```
class KeyDemoKeyListener implements KeyListener{

    private GamePanel panel;

    private Projectile p;
```

I created a projectile that can be used via the up, right, down, and left arrows. In order to make this work, I imported the KeyListener and KeyEvent packages which allow for keys to be entered to create a certain effect in the game. I created a class in the gameFrame class to work with this mission.

```
    @Override
    public void keyPressed(KeyEvent e) {
        // TODO Auto-generated method stub

        if(e.getKeyCode() == KeyEvent.VK_RIGHT) {
            panel.moveB(0);
        }
        if(e.getKeyCode()==KeyEvent.VK_DOWN) {
            panel.moveB(1);
        }
        if(e.getKeyCode()==KeyEvent.VK_UP) {
            panel.moveB(2);
        }
        if(e.getKeyCode()==KeyEvent.VK_LEFT) {
            panel.moveB(3);
        }

    }
```

The method above is an auto generated method from the imported package. In this code I had to describe which action would perform what movement. I assigned an integer value to each key that is pressed. This value is then used to process the moveB() method in the GamePanel class. The VK_RIGHT, VK_LEFT, are predefined in the imported package and are connected to

the keys that you press on your keyboard. Using that I then finally called moveB() from the GamePanel class to further process this data.

```
public void moveB(int i) {
    if(i==0) {
        p.setX("right");
    }else if(i==1) {
        p.setY("down");
    }else if(i==2) {
        p.setY("up");
    }else {
        p.setX("left");
    }

}
```

The above moveB() method processes the integer passed on as a parameter from the keyPressed() method in the GameFrame class. I assigned these integers to a direction. 0 equates to “right”, 1 equates to “down” and so on. This method processes the integer and sends data to the setX() or setY() methods created in my Projectile class.

```
public void setX(String h) {
    if(h == "right") {
        this.x += 50;
    }
    if(h =="left") {
        this.x -= 50;
    }

}

public void setY(String v) {
    if(v == "up") {
        this.y -= 50;
    }
    if(v == "down") {
        this.y += 50;
    }

}
```

Finally, the setX() and setY() mutator methods make the actual “projectile” move in various directions. As seen above, if the parameter with the value of “right” is passed on, then the private x-coordinate of the projectile is added by 50. The corresponding code is written for “left”, “up”, and “down” parameters, including the private y-coordinate. This projectile is finally used to attack Mario and Luigi and move to hit them, dropping their health.

In conclusion, in order to lead up to the setX(String), and setY(String) methods, I imported two packages, used them to create method that would allow for keys to be used to create an action in the game, in this case moving the projectile in various directions. The moveB() function calls the setX(String), and setY(String) methods, which creates this complex network of methods in order to allow for the projectile to move in ways the client wants it to move in.