 = new SmallDot();
addObject(smallDot180, 5, 20);
SmallDot smallDot181 = new SmallDot();
addObject(smallDot181, 6, 20);
SmallDot smallDot182 = new SmallDot();
addObject(smallDot182, 7, 20);
SmallDot smallDot183 = new SmallDot();
addObject(smallDot183, 8, 20);
SmallDot smallDot184 = new SmallDot();
addObject(smallDot184, 13, 20);
SmallDot smallDot185 = new SmallDot();
addObject(smallDot185, 22, 20);
SmallDot smallDot186 = new SmallDot();
addObject(smallDot186, 27, 20);
SmallDot smallDot187 = new SmallDot();
addObject(smallDot187, 28, 20);
SmallDot smallDot188 = new SmallDot();
addObject(smallDot188, 29, 20);
SmallDot smallDot189 = new SmallDot();
addObject(smallDot189, 30, 20);
SmallDot smallDot190 = new SmallDot();
addObject(smallDot190, 5, 21);
SmallDot smallDot191 = new SmallDot();
addObject(smallDot191, 8, 21);
```

```
SmallDot smallDot192 = new SmallDot();
addObject(smallDot192, 11, 21);
SmallDot smallDot193 = new SmallDot();
addObject(smallDot193, 12, 21);
SmallDot smallDot194 = new SmallDot();
addObject(smallDot194, 13, 21);
SmallDot smallDot195 = new SmallDot();
addObject(smallDot195, 14, 21);
SmallDot smallDot196 = new SmallDot();
addObject(smallDot196, 15, 21);
SmallDot smallDot197 = new SmallDot();
addObject(smallDot197, 16, 21);
//SmallDot smallDot198 = new SmallDot();
//addObject(smallDot198, 17, 21);
//SmallDot smallDot199 = new SmallDot();
//addObject(smallDot199, 18, 21);
SmallDot smallDot200 = new SmallDot();
addObject(smallDot200, 19, 21);
SmallDot smallDot201 = new SmallDot();
addObject(smallDot201, 20, 21);
SmallDot smallDot202 = new SmallDot();
addObject(smallDot202, 21, 21);
SmallDot smallDot203 = new SmallDot();
addObject(smallDot203, 22, 21);
SmallDot smallDot204 = new SmallDot();
addObject(smallDot204, 23, 21);
SmallDot smallDot205 = new SmallDot();
addObject(smallDot205, 24, 21);
SmallDot smallDot206 = new SmallDot();
addObject(smallDot206, 27, 21);
SmallDot smallDot207 = new SmallDot();
addObject(smallDot207, 30, 21);
SmallDot smallDot209 = new SmallDot();
```

```
addObject(smallDot209, 5, 22);
SmallDot smallDot210 = new SmallDot();
addObject(smallDot210, 8, 22);
SmallDot smallDot211 = new SmallDot();
addObject(smallDot211, 11, 22);
SmallDot smallDot212 = new SmallDot();
addObject(smallDot212, 24, 22);
SmallDot smallDot213 = new SmallDot();
addObject(smallDot213, 27, 22);
SmallDot smallDot214 = new SmallDot();
addObject(smallDot214, 30, 22);
SmallDot smallDot215 = new SmallDot();
addObject(smallDot215, 5, 23);
SmallDot smallDot216 = new SmallDot();
addObject(smallDot216, 8, 23);
SmallDot smallDot217 = new SmallDot();
addObject(smallDot217, 11, 23);
SmallDot smallDot220 = new SmallDot();
addObject(smallDot220, 24, 23);
SmallDot smallDot221 = new SmallDot();
addObject(smallDot221, 27, 23);
SmallDot smallDot222 = new SmallDot();
addObject(smallDot222, 30, 23);
SmallDot smallDot223 = new SmallDot();
addObject(smallDot223, 5, 24);
SmallDot smallDot224 = new SmallDot();
addObject(smallDot224, 8, 24);
SmallDot smallDot225 = new SmallDot();
addObject(smallDot225, 11, 24);
SmallDot smallDot226 = new SmallDot();
addObject(smallDot226, 12, 24);
SmallDot smallDot229 = new SmallDot();
addObject(smallDot229, 13, 24);
```

```
SmallDot smallDot230 = new SmallDot();
addObject(smallDot230, 14, 24);
SmallDot smallDot233 = new SmallDot();
addObject(smallDot233, 15, 24);
SmallDot smallDot234 = new SmallDot();
addObject(smallDot234, 16, 24);
SmallDot smallDot235 = new SmallDot();
addObject(smallDot235, 19, 24);
SmallDot smallDot236 = new SmallDot();
addObject(smallDot236, 20, 24);
SmallDot smallDot237 = new SmallDot();
addObject(smallDot237, 21, 24);
SmallDot smallDot238 = new SmallDot();
addObject(smallDot238, 22, 24);
SmallDot smallDot239 = new SmallDot();
addObject(smallDot239, 23, 24);
SmallDot smallDot240 = new SmallDot();
addObject(smallDot240, 24, 24);
SmallDot smallDot241 = new SmallDot();
addObject(smallDot241, 27, 24);
SmallDot smallDot242 = new SmallDot();
addObject(smallDot242, 30, 24);
SmallDot smallDot243 = new SmallDot();
addObject(smallDot243, 8, 25);
SmallDot smallDot244 = new SmallDot();
addObject(smallDot244, 16, 25);
SmallDot smallDot245 = new SmallDot();
addObject(smallDot245, 19, 25);
SmallDot smallDot246 = new SmallDot();
addObject(smallDot246, 27, 25);
SmallDot smallDot247 = new SmallDot();
addObject(smallDot247, 8, 26);
SmallDot smallDot248 = new SmallDot();
```

```
addObject(smallDot248, 16, 26);
SmallDot smallDot249 = new SmallDot();
addObject(smallDot249, 19, 26);
SmallDot smallDot250 = new SmallDot();
addObject(smallDot250, 27, 26);
//SmallDot smallDot251 = new SmallDot();
//addObject(smallDot251, 5, 27);
SmallDot smallDot252 = new SmallDot();
addObject(smallDot252, 6, 27);
SmallDot smallDot253 = new SmallDot();
addObject(smallDot253, 7, 27);
SmallDot smallDot254 = new SmallDot();
addObject(smallDot254, 8, 27);
SmallDot smallDot255 = new SmallDot();
addObject(smallDot255, 9, 27);
SmallDot smallDot256 = new SmallDot();
addObject(smallDot256, 10, 27);
SmallDot smallDot257 = new SmallDot();
addObject(smallDot257, 11, 27);
SmallDot smallDot258 = new SmallDot();
addObject(smallDot258, 12, 27);
SmallDot smallDot259 = new SmallDot();
addObject(smallDot259, 13, 27);
SmallDot smallDot260 = new SmallDot();
addObject(smallDot260, 14, 27);
SmallDot smallDot261 = new SmallDot();
addObject(smallDot261, 15, 27);
SmallDot smallDot262 = new SmallDot();
addObject(smallDot262, 16, 27);
SmallDot smallDot263 = new SmallDot();
addObject(smallDot263, 17, 27);
SmallDot smallDot264 = new SmallDot();
addObject(smallDot264, 18, 27);
```

