12. Loops and Logic

Topics:

- Loop-Body Return
- The Idea of a Random Walk
- Showcase: Open-Ended Iteration
- Showcase: Computing Averages
- Showcase: Top-Down Development

Loop-Body Returns

Loop-Body Returns

Another way to terminate a loop.

Uses the fact that in a function, control
is passed back to the calling program
as soon as a return statement is encountered.

A Problem

Write a function

`MyFind(char,s)`

that returns **`True`** if character **`char`** is in string **`s`** and returns False otherwise.

.

Typical While-Loop Solution

```
def MyFind(char,s):
    k = 0
    while k<len(s) and char!=s[k]:
        k = k+1
    if k==len(s):
        return False
    else:
        return True
```

When the loop ends, if k==len(s) is True, then we never found an instance of char.

While-Loop Solution with a Loop-Body Return

```
def MyFind(char,s):
    k = 0
    while k<len(s):
        if s[k]==char
            return True
        k = k+1
    return False
```

The function “jumps out of the loop” and returns True should it encounter an instance of char. If the loop runs to completion, that means there is no instance of char.

For Loop Solution with a Loop Body `return`

```
def MyFind(char,s):
    for k in range(len(s)):
        if s[k]==char:
            return True
    return False
```

The function “jumps out of the loop” and returns True should it encounter an instance of char. If the loop runs to completion, that means there is no instance of char.

Another For Loop Solution with a Loop Body `return`

```
def MyFind(char,s):
    for c in s:
        if c==char:
            return True
    return False
```

The function “jumps out of the loop” and returns True should it encounter an instance of char. If the loop runs to completion, that means there is no instance of char.

The Random Walk Idea

-5 -4 -3 -2 -1 0 1 2 3 4 5

We have a "runway" made up of 1x1 tiles.

There are 2L+1 tiles. (L = 5 in the above.)

We call L the "length of the runway.

The center tile is located at x = 0.

The Random Walk Idea

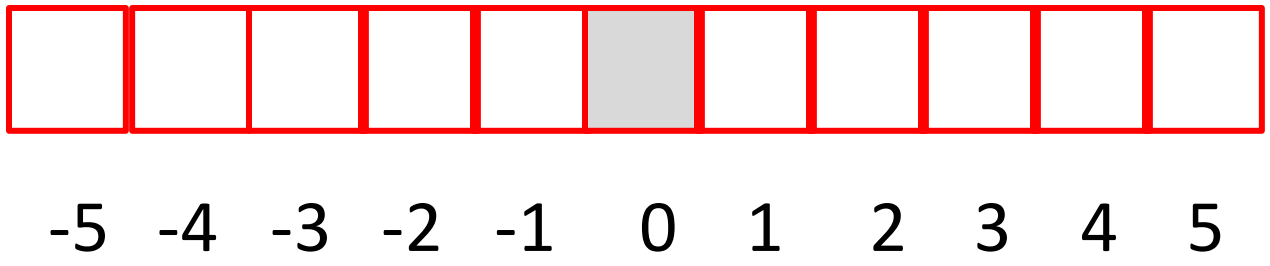
Starting at the center tile, a robot hops from tile to tile according to a coin flip.

Heads: Hop right one tile.

Tails: Hop left one tile.

The simulation over when robot reaches either end (a.k.a. the boundary) of the runway.

The Random Walk Idea



Question:

Given the runway length L, what is the average number of hops required for the robot to reach the boundary?

Implement ShowRandomWalk.py

```
from random import randint as randi

def RandomWalk(L):
    # Returns the number of hops for
    # a single random walk.

def AveRandomWalk(L,n):
    # Simulate n length-L random walks and
    # returns average number of required hops

if __name__ == '__main__':
    # Display the value of AveRandomWalk
    # for various values of L
```

The Application Script

Check out the cases L = 5,10,15,,20,25,30,35,40 :

```
if __name__ == '__main__':
    n = 1000  # Number of trials
    for L in range(5,45,5):
        print L, AveRandomWalk(L,n)
```

The Function AveRandomWalk(L,n)

```
def AveRandomWalk(L,n):
    s = 0
    for k in range(0,n):
        RequiredHops = RandomWalk(L)
        s += RequiredHops
    ave = float(s)/float(n)
    return ave
```

The Function RandomWalk(L)

```
def RandomWalk(L):
    hops = 0; x = 0
    while abs(x) < L:
        r = randi(0,1)
        if r == 0:
            x = x + 1
        else:
            x = x - 1
        hops += 1
    return hops
```

Initializations. The robot starts at x = 0.

