






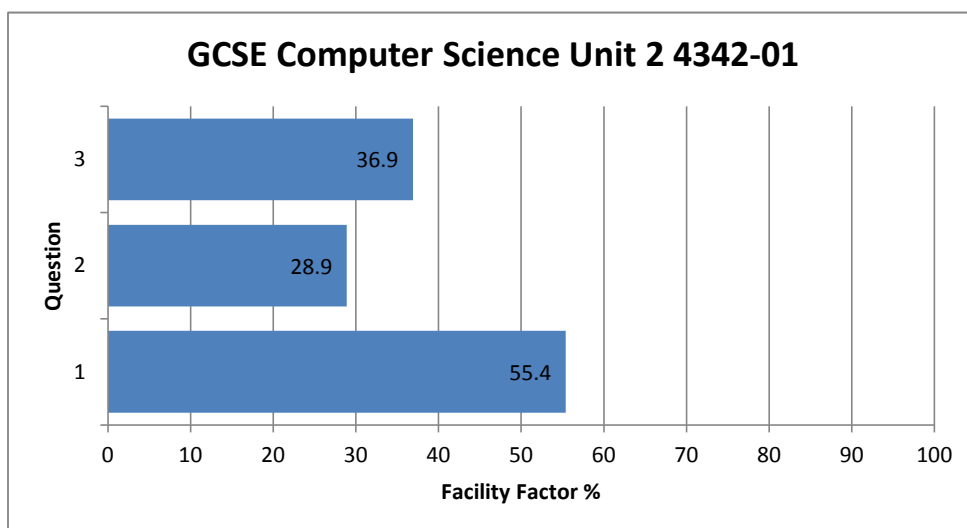


GCSE Computer Science Unit 2 4342-01

All Candidates' performance across questions

 <i>Question Title</i>	 <i>N</i>	 <i>Mean</i>	 <i>S D</i>	 <i>Max Mark</i>	 <i>FF</i>	 <i>Attempt %</i>
1	351	3.3	1.8	6	55.4	97.2
2	330	2.6	2.4	9	28.9	91.4
3	346	5.5	4.7	15	36.9	95.8



Answer Task 1, Task 2 and Task 3.

Task 1

[6]

A first attempt at producing a HTML webpage to advertise a phone recycling company is shown below.

Wanted!

Your old mobile phone for cash!

Click to visit www.phonerecycle.co.uk

Here at PhoneRecycle we can pay you for your old working mobile phones. We recycle the components and refurbish handsets ready for their next use. Please visit our website for a quote today!

The webpage was then improved using various HTML tags to provide the formatting shown below.

Wanted!

Your old mobile phone for cash!

Click to visit www.phonerecycle.co.uk

Here at PhoneRecycle we can pay you for your old working mobile phones. We recycle the components and refurbish handsets ready for their next use. Please visit our website for a quote **today!**