```
SmallDot smallDot265 = new SmallDot();
addObject(smallDot265, 19, 27);
SmallDot smallDot266 = new SmallDot();
addObject(smallDot266, 20, 27);
SmallDot smallDot267 = new SmallDot();
addObject(smallDot267, 21, 27);
SmallDot smallDot268 = new SmallDot();
addObject(smallDot268, 22, 27);
SmallDot smallDot269 = new SmallDot();
addObject(smallDot269, 23, 27);
SmallDot smallDot270 = new SmallDot();
addObject(smallDot270, 24, 27);
SmallDot smallDot271 = new SmallDot();
addObject(smallDot271, 25, 27);
SmallDot smallDot272 = new SmallDot();
addObject(smallDot272, 26, 27);
SmallDot smallDot273 = new SmallDot();
addObject(smallDot273, 27, 27);
SmallDot smallDot274 = new SmallDot();
addObject(smallDot274, 28, 27);
SmallDot smallDot275 = new SmallDot();
addObject(smallDot275, 29, 27);
//SmallDot smallDot276 = new SmallDot();
//addObject(smallDot276, 30, 27);
SmallDot smallDot277 = new SmallDot();
addObject(smallDot277, 5, 28);
SmallDot smallDot278 = new SmallDot();
addObject(smallDot278, 8, 28);
SmallDot smallDot279 = new SmallDot();
addObject(smallDot279, 27, 28);
SmallDot smallDot280 = new SmallDot();
addObject(smallDot280, 30, 28);
SmallDot smallDot281 = new SmallDot();
```

```
addObject(smallDot281, 5, 29);
SmallDot smallDot282 = new SmallDot();
addObject(smallDot282, 8, 29);
SmallDot smallDot283 = new SmallDot();
addObject(smallDot283, 27, 29);
SmallDot smallDot284 = new SmallDot();
addObject(smallDot284, 30, 29);
SmallDot smallDot285 = new SmallDot();
addObject(smallDot285, 5, 30);
SmallDot smallDot286 = new SmallDot();
addObject(smallDot286, 8, 30);
SmallDot smallDot287 = new SmallDot();
addObject(smallDot287, 27, 30);
SmallDot smallDot288 = new SmallDot();
addObject(smallDot288, 30, 30);
SmallDot smallDot289 = new SmallDot();
addObject(smallDot289, 5, 31);
SmallDot smallDot290 = new SmallDot();
addObject(smallDot290, 8, 31);
SmallDot smallDot291 = new SmallDot();
addObject(smallDot291, 27, 31);
SmallDot smallDot292 = new SmallDot();
addObject(smallDot292, 30, 31);
SmallDot smallDot293 = new SmallDot();
addObject(smallDot293, 5, 32);
SmallDot smallDot294 = new SmallDot();
addObject(smallDot294, 6, 32);
SmallDot smallDot295 = new SmallDot();
addObject(smallDot295, 7, 32);
SmallDot smallDot296 = new SmallDot();
addObject(smallDot296, 8, 32);
SmallDot smallDot297 = new SmallDot();
addObject(smallDot297, 27, 32);
```

```
    SmallDot smallDot298 = new SmallDot();
    addObject(smallDot298, 28, 32);
    SmallDot smallDot299 = new SmallDot();
    addObject(smallDot299, 29, 32);
    SmallDot smallDot300 = new SmallDot();
    addObject(smallDot300, 30, 32);
  }

  private void addWalls() {
    // Adds the level walls to the world

    // Walls are added from top to botom
    // min 4,3
    // top part
    Wall wall1 = new Wall();
    addObject(wall1, 10, 3);
    Wall wall2 = new Wall();
    addObject(wall2, 11, 3);
    Wall wall3 = new Wall();
    addObject(wall3, 12, 3);
    Wall wall4 = new Wall();
    addObject(wall4, 13, 3);
    Wall wall5 = new Wall();
    addObject(wall5, 14, 3);
    Wall wall6 = new Wall();
    addObject(wall6, 15, 3);
    Wall wall7 = new Wall();
    addObject(wall7, 16, 3);
    Wall wall8 = new Wall();
    addObject(wall8, 17, 3);
    Wall wall9 = new Wall();
    addObject(wall9, 18, 3);
```

```
Wall wall10 = new Wall();
addObject(wall10, 19, 3);
Wall wall11 = new Wall();
addObject(wall11, 20, 3);
Wall wall12 = new Wall();
addObject(wall12, 21, 3);
Wall wall13 = new Wall();
addObject(wall13, 22, 3);
Wall wall14 = new Wall();
addObject(wall14, 23, 3);
Wall wall15 = new Wall();
addObject(wall15, 24, 3);
Wall wall16 = new Wall();
addObject(wall16, 25, 3);
Wall wall17 = new Wall();
addObject(wall17, 10, 4);
Wall wall18 = new Wall();
addObject(wall18, 25, 4);
Wall wall19 = new Wall();
addObject(wall19, 10, 5);
Wall wall20 = new Wall();
addObject(wall20, 12, 5);
Wall wall21 = new Wall();
addObject(wall21, 13, 5);
Wall wall22 = new Wall();
addObject(wall22, 15, 5);
Wall wall23 = new Wall();
addObject(wall23, 16, 5);
Wall wall24 = new Wall();
addObject(wall24, 17, 5);
Wall wall25 = new Wall();
addObject(wall25, 18, 5);
Wall wall26 = new Wall();
```

```
addObject(wall26, 19, 5);
Wall wall27 = new Wall();
addObject(wall27, 20, 5);
Wall wall28 = new Wall();
addObject(wall28, 22, 5);
Wall wall29 = new Wall();
addObject(wall29, 23, 5);
Wall wall30 = new Wall();
addObject(wall30, 25, 5);
Wall wall31 = new Wall();
addObject(wall31, 10, 6);
Wall wall32 = new Wall();
addObject(wall32, 12, 6);
Wall wall33 = new Wall();
addObject(wall33, 13, 6);
Wall wall34 = new Wall();
addObject(wall34, 17, 6);
Wall wall35 = new Wall();
addObject(wall35, 18, 6);
Wall wall36 = new Wall();
addObject(wall36, 22, 6);
Wall wall37 = new Wall();
addObject(wall37, 23, 6);
Wall wall38 = new Wall();
addObject(wall38, 25, 6);
Wall wall39 = new Wall();
addObject(wall39, 4, 7);
Wall wall40 = new Wall();
addObject(wall40, 5, 7);
Wall wall41 = new Wall();
addObject(wall41, 6, 7);
Wall wall42 = new Wall();
addObject(wall42, 7, 7);
```

```
Wall wall43 = new Wall();
addObject(wall43, 8, 7);
Wall wall44 = new Wall();
addObject(wall44, 9, 7);
Wall wall45 = new Wall();
addObject(wall45, 10, 7);
Wall wall46 = new Wall();
addObject(wall46, 12, 7);
Wall wall47 = new Wall();
addObject(wall47, 13, 7);
Wall wall48 = new Wall();
addObject(wall48, 14, 7);
Wall wall49 = new Wall();
addObject(wall49, 15, 7);
Wall wall50 = new Wall();
addObject(wall50, 17, 7);
Wall wall51 = new Wall();
addObject(wall51, 18, 7);
Wall wall52 = new Wall();
addObject(wall52, 20, 7);
Wall wall53 = new Wall();
addObject(wall53, 21, 7);
Wall wall54 = new Wall();
addObject(wall54, 22, 7);
Wall wall55 = new Wall();
addObject(wall55, 23, 7);
Wall wall56 = new Wall();
addObject(wall56, 25, 7);
Wall wall57 = new Wall();
addObject(wall57, 4, 8);
Wall wall58 = new Wall();
addObject(wall58, 12, 8);
Wall wall59 = new Wall();
```