The Function RandomWalk(L)

```
def RandomWalk(L):
    hops = 0; x = 0
    while abs(x) < L:
        r = randi(0,1)
        if r == 0:
            x = x + 1
        else:
            x = x - 1
        hops += 1
    return hops
```

If the condition is True, the robot has not yet reached the boundary and we keep iterating..

The Function RandomWalk(L)

```
def RandomWalk(L):
    hops = 0; x = 0
    while abs(x) < L:
        r = randi(0,1)
        if r == 0:
            x = x + 1
        else:
            x = x - 1
        hops += 1
    return hops
```

We simulate the coin toss by picking 0 or 1 at random.

The Function RandomWalk(L)

```
def RandomWalk(L):
    hops = 0; x = 0
    while abs(x) < L:
        r = randi(0,1)
        if r == 0:
            x = x + 1
        else:
            x = x - 1
        hops += 1
    return hops
```

Hop right

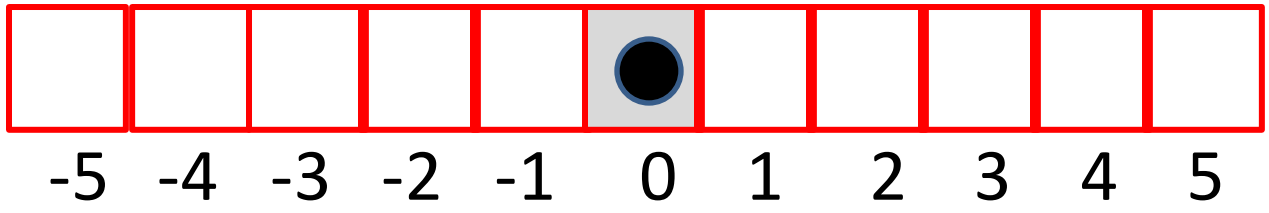
Hop left

The While Loop

To more fully understand how this works, let's look at the execution of this while loop:

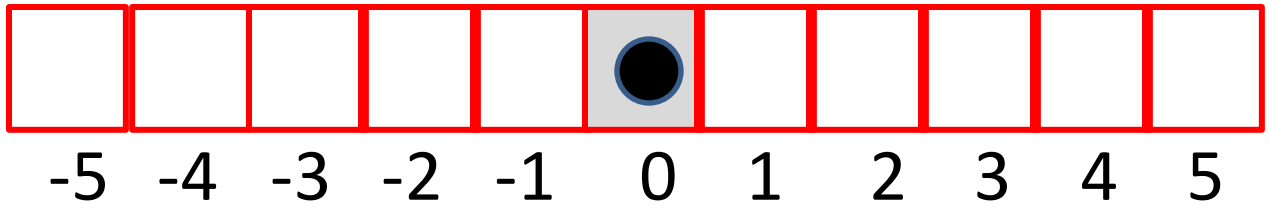
```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While-Loop