Open the file *phonerecycle.txt* using a basic text editor. Insert the required HTML tags that would be needed to display the formatting shown in the improved webpage. Save your completed work as *Finalphonerecycle.txt*

```
<html>
<body>
<p><center><h1>Wanted!</h1></center></p>
<p><center><b><i>Your old mobile phone for cash! </i></b></center></p>
<p><center>Click to visit <a href="http://www.phonerecycle.co.uk">www.phonerecycle.co.uk</a></center></p>
<p>Here at PhoneRecycle we can pay you for your old working mobile phones. We recycle the components and refurbish
handsets ready for their next use. Please visit our website for a quote <b><u>today!</b></u></p>
</body>
</html>
```

```
<html>
<body>
<p><center><h1>Wanted!</h1></center></p>
<p><center><b><i>Your old mobile phone for cash! </i></b></center></p>
<p><center>Click to visit <a href="http://www.phonerecycle.co.uk">www.phonerecycle.co.uk</a></center></p>
<p>Here at PhoneRecycle we can pay you for your old working mobile phones. We recycle the components and refurbish
handsets ready for their next use. Please visit our website for a quote <b><u>today!</b></u></p>
</body>
</html>
```



```
<html>
<head>
<h1><b><"center">Wanted!</"center"></b></h1></br>
</head>
<body>
</br>
<b><i><"center">Your old mobile phone for cash!</"center"></i></b></br>
</br>
<"center">Click to visit <a href = "www.phonerecycle.co.uk" /> www.phonerecycle.co.uk</a></"center"></br>
<p>
Here at PhoneRecycle we can pay you for your old working mobile phones.
We recycle the components and refurbish handsets ready for their next use.
Please visit our website for a quote <b><u>today!</u></b>
</p>
</body>
</html>
```

```
<html>
<head>
<h1><b><"center">Wanted!</"center"></b></h1></br>
</head>
<body>
</br>
<b><i><"center">Your old mobile phone for cash!</"center"></i></b></br>
</br>
<"center">Click to visit <a href = "www.phonerecycle.co.uk" /> www.phonerecycle.co.uk</a></"center"></br>
<p>
Here at PhoneRecycle we can pay you for your old working mobile phones.
We recycle the components and refurbish handsets ready for their next use.
Please visit our website for a quote <b><u>today!</u></b>
</p>
</body>
</html>
```



<p style="text-align: center;">Wanted! </p>

<p style="text-align: center;"><i>Your old mobile phone for cash! </i></p>

<p style="text-align: center;"> Click to visit www.phonerecycle.co.uk </p>

<p> Here at PhoneRecycle we can pay you for your old working mobile phones.
We recycle the components and refurbish handsets ready for their next use.
Please visit our website for a quote <b style="font-weight: bold;">today! </p>

Wanted!

Your old mobile phone for cash!

Click to visit www.phonerecycle.co.uk

Here at PhoneRecycle we can pay you for your old working mobile phones.
We recycle the components and refurbish handsets ready for their next use.
Please visit our website for a quote **today!**



Task 2

[9]

A diving competition calculates the final mark for each dive based on the marks of six judges. Each judge awards a mark individually (up to a maximum mark of 6.0).

The highest mark and the lowest mark are recorded but **not used** to calculate the final mark. The final mark is calculated by adding the four remaining marks together and dividing by four.

Using a basic text editor, write an algorithm, which inputs six judges' marks and outputs the lowest mark, the highest mark and the final mark. Save your completed algorithm as DivingAlgorithm.txt

For example, with inputs:

5.9

6.0

5.9

5.7

5.6

5.7

The output would be:

Highest:

6.0

Lowest:

5.6

Final Mark:

5.8

```
Highest = 0
Lowest=6
Final Mark=0
For i=1 to i=6
    Input Mark
        If Mark > Highest
            Mark = Highest
        EndIf
        If Mark < Lowest
            Mark = Lowest
        EndIf
    Sum = Sum + Mark
    i = i +1
EndFor
Final Mark = (Sum -| (Highest + Lowest) /4)
Output Final Mark
Output Highest
Output Lowest
```

```
Highest = 0
Lowest=6
Final Mark=0
For i=1 to i=6
    Input Mark
        If Mark > Highest
            Mark = Highest
        EndIf
        If Mark < Lowest
            Mark = Lowest
        EndIf
    Sum = Sum + Mark
    i = i +1
EndFor
Final Mark = (Sum -| (Highest + Lowest) /4)
Output Final Mark
Output Highest
Output Lowest
```



```
First_score = input
highest_score = first_score
lowest_score = first_score
second_score = input
If second_score > highest_score
    highest_score = second_score
endif
if second_score < lowest_score
    lowest_score = second_score
endif
third_score = input
If third_score > highest_score
    highest_score = third_score
end if
if third_score < lowest_score
    lowest_score = third_score
endif
fourth_score = input
If fourth_score > highest_score
    highest_score = forth_score
end if
if fourth_score < lowest_score
    lowest_score = fourth_score
endif
fith_score = input
If fith_score > highest_score
    highest_score = fith_score
end if
if fith_score < lowest_score
    lowest_score = fith_score
endif
sixth_score = input
If sixth_score > highest_score
    highest_score = sixth_score
end if
if sixth_score < lowest_score
    lowest_score = sixth_score
endif
Final_score = (first_score + second_score + fourth_score + fith_score + sith_score - lowest_score - highest_score)/4
Print "Highest:" highest_score
Print "Lowest:" lowest_score
Print " Final Mark: " Final_score
```

```
First_score = input
highest_score = first_score
lowest_score = first_score
second_score = input
If second_score > highest_score
    highest_score = second_score
endif
if second_score < lowest_score
    lowest_score = second_score
endif
third_score = input
If third_score > highest_score
    highest_score = third_score
end if
if third_score < lowest_score
    lowest_score = third_score
endif
fourth_score = input
If fourth_score > highest_score
    highest_score = forth_score
end if
if fourth_score < lowest_score
    lowest_score = fourth_score
endif
fith_score = input
If fith_score > highest_score
    highest_score = fith_score
end if
if fith_score < lowest_score
    lowest_score = fith_score
endif
sixth_score = input
If sixth_score > highest_score
    highest_score = sixth_score
end if
if sixth_score < lowest_score
    lowest_score = sixth_score
endif
Final_score = (first_score + second_score + fourth_score + fith_score + sith_score - lowest_score - highest_score)/4
Print "Highest:" highest_score
Print "Lowest:" lowest_score
Print " Final Mark: " Final_score
```



```
input all six marks from judges
or
input mark from judge 1
input mark from judge 2
input mark from judge 3
input mark from judge 4
input mark from judge 5
input mark from judge 6

out put highest number inputed
out put the lowest number inputed
and
collect remaining four numbers and add them together and then divided them by four to get final mark
output final mark
```

```
input all six marks from judges
or
input mark from judge 1
input mark from judge 2
input mark from judge 3
input mark from judge 4
input mark from judge 5
input mark from judge 6

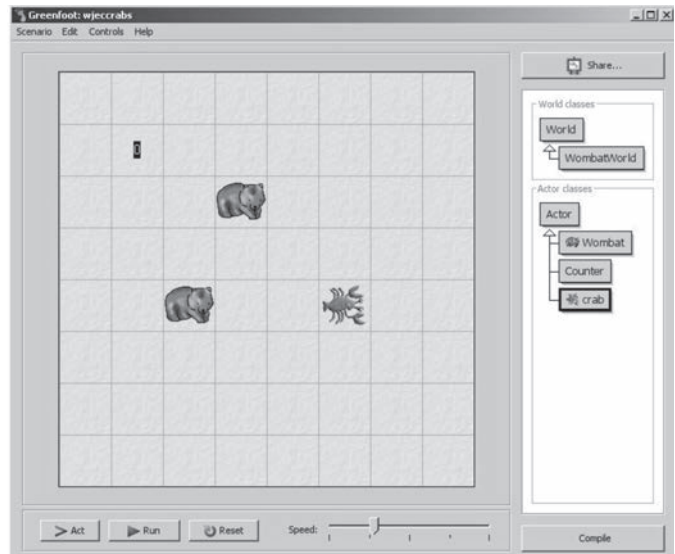
out put highest number inputed
out put the lowest number inputed
and
collect remaining four numbers and add them together and then divided them by four to get final mark
output final mark
```