```
addObject(wall59, 13, 8);
Wall wall60 = new Wall();
addObject(wall60, 14, 8);
Wall wall61 = new Wall();
addObject(wall61, 15, 8);
Wall wall62 = new Wall();
addObject(wall62, 17, 8);
Wall wall63 = new Wall();
addObject(wall63, 18, 8);
Wall wall64 = new Wall();
addObject(wall64, 20, 8);
Wall wall65 = new Wall();
addObject(wall65, 21, 8);
Wall wall66 = new Wall();
addObject(wall66, 22, 8);
Wall wall67 = new Wall();
addObject(wall67, 23, 8);
Wall wall99 = new Wall();
addObject(wall99, 31, 8);
Wall wall91 = new Wall();
addObject(wall91, 25, 7);
Wall wall92 = new Wall();
addObject(wall92, 26, 7);
Wall wall93 = new Wall();
addObject(wall93, 27, 7);
Wall wall94 = new Wall();
addObject(wall94, 28, 7);
Wall wall95 = new Wall();
addObject(wall95, 29, 7);
Wall wall96 = new Wall();
addObject(wall96, 30, 7);
Wall wall97 = new Wall();
addObject(wall97, 31, 7);
```

```
Wall wall69 = new Wall();
addObject(wall69, 4, 9);
Wall wall70 = new Wall();
addObject(wall70, 5, 9);
Wall wall71 = new Wall();
addObject(wall71, 6, 9);
Wall wall72 = new Wall();
addObject(wall72, 8, 9);
Wall wall73 = new Wall();
addObject(wall73, 9, 9);
Wall wall74 = new Wall();
addObject(wall74, 10, 9);
Wall wall75 = new Wall();
addObject(wall75, 11, 9);
Wall wall76 = new Wall();
addObject(wall76, 12, 9);
Wall wall77 = new Wall();
addObject(wall77, 13, 9);
Wall wall78 = new Wall();
addObject(wall78, 22, 9);
Wall wall79 = new Wall();
addObject(wall79, 23, 9);
Wall wall80 = new Wall();
addObject(wall80, 24, 9);
Wall wall81 = new Wall();
addObject(wall81, 25, 9);
Wall wall82 = new Wall();
addObject(wall82, 26, 9);
Wall wall83 = new Wall();
addObject(wall83, 27, 9);
Wall wall84 = new Wall();
addObject(wall84, 29, 9);
Wall wall85 = new Wall();
```

```
addObject(wall85, 30, 9);
Wall wall86 = new Wall();
addObject(wall86, 31, 9);
Wall wall88 = new Wall();
addObject(wall88, 4, 10);
Wall wall89 = new Wall();
addObject(wall89, 5, 10);
Wall wall90 = new Wall();
addObject(wall90, 6, 10);
Wall wall100 = new Wall();
addObject(wall100, 8, 10);
Wall wall101 = new Wall();
addObject(wall101, 9, 10);
Wall wall102 = new Wall();
addObject(wall102, 10, 10);
Wall wall103 = new Wall();
addObject(wall103, 11, 10);
Wall wall104 = new Wall();
addObject(wall104, 12, 10);
Wall wall105 = new Wall();
addObject(wall105, 13, 10);
Wall wall106 = new Wall();
addObject(wall106, 15, 10);
Wall wall107 = new Wall();
addObject(wall107, 16, 10);
Wall wall108 = new Wall();
addObject(wall108, 17, 10);
Wall wall109 = new Wall();
addObject(wall109, 18, 10);
Wall wall110 = new Wall();
addObject(wall110, 19, 10);
Wall wall111 = new Wall();
addObject(wall111, 20, 10);
```

```
Wall wall112 = new Wall();
addObject(wall112, 22, 10);
Wall wall113 = new Wall();
addObject(wall113, 23, 10);
Wall wall114 = new Wall();
addObject(wall114, 24, 10);
Wall wall115 = new Wall();
addObject(wall115, 25, 10);
Wall wall116 = new Wall();
addObject(wall116, 26, 10);
Wall wall117 = new Wall();
addObject(wall117, 27, 10);
Wall wall118 = new Wall();
addObject(wall118, 29, 10);
Wall wall119 = new Wall();
addObject(wall119, 30, 10);
Wall wall120 = new Wall();
addObject(wall120, 31, 10);
Wall wall122 = new Wall();
addObject(wall122, 4, 11);
Wall wall123 = new Wall();
addObject(wall123, 12, 11);
Wall wall124 = new Wall();
addObject(wall124, 13, 11);
Wall wall125 = new Wall();
addObject(wall125, 15, 11);
Wall wall126 = new Wall();
addObject(wall126, 16, 11);
Wall wall127 = new Wall();
addObject(wall127, 17, 11);
Wall wall128 = new Wall();
addObject(wall128, 18, 11);
Wall wall129 = new Wall();
```

```
addObject(wall129, 19, 11);
Wall wall130 = new Wall();
addObject(wall130, 20, 11);
Wall wall131 = new Wall();
addObject(wall131, 22, 11);
Wall wall132 = new Wall();
addObject(wall132, 23, 11);
Wall wall133 = new Wall();
addObject(wall133, 31, 11);
Wall wall134 = new Wall();
addObject(wall134, 4, 12);
Wall wall135 = new Wall();
addObject(wall135, 6, 12);
Wall wall136 = new Wall();
addObject(wall136, 7, 12);
Wall wall137 = new Wall();
addObject(wall137, 8, 12);
Wall wall138 = new Wall();
addObject(wall138, 9, 12);
Wall wall139 = new Wall();
addObject(wall139, 10, 12);
Wall wall140 = new Wall();
addObject(wall140, 12, 12);
Wall wall141 = new Wall();
addObject(wall141, 13, 12);
Wall wall142 = new Wall();
addObject(wall142, 17, 12);
Wall wall143 = new Wall();
addObject(wall143, 18, 12);
Wall wall144 = new Wall();
addObject(wall144, 22, 12);
Wall wall145 = new Wall();
addObject(wall145, 23, 12);
```

```
Wall wall146 = new Wall();
addObject(wall146, 25, 12);
Wall wall147 = new Wall();
addObject(wall147, 26, 12);
Wall wall148 = new Wall();
addObject(wall148, 27, 12);
Wall wall149 = new Wall();
addObject(wall149, 28, 12);
Wall wall150 = new Wall();
addObject(wall150, 29, 12);
Wall wall151 = new Wall();
addObject(wall151, 31, 12);
Wall wall153 = new Wall();
addObject(wall153, 4, 13);
Wall wall154 = new Wall();
addObject(wall154, 6, 13);
Wall wall155 = new Wall();
addObject(wall155, 7, 13);
Wall wall156 = new Wall();
addObject(wall156, 8, 13);
Wall wall157 = new Wall();
addObject(wall157, 9, 13);
Wall wall158 = new Wall();
addObject(wall158, 10, 13);
Wall wall159 = new Wall();
addObject(wall159, 12, 13);
Wall wall160 = new Wall();
addObject(wall160, 13, 13);
Wall wall161 = new Wall();
addObject(wall161, 14, 13);
Wall wall162 = new Wall();
addObject(wall162, 15, 13);
Wall wall163 = new Wall();
```