```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



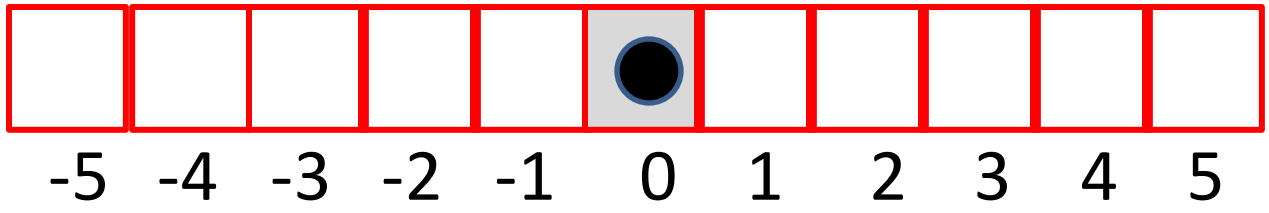
Assume r = 0

Coin = Heads

Hop Right

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

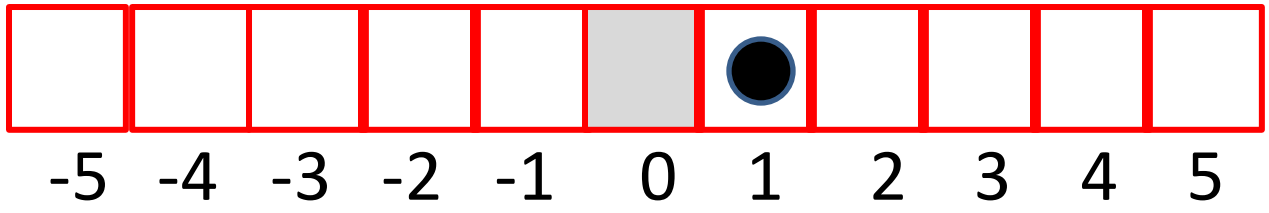
Understanding the While Loop



The value of **x** is increased from 0 to 1.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



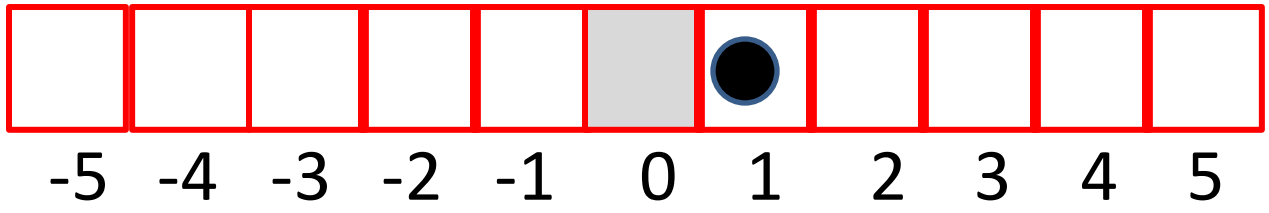
abs(x) < 5 is true.

Robot not at boundary.

Loop continues.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



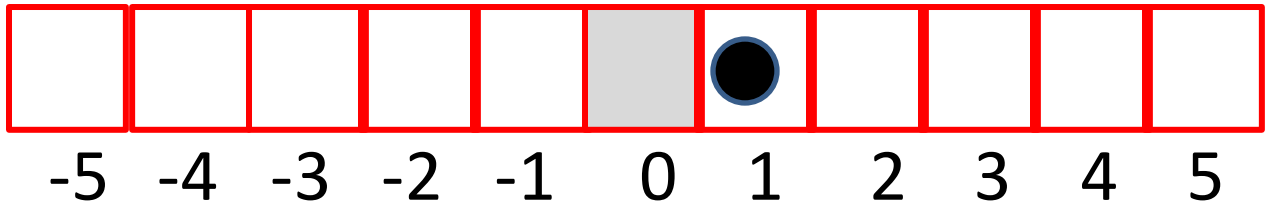
Assume r = 1

Coin = Tails

Hop Left

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

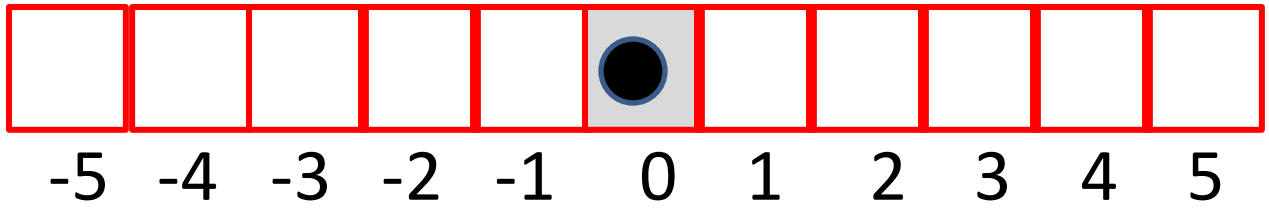
Understanding the While Loop



The value of **x** is decreased from 1 to 0.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