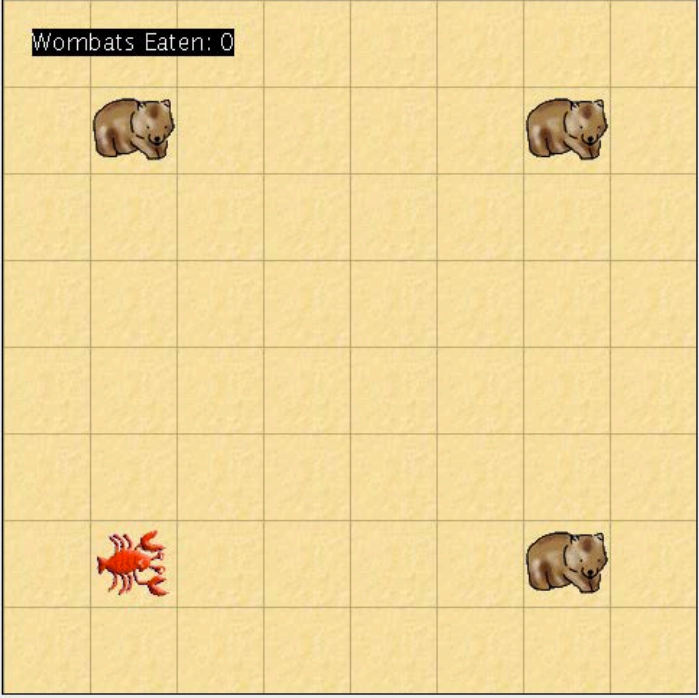
Task 3

[15]

- (a) Open the WJECCrabs scenario in Greenfoot.
- (b) Populate the world with a *crab* and some *wombats*.
- (c) Edit the *wombat* so that it turns and moves randomly.
- (d) Edit the program code to make the *crab* move in the direction of the arrow keys when pressed.
- (e) Edit the *crab* so that it “eats” a *wombat* (removes the wombat from the world) if they collide.
- (f) Add a sound which will play every time a *crab* “eats” a *wombat*.
- (g) Add a counter and edit the crab’s code so that the counter displays how many wombats the crab has “eaten”.
- (h) Save your completed world as FinalWJECCrabs

**END OF PAPER**

Wombats Eaten: 0



Share...

World classes

```

graph BT
    World --> WombatWorld
        
```

Actor classes

```

graph BT
    Actor --> Wombat
    Actor --> Counter
    Actor --> crab
        
```

Act
Run
Reset

Speed:

Compile

```

/**
 * A method that checks if there is a wombat present in the cell that the crab is currently in,
 * and if so, eats it.
 */
public void checkForWombat()
{
    Actor theWombat = getObjectAtOffset(0,0,Wombat.class);
    if (theWombat != null)
    {
        WombatWorld eatingWorld = (WombatWorld) getWorld();
        eatingWorld.removeObject(theWombat);
        Greenfoot.playSound("pop.wav"); //plays a 'popping' sound when a wombat is eaten.
        bumpCounter(1);
    }
}

/**
 * A method that accesses the world's counter and bumps it by a specified integer.
 */
public void bumpCounter(int bumpAmount)
{
    WombatWorld counterWorld = (WombatWorld) getWorld();
    Counter theCounter = counterWorld.getCounter();
    theCounter.bumpCount(bumpAmount);
}