```
addObject(wall163, 17, 13);
Wall wall164 = new Wall();
addObject(wall164, 18, 13);
Wall wall165 = new Wall();
addObject(wall165, 20, 13);
Wall wall166 = new Wall();
addObject(wall166, 21, 13);
Wall wall167 = new Wall();
addObject(wall167, 22, 13);
Wall wall168 = new Wall();
addObject(wall168, 23, 13);
Wall wall169 = new Wall();
addObject(wall169, 25, 13);
Wall wall170 = new Wall();
addObject(wall170, 26, 13);
Wall wall171 = new Wall();
addObject(wall171, 27, 13);
Wall wall172 = new Wall();
addObject(wall172, 28, 13);
Wall wall173 = new Wall();
addObject(wall173, 29, 13);
Wall wall174 = new Wall();
addObject(wall174, 31, 13);
Wall wall176 = new Wall();
addObject(wall176, 4, 14);
Wall wall177 = new Wall();
addObject(wall177, 9, 14);
Wall wall178 = new Wall();
addObject(wall178, 10, 14);
Wall wall179 = new Wall();
addObject(wall179, 12, 14);
Wall wall180 = new Wall();
addObject(wall180, 13, 14);
```

```
Wall wall181 = new Wall();
addObject(wall181, 14, 14);
Wall wall182 = new Wall();
addObject(wall182, 15, 14);
Wall wall183 = new Wall();
addObject(wall183, 17, 14);
Wall wall184 = new Wall();
addObject(wall184, 18, 14);
Wall wall185 = new Wall();
addObject(wall185, 20, 14);
Wall wall186 = new Wall();
addObject(wall186, 21, 14);
Wall wall187 = new Wall();
addObject(wall187, 22, 14);
Wall wall188 = new Wall();
addObject(wall188, 23, 14);
Wall wall189 = new Wall();
addObject(wall189, 25, 14);
Wall wall190 = new Wall();
addObject(wall190, 26, 14);
Wall wall191 = new Wall();
addObject(wall191, 31, 14);
Wall wall192 = new Wall();
addObject(wall192, 4, 15);
Wall wall193 = new Wall();
addObject(wall193, 5, 15);
Wall wall194 = new Wall();
addObject(wall194, 6, 15);
Wall wall195 = new Wall();
addObject(wall195, 7, 15);
Wall wall196 = new Wall();
addObject(wall196, 9, 15);
Wall wall197 = new Wall();
```

```
addObject(wall197, 10, 15);
Wall wall198 = new Wall();
addObject(wall198, 25, 15);
Wall wall199 = new Wall();
addObject(wall199, 26, 15);
Wall wall200 = new Wall();
addObject(wall200, 28, 15);
Wall wall201 = new Wall();
addObject(wall201, 29, 15);
Wall wall202 = new Wall();
addObject(wall202, 30, 15);
Wall wall203 = new Wall();
addObject(wall203, 31, 15);
Wall wall204 = new Wall();
addObject(wall204, 4, 16);
Wall wall205 = new Wall();
addObject(wall205, 5, 16);
Wall wall206 = new Wall();
addObject(wall206, 6, 16);
Wall wall207 = new Wall();
addObject(wall207, 7, 16);
Wall wall208 = new Wall();
addObject(wall208, 9, 16);
Wall wall209 = new Wall();
addObject(wall209, 10, 16);
Wall wall210 = new Wall();
addObject(wall210, 11, 16);
Wall wall211 = new Wall();
addObject(wall211, 12, 16);
Wall wall212 = new Wall();
addObject(wall212, 14, 16);
Wall wall213 = new Wall();
addObject(wall213, 15, 16);
```

```
Wall wall214 = new Wall();
addObject(wall214, 16, 16);
Wall wall215 = new Wall();
addObject(wall215, 19, 16);
Wall wall216 = new Wall();
addObject(wall216, 20, 16);
Wall wall217 = new Wall();
addObject(wall217, 21, 16);
Wall wall218 = new Wall();
addObject(wall218, 23, 16);
Wall wall219 = new Wall();
addObject(wall219, 24, 16);
Wall wall220 = new Wall();
addObject(wall220, 25, 16);
Wall wall221 = new Wall();
addObject(wall221, 26, 16);
Wall wall222 = new Wall();
addObject(wall222, 28, 16);
Wall wall223 = new Wall();
addObject(wall223, 29, 16);
Wall wall224 = new Wall();
addObject(wall224, 30, 16);
Wall wall225 = new Wall();
addObject(wall225, 31, 16);
Wall wall226 = new Wall();
addObject(wall226, 4, 17);
Wall wall227 = new Wall();
addObject(wall227, 6, 17);
Wall wall228 = new Wall();
addObject(wall228, 7, 17);
Wall wall229 = new Wall();
addObject(wall229, 9, 17);
Wall wall230 = new Wall();
```

```
addObject(wall230, 10, 17);
Wall wall231 = new Wall();
addObject(wall231, 11, 17);
Wall wall232 = new Wall();
addObject(wall232, 12, 17);
Wall wall233 = new Wall();
addObject(wall233, 14, 17);
Wall wall234 = new Wall();
addObject(wall234, 21, 17);
Wall wall235 = new Wall();
addObject(wall235, 23, 17);
Wall wall236 = new Wall();
addObject(wall236, 24, 17);
Wall wall237 = new Wall();
addObject(wall237, 25, 17);
Wall wall238 = new Wall();
addObject(wall238, 26, 17);
Wall wall239 = new Wall();
addObject(wall239, 28, 17);
Wall wall240 = new Wall();
addObject(wall240, 29, 17);
Wall wall241 = new Wall();
addObject(wall241, 31, 17);
Wall wall242 = new Wall();
addObject(wall242, 4, 18);
Wall wall243 = new Wall();
addObject(wall243, 6, 18);
Wall wall244 = new Wall();
addObject(wall244, 7, 18);
Wall wall245 = new Wall();
addObject(wall245, 14, 18);
Wall wall246 = new Wall();
addObject(wall246, 21, 18);
```

```
Wall wall247 = new Wall();
addObject(wall247, 28, 18);
Wall wall248 = new Wall();
addObject(wall248, 29, 18);
Wall wall249 = new Wall();
addObject(wall249, 31, 18);
Wall wall250 = new Wall();
addObject(wall250, 4, 19);
Wall wall251 = new Wall();
addObject(wall251, 6, 19);
Wall wall252 = new Wall();
addObject(wall252, 7, 19);
Wall wall253 = new Wall();
addObject(wall253, 9, 19);
Wall wall254 = new Wall();
addObject(wall254, 10, 19);
Wall wall255 = new Wall();
addObject(wall255, 11, 19);
Wall wall256 = new Wall();
addObject(wall256, 12, 19);
Wall wall257 = new Wall();
addObject(wall257, 14, 19);
Wall wall258 = new Wall();
addObject(wall258, 21, 19);
Wall wall259 = new Wall();
addObject(wall259, 23, 19);
Wall wall260 = new Wall();
addObject(wall260, 24, 19);
Wall wall261 = new Wall();
addObject(wall261, 25, 19);
Wall wall262 = new Wall();
addObject(wall262, 26, 19);
Wall wall263 = new Wall();
```

```
addObject(wall263, 28, 19);
Wall wall264 = new Wall();
addObject(wall264, 29, 19);
Wall wall265 = new Wall();
addObject(wall265, 31, 19);
Wall wall266 = new Wall();
addObject(wall266, 4, 20);
Wall wall267 = new Wall();
addObject(wall267, 9, 20);
Wall wall268 = new Wall();
addObject(wall268, 10, 20);
Wall wall269 = new Wall();
addObject(wall269, 11, 20);
Wall wall270 = new Wall();
addObject(wall270, 12, 20);
Wall wall271 = new Wall();
addObject(wall271, 14, 20);
Wall wall272 = new Wall();
addObject(wall272, 15, 20);
Wall wall273 = new Wall();
addObject(wall273, 16, 20);
Wall wall274 = new Wall();
addObject(wall274, 17, 20);
Wall wall275 = new Wall();
addObject(wall275, 18, 20);
Wall wall276 = new Wall();
addObject(wall276, 19, 20);
Wall wall277 = new Wall();
addObject(wall277, 20, 20);
Wall wall278 = new Wall();
addObject(wall278, 21, 20);
Wall wall279 = new Wall();
addObject(wall279, 23, 20);
```