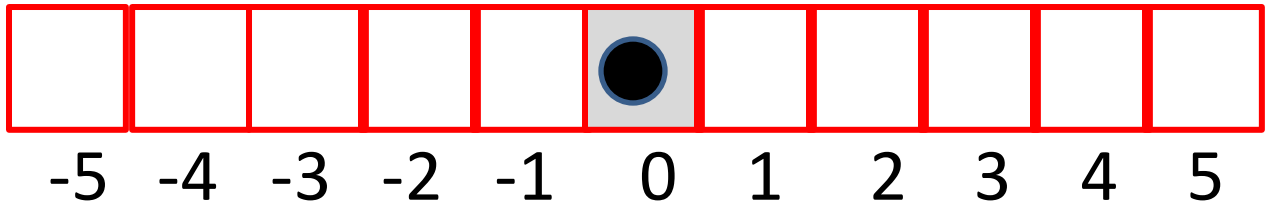
abs(x) < 5 is true.

Robot not at boundary.

Loop continues

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



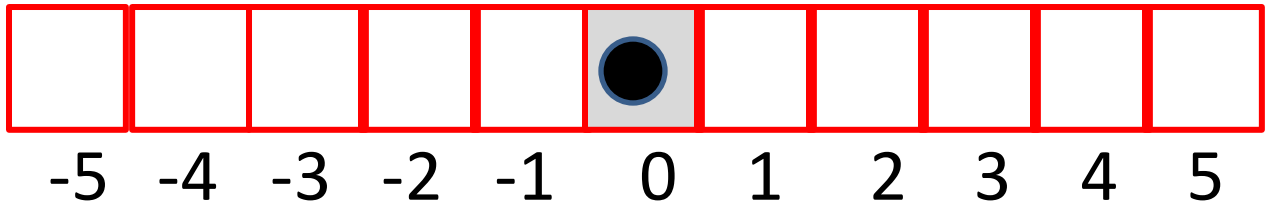
Assume r = 0

Coin = Heads

Hop Right

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

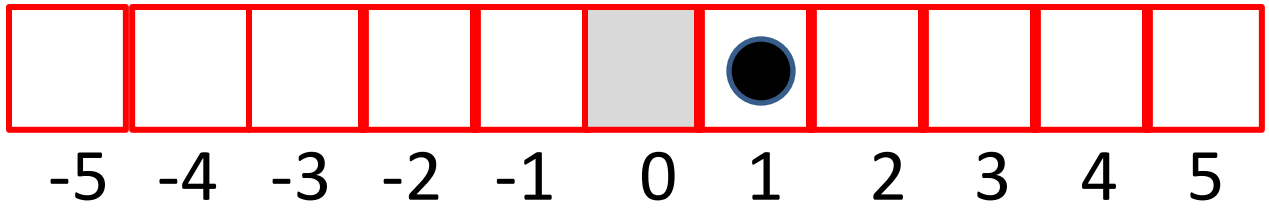
Understanding the While Loop



The value of **x** is increased from 0 to 1.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



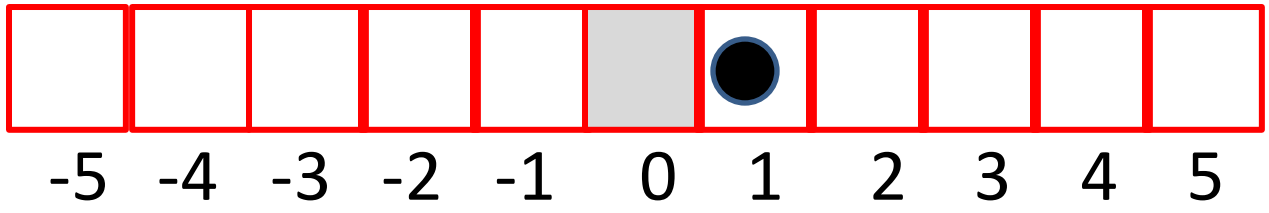
abs(x) < 5 is true.