```

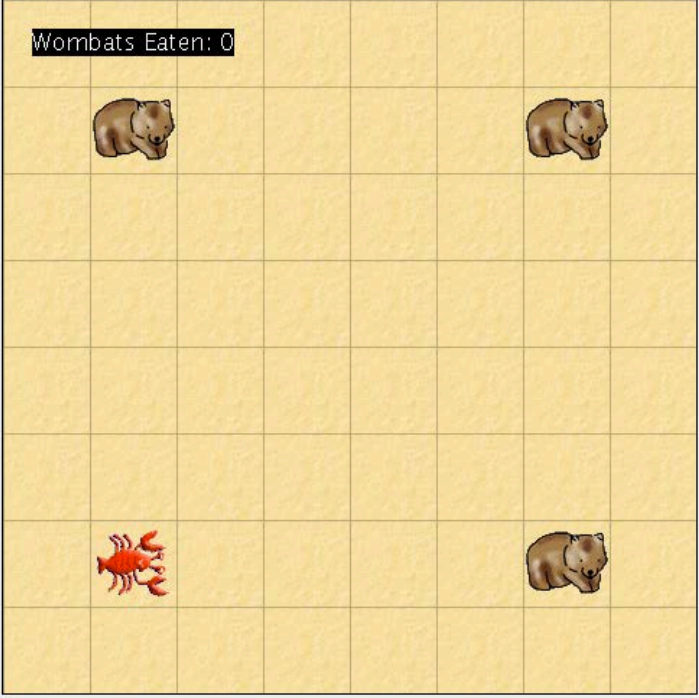
```
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)
import java.awt.Color;
```

```
/**
 *
 */
public class Counter extends Actor
{
    private int totalCount = 0;
    //Creates a variable for storing the counter's count value.

    /**
     * A constructor for the counter, which sets the counter's initial text image.
     */
    public Counter()
    {
        setImage(new GreenfootImage("Wombats Eaten: " + totalCount, 20, Color.WHITE, Color.BLACK));
    }

    /**
     * Increase the total amount displayed on the counter, by a given amount.
     */
    public void bumpCount(int amount)
    {
        totalCount += amount;
        setImage(new GreenfootImage("Wombats Eaten: " + totalCount, 20, Color.WHITE, Color.BLACK));
    }
}
```

Wombats Eaten: 0



Share...

World classes

```

graph BT
    WombatWorld --|> World
          
```

Actor classes

```

graph BT
    Wombat --|> Actor
    Counter --|> Actor
    crab --|> Actor
          
```

Act
Run
Reset

Speed:

Compile



```

/**
 * A method that checks if there is a wombat present in the cell that the crab is currently in,
 * and if so, eats it.
 */
public void checkForWombat()
{
    Actor theWombat = getObjectAtOffset(0,0,Wombat.class);
    if (theWombat != null)
    {
        WombatWorld eatingWorld = (WombatWorld) getWorld();
        eatingWorld.removeObject(theWombat);
        Greenfoot.playSound("pop.wav"); //plays a 'popping' sound when a wombat is eaten.
        bumpCounter(1);
    }
}

/**
 * A method that accesses the world's counter and bumps it by a specified integer.
 */
public void bumpCounter(int bumpAmount)
{
    WombatWorld counterWorld = (WombatWorld) getWorld();
    Counter theCounter = counterWorld.getCounter();
    theCounter.bumpCount(bumpAmount);
}