```
Wall wall280 = new Wall();
addObject(wall280, 24, 20);
Wall wall281 = new Wall();
addObject(wall281, 25, 20);
Wall wall282 = new Wall();
addObject(wall282, 26, 20);
Wall wall283 = new Wall();
addObject(wall283, 31, 20);
Wall wall284 = new Wall();
addObject(wall284, 4, 21);
Wall wall285 = new Wall();
addObject(wall285, 6, 21);
Wall wall286 = new Wall();
addObject(wall286, 7, 21);
Wall wall287 = new Wall();
addObject(wall287, 9, 21);
Wall wall288 = new Wall();
addObject(wall288, 10, 21);
Wall wall289 = new Wall();
addObject(wall289, 25, 21);
Wall wall290 = new Wall();
addObject(wall290, 26, 21);
Wall wall291 = new Wall();
addObject(wall291, 28, 21);
Wall wall292 = new Wall();
addObject(wall292, 29, 21);
Wall wall293 = new Wall();
addObject(wall293, 31, 21);
Wall wall294 = new Wall();
addObject(wall294, 4, 22);
Wall wall295 = new Wall();
addObject(wall295, 6, 22);
Wall wall296 = new Wall();
```

```
addObject(wall296, 7, 22);
Wall wall297 = new Wall();
addObject(wall297, 9, 22);
Wall wall298 = new Wall();
addObject(wall298, 10, 22);
Wall wall299 = new Wall();
addObject(wall299, 12, 22);
Wall wall300 = new Wall();
addObject(wall300, 13, 22);
Wall wall301 = new Wall();
addObject(wall301, 14, 22);
Wall wall302 = new Wall();
addObject(wall302, 15, 22);
Wall wall303 = new Wall();
addObject(wall303, 16, 22);
Wall wall304 = new Wall();
addObject(wall304, 17, 22);
Wall wall305 = new Wall();
addObject(wall305, 18, 22);
Wall wall306 = new Wall();
addObject(wall306, 19, 22);
Wall wall307 = new Wall();
addObject(wall307, 20, 22);
Wall wall308 = new Wall();
addObject(wall308, 21, 22);
Wall wall309 = new Wall();
addObject(wall309, 22, 22);
Wall wall310 = new Wall();
addObject(wall310, 23, 22);
Wall wall311 = new Wall();
addObject(wall311, 25, 22);
Wall wall312 = new Wall();
addObject(wall312, 26, 22);
```

```
Wall wall313 = new Wall();
addObject(wall313, 28, 22);
Wall wall314 = new Wall();
addObject(wall314, 29, 22);
Wall wall315 = new Wall();
addObject(wall315, 31, 22);
Wall wall316 = new Wall();
addObject(wall316, 4, 23);
Wall wall317 = new Wall();
addObject(wall317, 6, 23);
Wall wall318 = new Wall();
addObject(wall318, 7, 23);
Wall wall319 = new Wall();
addObject(wall319, 9, 23);
Wall wall320 = new Wall();
addObject(wall320, 10, 23);
Wall wall321 = new Wall();
addObject(wall321, 12, 23);
Wall wall322 = new Wall();
addObject(wall322, 13, 23);
Wall wall323 = new Wall();
addObject(wall323, 14, 23);
Wall wall324 = new Wall();
addObject(wall324, 15, 23);
Wall wall325 = new Wall();
addObject(wall325, 16, 23);
Wall wall326 = new Wall();
addObject(wall326, 17, 23);
Wall wall327 = new Wall();
addObject(wall327, 18, 23);
Wall wall328 = new Wall();
addObject(wall328, 19, 23);
Wall wall329 = new Wall();
```

```
addObject(wall329, 20, 23);
Wall wall330 = new Wall();
addObject(wall330, 21, 23);
Wall wall331 = new Wall();
addObject(wall331, 22, 23);
Wall wall332 = new Wall();
addObject(wall332, 23, 23);
Wall wall333 = new Wall();
addObject(wall333, 25, 23);
Wall wall334 = new Wall();
addObject(wall334, 26, 23);
Wall wall335 = new Wall();
addObject(wall335, 28, 23);
Wall wall336 = new Wall();
addObject(wall336, 29, 23);
Wall wall337 = new Wall();
addObject(wall337, 31, 23);
Wall wall338 = new Wall();
addObject(wall338, 4, 24);
Wall wall339 = new Wall();
addObject(wall339, 6, 24);
Wall wall340 = new Wall();
addObject(wall340, 7, 24);
Wall wall341 = new Wall();
addObject(wall341, 9, 24);
Wall wall342 = new Wall();
addObject(wall342, 10, 24);
Wall wall343 = new Wall();
addObject(wall343, 17, 24);
Wall wall344 = new Wall();
addObject(wall344, 18, 24);
Wall wall345 = new Wall();
addObject(wall345, 25, 24);
```

```
Wall wall346 = new Wall();
addObject(wall346, 26, 24);
Wall wall347 = new Wall();
addObject(wall347, 28, 24);
Wall wall348 = new Wall();
addObject(wall348, 29, 24);
Wall wall349 = new Wall();
addObject(wall349, 31, 24);
Wall wall350 = new Wall();
addObject(wall350, 4, 25);
Wall wall351 = new Wall();
addObject(wall351, 5, 25);
Wall wall352 = new Wall();
addObject(wall352, 6, 25);
Wall wall353 = new Wall();
addObject(wall353, 7, 25);
Wall wall354 = new Wall();
addObject(wall354, 9, 25);
Wall wall355 = new Wall();
addObject(wall355, 10, 25);
Wall wall356 = new Wall();
addObject(wall356, 11, 25);
Wall wall357 = new Wall();
addObject(wall357, 12, 25);
Wall wall358 = new Wall();
addObject(wall358, 13, 25);
Wall wall359 = new Wall();
addObject(wall359, 14, 25);
Wall wall360 = new Wall();
addObject(wall360, 15, 25);
Wall wall361 = new Wall();
addObject(wall361, 17, 25);
Wall wall362 = new Wall();
```

```
addObject(wall362, 18, 25);
Wall wall363 = new Wall();
addObject(wall363, 20, 25);
Wall wall364 = new Wall();
addObject(wall364, 21, 25);
Wall wall365 = new Wall();
addObject(wall365, 22, 25);
Wall wall366 = new Wall();
addObject(wall366, 23, 25);
Wall wall367 = new Wall();
addObject(wall367, 24, 25);
Wall wall368 = new Wall();
addObject(wall368, 25, 25);
Wall wall369 = new Wall();
addObject(wall369, 26, 25);
Wall wall370 = new Wall();
addObject(wall370, 28, 25);
Wall wall371 = new Wall();
addObject(wall371, 29, 25);
Wall wall372 = new Wall();
addObject(wall372, 30, 25);
Wall wall373 = new Wall();
addObject(wall373, 31, 25);
Wall wall374 = new Wall();
addObject(wall374, 4, 26);
Wall wall375 = new Wall();
addObject(wall375, 5, 26);
Wall wall376 = new Wall();
addObject(wall376, 6, 26);
Wall wall377 = new Wall();
addObject(wall377, 7, 26);
Wall wall378 = new Wall();
addObject(wall378, 9, 26);
```