Robot not at boundary.

Loop continues

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



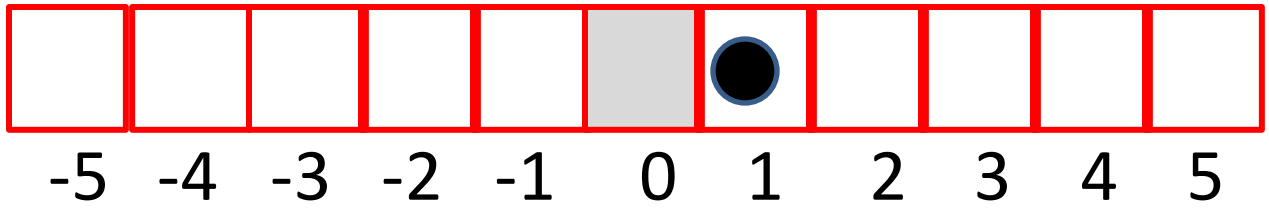
Assume r = 0

Coin = Heads

Hop Right

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

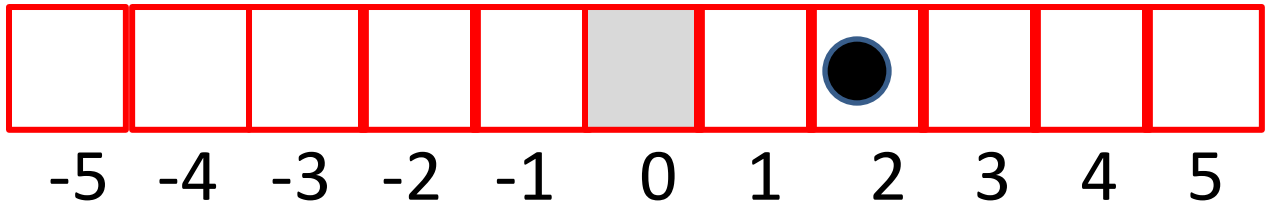
Understanding the While Loop



The value of **x** is increased from 1 to 2.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



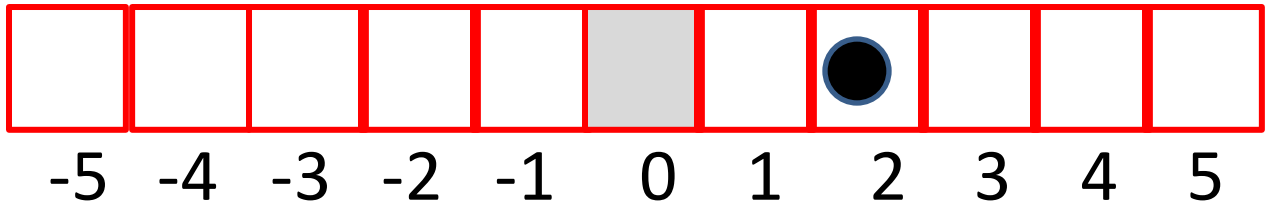
abs(x) < 5 is true.

Robot not at boundary.