```



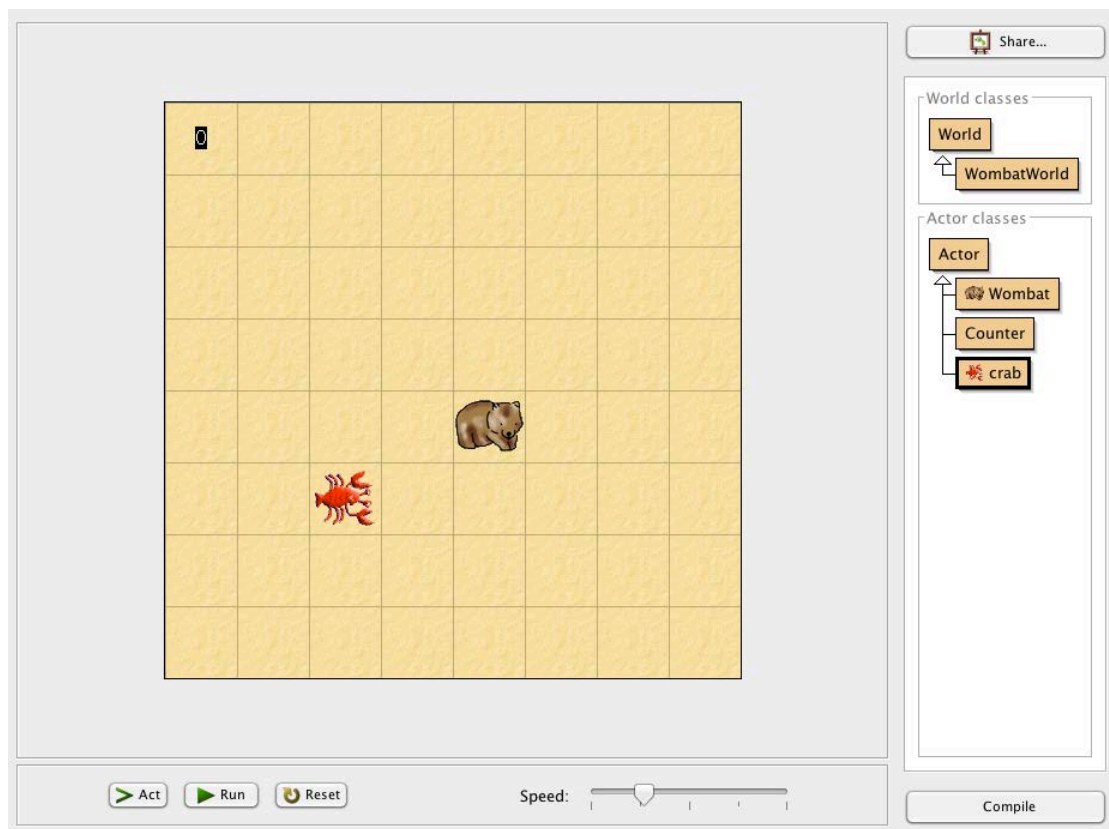
```
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)
import java.awt.Color;
```

```
/**
 *
 */
public class Counter extends Actor
{
    private int totalCount = 0;
    //Creates a variable for storing the counter's count value.

    /**
     * A constructor for the counter, which sets the counter's initial text image.
     */
    public Counter()
    {
        setImage(new GreenfootImage("Wombats Eaten: " + totalCount, 20, Color.WHITE, Color.BLACK));
    }

    /**
     * Increase the total amount displayed on the counter, by a given amount.
     */
    public void bumpCount(int amount)
    {
        totalCount += amount;
        setImage(new GreenfootImage("Wombats Eaten: " + totalCount, 20, Color.WHITE, Color.BLACK));
    }
}
```