```
Wall wall379 = new Wall();
addObject(wall379, 10, 26);
Wall wall380 = new Wall();
addObject(wall380, 11, 26);
Wall wall381 = new Wall();
addObject(wall381, 12, 26);
Wall wall382 = new Wall();
addObject(wall382, 13, 26);
Wall wall383 = new Wall();
addObject(wall383, 14, 26);
Wall wall384 = new Wall();
addObject(wall384, 15, 26);
Wall wall385 = new Wall();
addObject(wall385, 17, 26);
Wall wall386 = new Wall();
addObject(wall386, 18, 26);
Wall wall387 = new Wall();
addObject(wall387, 20, 26);
Wall wall388 = new Wall();
addObject(wall388, 21, 26);
Wall wall389 = new Wall();
addObject(wall389, 22, 26);
Wall wall390 = new Wall();
addObject(wall390, 23, 26);
Wall wall391 = new Wall();
addObject(wall391, 24, 26);
Wall wall392 = new Wall();
addObject(wall392, 25, 26);
Wall wall393 = new Wall();
addObject(wall393, 26, 26);
Wall wall394 = new Wall();
addObject(wall394, 28, 26);
Wall wall395 = new Wall();
```

```
addObject(wall395, 29, 26);
Wall wall396 = new Wall();
addObject(wall396, 30, 26);
Wall wall397 = new Wall();
addObject(wall397, 31, 26);
Wall wall398 = new Wall();
addObject(wall398, 4, 27);
Wall wall399 = new Wall();
addObject(wall399, 31, 27);
Wall wall400 = new Wall();
addObject(wall400, 4, 28);
Wall wall401 = new Wall();
addObject(wall401, 6, 28);
Wall wall402 = new Wall();
addObject(wall402, 7, 28);
Wall wall403 = new Wall();
addObject(wall403, 9, 28);
Wall wall404 = new Wall();
addObject(wall404, 10, 28);
Wall wall405 = new Wall();
addObject(wall405, 11, 28);
Wall wall406 = new Wall();
addObject(wall406, 12, 28);
Wall wall407 = new Wall();
addObject(wall407, 13, 28);
Wall wall408 = new Wall();
addObject(wall408, 14, 28);
Wall wall409 = new Wall();
addObject(wall409, 15, 28);
Wall wall410 = new Wall();
addObject(wall410, 16, 28);
Wall wall411 = new Wall();
addObject(wall411, 17, 28);
```

```
Wall wall412 = new Wall();
addObject(wall412, 18, 28);
Wall wall413 = new Wall();
addObject(wall413, 19, 28);
Wall wall414 = new Wall();
addObject(wall414, 20, 28);
Wall wall415 = new Wall();
addObject(wall415, 21, 28);
Wall wall416 = new Wall();
addObject(wall416, 22, 28);
Wall wall417 = new Wall();
addObject(wall417, 23, 28);
Wall wall418 = new Wall();
addObject(wall418, 24, 28);
Wall wall419 = new Wall();
addObject(wall419, 25, 28);
Wall wall457 = new Wall();
addObject(wall457, 26, 28);
Wall wall420 = new Wall();
addObject(wall420, 28, 28);
Wall wall421 = new Wall();
addObject(wall421, 29, 28);
Wall wall422 = new Wall();
addObject(wall422, 31, 28);
Wall wall423 = new Wall();
addObject(wall423, 4, 29);
Wall wall424 = new Wall();
addObject(wall424, 6, 29);
Wall wall425 = new Wall();
addObject(wall425, 7, 29);
Wall wall426 = new Wall();
addObject(wall426, 9, 29);
Wall wall427 = new Wall();
```

```
addObject(wall427, 26, 29);
Wall wall428 = new Wall();
addObject(wall428, 28, 29);
Wall wall429 = new Wall();
addObject(wall429, 29, 29);
Wall wall430 = new Wall();
addObject(wall430, 31, 29);
Wall wall431 = new Wall();
addObject(wall431, 4, 30);
Wall wall432 = new Wall();
addObject(wall432, 6, 30);
Wall wall433 = new Wall();
addObject(wall433, 7, 30);
Wall wall434 = new Wall();
addObject(wall434, 9, 30);
Wall wall435 = new Wall();
addObject(wall435, 26, 30);
Wall wall436 = new Wall();
addObject(wall436, 28, 30);
Wall wall437 = new Wall();
addObject(wall437, 29, 30);
Wall wall438 = new Wall();
addObject(wall438, 31, 30);
Wall wall439 = new Wall();
addObject(wall439, 4, 31);
Wall wall440 = new Wall();
addObject(wall440, 6, 31);
Wall wall441 = new Wall();
addObject(wall441, 7, 31);
Wall wall442 = new Wall();
addObject(wall442, 9, 31);
Wall wall443 = new Wall();
addObject(wall443, 26, 31);
```

```
Wall wall444 = new Wall();
addObject(wall444, 28, 31);
Wall wall445 = new Wall();
addObject(wall445, 29, 31);
Wall wall446 = new Wall();
addObject(wall446, 31, 31);
Wall wall447 = new Wall();
addObject(wall447, 4, 32);
Wall wall448 = new Wall();
addObject(wall448, 9, 32);
Wall wall449 = new Wall();
addObject(wall449, 26, 32);
Wall wall450 = new Wall();
addObject(wall450, 31, 32);
Wall wall451 = new Wall();
addObject(wall451, 4, 33);
Wall wall452 = new Wall();
addObject(wall452, 5, 33);
Wall wall453 = new Wall();
addObject(wall453, 6, 33);
Wall wall454 = new Wall();
addObject(wall454, 7, 33);
Wall wall455 = new Wall();
addObject(wall455, 8, 33);
Wall wall456 = new Wall();
addObject(wall456, 9, 33);

Wall wall458 = new Wall();
addObject(wall458, 26, 33);
Wall wall459 = new Wall();
addObject(wall459, 27, 33);
Wall wall460 = new Wall();
```

```
        addObject(wall460, 28, 33);
        Wall wall461 = new Wall();
        addObject(wall461, 29, 33);
        Wall wall462 = new Wall();
        addObject(wall462, 30, 33);
        Wall wall463 = new Wall();
        addObject(wall463, 31, 33);
      }

      // World behaviour
      private void resetCoins() {
        // Resets the coins after they've all been ate
        if (getNumberOfCoinsAte() == getNumberOfCoinsInLevel()) {
          addSmallDots();
          addBigDots();
        }
      }

      public void act() {
        countdown(this);


        showHealthBar(this);

        showScore(this);

        returnToMenu(backgroundMusic);

        resetCoins();
      }
    }
    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 MainMenu extends World
{

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

                setBackground("background.png");

        // Play Button
        ButtonPlay playButton = new ButtonPlay();
        addObject(playButton, 300, 100);

                //Instructions Button
        ButtonInstructions instructionsButton = new ButtonInstructions();
        addObject(instructionsButton, 300, 200);

        // Exit button
        ButtonExit exitButton = new ButtonExit();
```

```
            addObject(exitButton, 300, 300);
        }
    }
    import greenfoot.*;
    import java.util.List;

    public class Player extends Actor {

      private boolean wasInitialized = false;

      // Movement
      private boolean isMoving;
      private int movementDirection; // 0 - Up, 1 - Right, 2 - Down, 3 - Left
      private int bearMovementAnimationIndex;
      char[] player1MovementKeys;
      String[] player2MovementKeys;

      // Spawn
      private int spawnX;
      private int spawnY;

      // Power up
      private boolean isPoweredUp = false;
      private long poweredUpStartingTime;
      private long poweredUpBlinkingStartingTime;