Loop continues

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



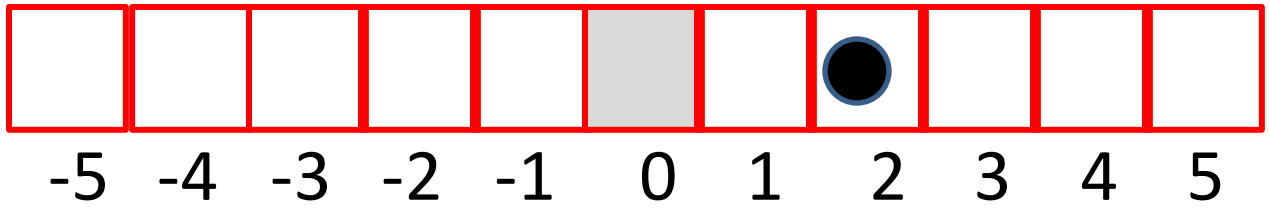
Assume r = 0

Coin = Heads

Hop Right

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

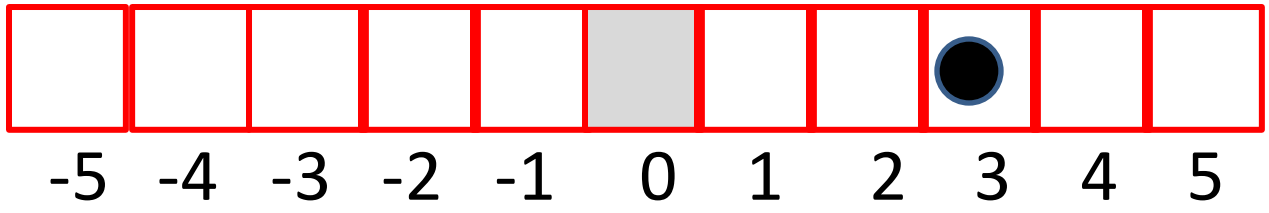
Understanding the While Loop



The value of **x** is increased from 2 to 3.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



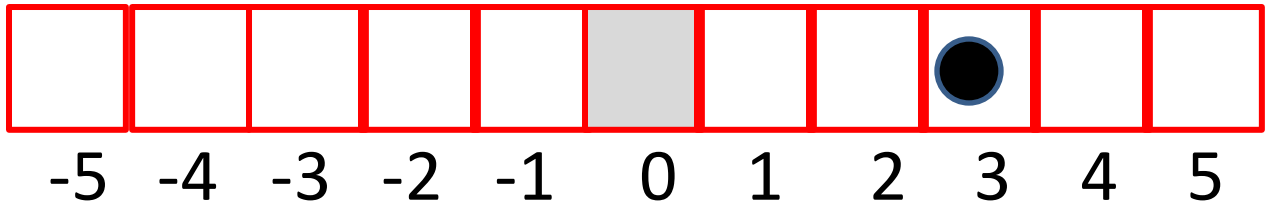
abs(x) < 5 is true.

Robot not at boundary.

Loop continues

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



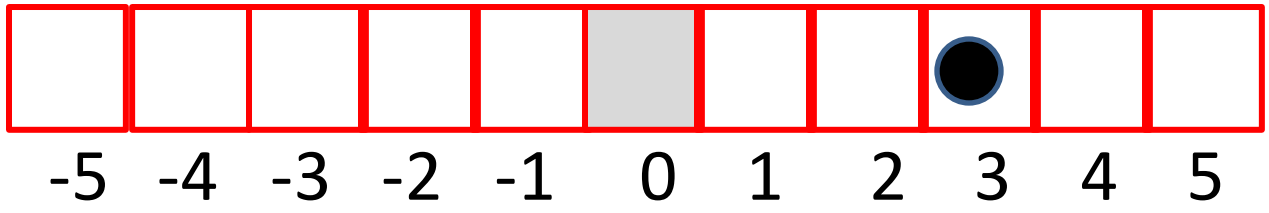
Assume r = 1

Coin = Tails

Hop Left

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

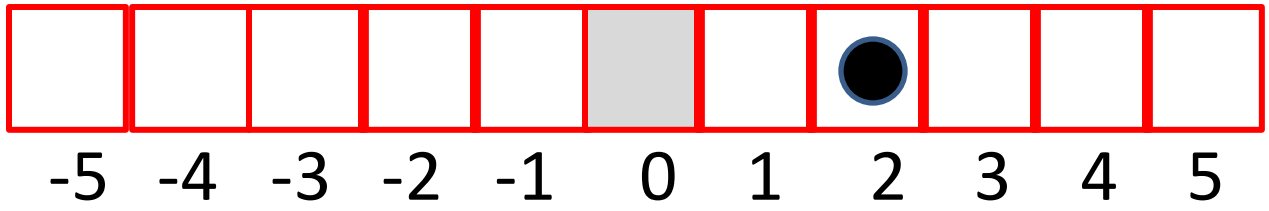
Understanding the While Loop



The value of **x** is decreased from 3 to 2.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