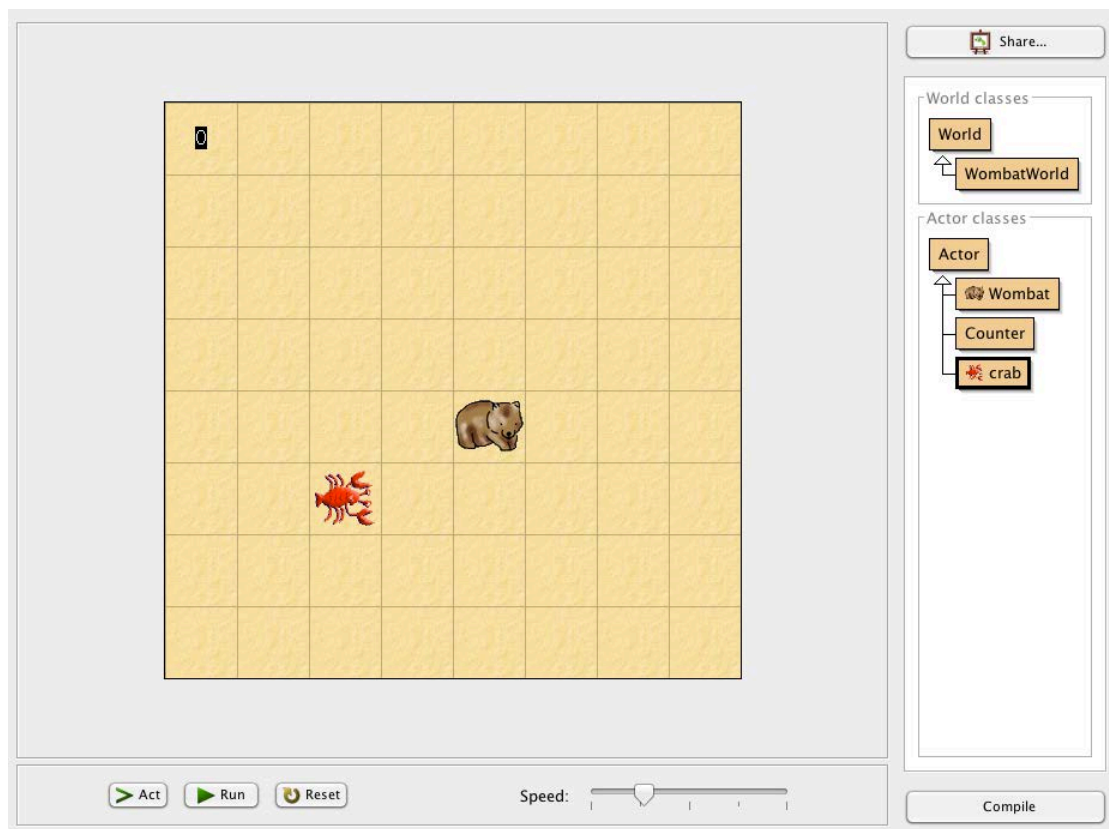




```
move(Greenfoot.getRandomNumber(2));  
turn(Greenfoot.getRandomNumber(90));
```

```
public void act()
{
    if (Greenfoot.isKeyDown("up"))
        move (1);
    if (Greenfoot.isKeyDown("down"))
        move (-1);
    if (Greenfoot.isKeyDown("left"))
    {turn (-90);
    move(1);}
    if (Greenfoot.isKeyDown("right"))
    {turn (90);
    move(1);}
    Actor Wombat = getOneObjectAtOffset(0, 0, Wombat.class);
    if(Wombat != null) {
        getWorld().removeObject(Wombat);
        Greenfoot.playSound("pop.wav");
        totalCount = totalCount + 1;
    }
}

public int aetttotalCount()
```



```
move(Greenfoot.getRandomNumber(2));  
turn(Greenfoot.getRandomNumber(90));
```



```
public void act()
{
    if (Greenfoot.isKeyDown("up"))
        move (1);
    if (Greenfoot.isKeyDown("down"))
        move (-1);
    if (Greenfoot.isKeyDown("left"))
    {turn (-90);
    move(1);}
    if (Greenfoot.isKeyDown("right"))
    {turn (90);
    move(1);}
    Actor Wombat = getOneObjectAtOffset(0, 0, Wombat.class);
    if(Wombat != null) {
        getWorld().removeObject(Wombat);
        Greenfoot.playSound("pop.wav");
        totalCount = totalCount + 1;
    }
}

public int aetttotalCount()
```



Share...

World classes

World

WombatWorld

Actor classes

Actor

Wombat

Counter

crab

Compile


> Act

Pause

Reset

Speed:

0



```
public void act()
{
    move();
}
```

```
public class crab extends Actor
{
    /**
     * Act - do whatever the crab wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void act()
    {
        if (Greenfoot.isKeyDown("right"))
            turn(90);
        if (Greenfoot.isKeyDown("left"))
            turn(-90);
        if (Greenfoot.isKeyDown("up"))
            move(1);
        if (Greenfoot.isKeyDown("down"))
            move(-1);
    }
    public void eat()
    {
    }
}
```



Share...

World classes

World

WombatWorld

Actor classes

Actor

Wombat

Counter

crab

> Act

Pause

Reset

Speed:

Compile



```
public void act()
{
    move();
}
```





```
public class crab extends Actor
{
    /**
     * Act - do whatever the crab wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void act()
    {
        if (Greenfoot.isKeyDown("right"))
            turn(90);
        if (Greenfoot.isKeyDown("left"))
            turn(-90);
        if (Greenfoot.isKeyDown("up"))
            move(1);
        if (Greenfoot.isKeyDown("down"))
            move(-1);
    }
    public void eat()
    {
    }
}
```