      // Constructors
      public Player(char[] player1MovementKeys) {
        this.player1MovementKeys = player1MovementKeys;
      }

      public Player(String[] player2MovementKeys) {
```

```
        this.player2MovementKeys = player2MovementKeys;
      }

      // Spawn coordinates
      protected int getSpawnX() {
        return spawnX;
      }

      protected int getSpawnY() {
        return spawnY;
      }

      // Power up
      protected boolean isPoweredUp() {
        return isPoweredUp;
      }

      protected void setIsPoweredUp(boolean newIsPoweredUp) {
        isPoweredUp = newIsPoweredUp;
      }

      protected long getPowerUpStartingTime() {
        return poweredUpStartingTime;
      }

      protected void setPoweredUpStartingTime(long
newPoweredUpStartingTime) {
        poweredUpStartingTime = newPoweredUpStartingTime;
      }

      protected void setPoweredUpBlinkingStartingTime(
        long newPoweredUpBlinkingStartingTime
      ) {
```

```
          poweredUpBlinkingStartingTime =
newPoweredUpBlinkingStartingTime;
        }

        private void blink() {
          // Makes the player blink

          long currentTime = System.currentTimeMillis();

          // 200 ms transparent
          if (currentTime - poweredUpBlinkingStartingTime < 400) {
            getImage().setTransparency(50);
          }
          // 200 ms opaque
          else if (
            (currentTime - poweredUpBlinkingStartingTime) > 400 &&
            (currentTime - poweredUpBlinkingStartingTime) < 800
          ) {
            getImage().setTransparency(255);
          }
          // Resets the cycle
          else {
            poweredUpBlinkingStartingTime =
System.currentTimeMillis();
          }
        }

        private void powerUp() {
          // Makes the player blink if the player is powered up, and
stops the power up after 15 seconds

          // Powered up
          if (isPoweredUp) {
```

```
            blink();
          }

          // Stop the power up after 15 seconds
          if (
            (isPoweredUp) &&
            (System.currentTimeMillis() - poweredUpStartingTime) >
15000
          ) {
            // TODO: Add sound after power up ends
            getImage().setTransparency(255);
            isPoweredUp = false;
          }
        }

        // Movement
        protected void setIsMoving(boolean newIsMoving) {
          isMoving = newIsMoving;
        }

        private void bearMovementAnimation(
          // Animates the bears while they're movings

          GreenfootImage[] BearUpAnimation,
          GreenfootImage[] BearRightAnimation,
          GreenfootImage[] BearDownAnimation,
          GreenfootImage[] BearLeftAnimation
        ) {
          // Isn't moving
          if (!isMoving) {}
          // Moving up
          else if (movementDirection == 0) {
            bearMovementAnimationIndex += 1;
```

```
            if (bearMovementAnimationIndex >= BearUpAnimation.length)
{
              bearMovementAnimationIndex = 0;
            }

            BearUpAnimation[bearMovementAnimationIndex].scale(12,
17);
            setImage(BearUpAnimation[bearMovementAnimationIndex]);
          }
          // Moving right
          else if (movementDirection == 1) {
            bearMovementAnimationIndex += 1;

            if (bearMovementAnimationIndex >=
BearRightAnimation.length) {
              bearMovementAnimationIndex = 0;
            }

            BearRightAnimation[bearMovementAnimationIndex].scale(17,
12);
            setImage(BearRightAnimation[bearMovementAnimationIndex]);
          }
          // Moving down
          else if (movementDirection == 2) {
            bearMovementAnimationIndex += 1;

            if (bearMovementAnimationIndex >=
BearDownAnimation.length) {
              bearMovementAnimationIndex = 0;
            }

            BearDownAnimation[bearMovementAnimationIndex].scale(12,
17);
```

```
            setImage(BearDownAnimation[bearMovementAnimationIndex]);
          }
          // Moving left
          else if (movementDirection == 3) {
            bearMovementAnimationIndex += 1;

            if (bearMovementAnimationIndex >=
BearLeftAnimation.length) {
              bearMovementAnimationIndex = 0;
            }

            BearLeftAnimation[bearMovementAnimationIndex].scale(17,
12);
            setImage(BearLeftAnimation[bearMovementAnimationIndex]);
          }
        }

        private void move(char[] keyboardKeys) {
          // Player 1 movement

          if (Greenfoot.isKeyDown(String.valueOf(keyboardKeys[0]))) {
            isMoving = true;
            movementDirection = 0;
          } else if
(Greenfoot.isKeyDown(String.valueOf(keyboardKeys[1]))) {
            isMoving = true;
            movementDirection = 1;
          } else if
(Greenfoot.isKeyDown(String.valueOf(keyboardKeys[2]))) {
            isMoving = true;
            movementDirection = 2;
          } else if
(Greenfoot.isKeyDown(String.valueOf(keyboardKeys[3]))) {
            isMoving = true;
```

```
            movementDirection = 3;
          }


          if (isMoving) {
            // Moving up
            if (movementDirection == 0) {
              // Checks if there is a wall above before moving up
              if (
                (getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class)).isEmpty() ==
                false
              ) {
                isMoving = false;
              } else {
                setLocation(getX(), getY() - 1);
              }
            }
            // Moving to the right
            else if (movementDirection == 1) {
              // Checks if there is a wall to the right before moving
to the right
              if (
                (getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class)).isEmpty() ==
                false
              ) {
                isMoving = false;
              } else {
                setLocation(getX() + 1, getY());
              }
              // Moving down
            } else if (movementDirection == 2) {
              // Checks if there is a wall below before moving down
              if (
```

```
                (getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class)).isEmpty() ==
                false
              ) {
                isMoving = false;
              } else {
                setLocation(getX(), getY() + 1);
              }
              // Moving to the left
            } else if (movementDirection == 3) {
              // Checks if there is a wall to the left before moving
to the left
              if (
                getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class).isEmpty() ==
                false
              ) {
                isMoving = false;
              } else {
                setLocation(getX() - 1, getY());
              }
            }
          }
        }

        private void move(String[] keyboardKeys) {
          // Player 2 movement

          if (Greenfoot.isKeyDown(keyboardKeys[0])) {
            isMoving = true;
            movementDirection = 0;
          } else if (Greenfoot.isKeyDown(keyboardKeys[1])) {
            isMoving = true;
            movementDirection = 1;
```

```
          } else if (Greenfoot.isKeyDown(keyboardKeys[2])) {
            isMoving = true;
            movementDirection = 2;
          } else if (Greenfoot.isKeyDown(keyboardKeys[3])) {
            isMoving = true;
            movementDirection = 3;
          }

          if (isMoving) {
            // Moving up
            if (movementDirection == 0) {
              // Checks if there is a wall above before moving up
              if (
                (getWorld().getObjectsAt(getX(), getY() - 1,
Wall.class)).isEmpty() ==
                false
              ) {
                isMoving = false;
              } else {
                setLocation(getX(), getY() - 1);
              }
            }
            // Moving to the right
            else if (movementDirection == 1) {
              // Checks if there is a wall to the right before moving
to the right
              if (
                (getWorld().getObjectsAt(getX() + 1, getY(),
Wall.class)).isEmpty() ==
                false
              ) {
                isMoving = false;
              } else {
                setLocation(getX() + 1, getY());
```

```
            }
            // Moving down
          } else if (movementDirection == 2) {
            // Checks if there is a wall below before moving down
            if (
              (getWorld().getObjectsAt(getX(), getY() + 1,
Wall.class)).isEmpty() ==
              false
            ) {
              isMoving = false;
            } else {
              setLocation(getX(), getY() + 1);
            }
            // Moving to the left
          } else if (movementDirection == 3) {
            // Checks if there is a wall to the left before moving
to the left
            if (
              getWorld().getObjectsAt(getX() - 1, getY(),
Wall.class).isEmpty() ==
              false
            ) {
              isMoving = false;
            } else {
              setLocation(getX() - 1, getY());
            }
          }
        }
      }

      public void act() {
        // Individual player initialization
        if (!wasInitialized) {
```

```
            // Sets the respawning position. The respawning position
is set equal to the position that the player actor is initialized
(the position in the first act).
            spawnX = getX();
            spawnY = getY();

            // Player image
            if (player1MovementKeys != null) {
              setImage("be_1.png");
              getImage().scale(17, 12);
            } else {
              setImage("ad_1.png");
              getImage().scale(17, 12);
            }

            wasInitialized = true;
          }

          if (
            (getWorldOfType(Level.class).wasCountdownAlreadyShown())
&&
            (!getWorldOfType(Level.class).getGameOver())
          ) {
            // Player 1 movement
            if (player1MovementKeys != null) {
              move(player1MovementKeys);

              // Player 1 bear up animations
              GreenfootImage[] player1BearUpAnimation = new
GreenfootImage[4];
              player1BearUpAnimation[0] = new
GreenfootImage("bc_1.png");
              player1BearUpAnimation[1] = new
GreenfootImage("bc_2.png");
```

```
            player1BearUpAnimation[2] = new
GreenfootImage("bc_1.png");

            player1BearUpAnimation[3] = new
GreenfootImage("bc_3.png");


            // Player 1 bear right animations

            GreenfootImage[] player1BearRightAnimation = new
GreenfootImage[4];

            player1BearRightAnimation[0] = new
GreenfootImage("bd_1.png");

            player1BearRightAnimation[1] = new
GreenfootImage("bd_2.png");

            player1BearRightAnimation[2] = new
GreenfootImage("bd_1.png");

            player1BearRightAnimation[3] = new
GreenfootImage("bd_3.png");


            // Player 1 bear down animations

            GreenfootImage[] player1BearDownAnimation = new
GreenfootImage[4];

            player1BearDownAnimation[0] = new
GreenfootImage("bb_1.png");

            player1BearDownAnimation[1] = new
GreenfootImage("bb_2.png");

            player1BearDownAnimation[2] = new
GreenfootImage("bb_1.png");

            player1BearDownAnimation[3] = new
GreenfootImage("bb_3.png");


            // Player 1 bear left animations

            GreenfootImage[] player1BearLeftAnimation = new
GreenfootImage[4];

            player1BearLeftAnimation[0] = new
GreenfootImage("be_1.png");

            player1BearLeftAnimation[1] = new
GreenfootImage("be_2.png");

            player1BearLeftAnimation[2] = new
GreenfootImage("be_1.png");
```

```
            player1BearLeftAnimation[3] = new
GreenfootImage("be_3.png");


            bearMovementAnimation(
              player1BearUpAnimation,
              player1BearRightAnimation,
              player1BearDownAnimation,
              player1BearLeftAnimation
            );
          }
          // Player 2 movement
          else {
            move(player2MovementKeys);


            // Player 2 bear up animations
            GreenfootImage[] player1BearUpAnimation = new
GreenfootImage[4];
            player1BearUpAnimation[0] = new
GreenfootImage("ac_1.png");
            player1BearUpAnimation[1] = new
GreenfootImage("ac_2.png");
            player1BearUpAnimation[2] = new
GreenfootImage("ac_1.png");
            player1BearUpAnimation[3] = new
GreenfootImage("ac_3.png");


            // Player 2 bear right animations
            GreenfootImage[] player1BearRightAnimation = new
GreenfootImage[4];
            player1BearRightAnimation[0] = new
GreenfootImage("ad_1.png");
            player1BearRightAnimation[1] = new
GreenfootImage("ad_2.png");
            player1BearRightAnimation[2] = new
GreenfootImage("ad_1.png");
```

```
            player1BearRightAnimation[3] = new
GreenfootImage("ad_3.png");


            // Player 2 bear down animations
            GreenfootImage[] player1BearDownAnimation = new
GreenfootImage[4];
            player1BearDownAnimation[0] = new
GreenfootImage("ab_1.png");
            player1BearDownAnimation[1] = new
GreenfootImage("ab_2.png");
            player1BearDownAnimation[2] = new
GreenfootImage("ab_1.png");
            player1BearDownAnimation[3] = new
GreenfootImage("ab_3.png");


            // Player 2 bear left animations
            GreenfootImage[] player1BearLeftAnimation = new
GreenfootImage[4];
            player1BearLeftAnimation[0] = new
GreenfootImage("ae_1.png");
            player1BearLeftAnimation[1] = new
GreenfootImage("ae_2.png");
            player1BearLeftAnimation[2] = new
GreenfootImage("ae_1.png");
            player1BearLeftAnimation[3] = new
GreenfootImage("ae_3.png");


            bearMovementAnimation(
              player1BearUpAnimation,
              player1BearRightAnimation,
              player1BearDownAnimation,
              player1BearLeftAnimation
            );
          }
        }
```

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

      public class SmallDot extends Actor {

        public SmallDot() {
          GreenfootImage image = getImage();
          image.scale(14, 14);
          setImage(image);
        }

        private void hitPlayer() {
          List<Player> playerHit = getWorld()
            .getObjectsAt(getX(), getY(), Player.class);

          if (playerHit.isEmpty() == false) {
            Level level = getWorldOfType(Level.class);

            level.setScore(level.getScore() + 10);
            level.setNumberOfCoinsAte(level.getNumberOfCoinsAte() +
1);

            GreenfootSound eat = new GreenfootSound("eat.mp3");
            eat.play();
            eat.setVolume(30);

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

```
  public void act() {
    hitPlayer();
  }
}
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 StageChooser extends World
{

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

        setBackground("background.png");

        // Stage 1 Button
                        showText("Stage 1", 200, 100);

        ButtonStage1 stage1Button = new ButtonStage1();
```

```
            addObject(stage1Button, 195, 220);




            // Stage 2 button
                            showText("Stage 2", 400, 100);
            ButtonStage2 stage2Button = new ButtonStage2();
            addObject(stage2Button, 405, 220);
        }
    }
    import greenfoot.*;  // (World, Actor, GreenfootImage,
Greenfoot and MouseInfo)


    /**
    * Write a description of class TurnBack here.
    *
    * @author (your name)
    * @version (a version number or a date)
    */
    public class TurnBack extends Actor
    {
        /**
         * Act - do whatever the TurnBack wants to do. This method
is called whenever
         * the 'Act' or 'Run' button gets pressed in the
environment.
         */


        public TurnBack(){
            GreenfootImage image = getImage();
            image.scale(15, 15);
            setImage(image);
```

```
            image.setTransparency(0);



        }



        public void act()
        {
            //
        }
    }
    import greenfoot.*;  // (World, Actor, GreenfootImage,
Greenfoot and MouseInfo)


    /**
    * Write a description of class Wall here.
    *
    * @author (your name)
    * @version (a version number or a date)
    */
    public class Wall extends Actor
    {
        /**
         * Act - do whatever the Wall wants to do. This method is
called whenever
         * the 'Act' or 'Run' button gets pressed in the
environment.
         */


        public Wall(){


            GreenfootImage image = getImage();
            image.scale(17, 17);
```

```
            setImage(image);
        }



        @Override
        public String toString(){


            return getClass().getName() + "@"+
Integer.toHexString(hashCode()) + ". Desta maneira se faz um
overload";


        }


        public void act()
        {
            // Add your action code here.
        }
    }
```