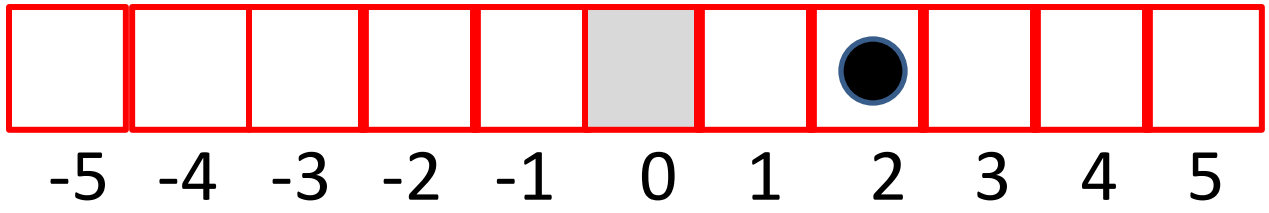
abs(x) < 5 is true.

Robot not at boundary.

Loop continues

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understandingthe While Loop



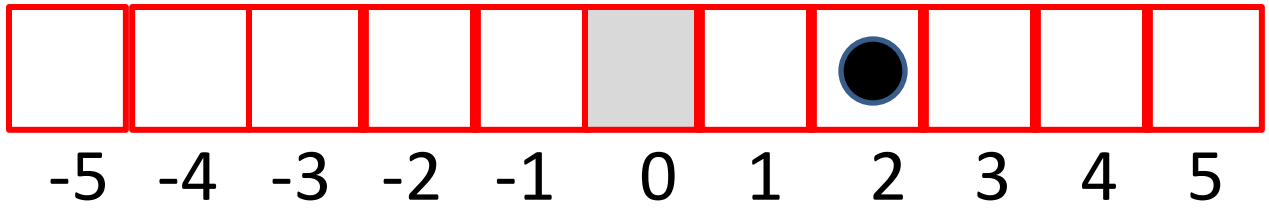
Assume r = 1

Coin = Heads

Hop Right

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

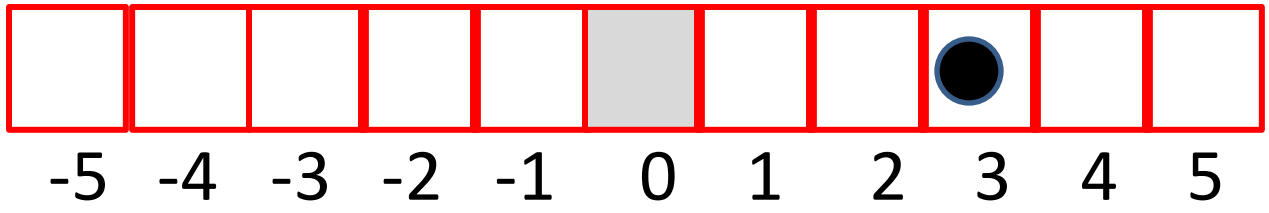
Understanding the While Loop



The value of **x** is increased from 2 to 3.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



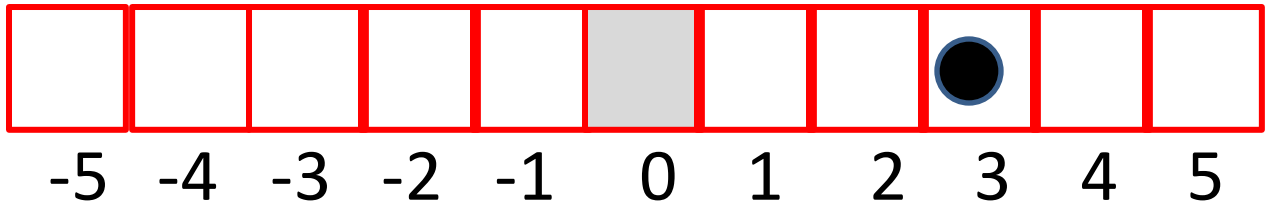
abs(x) < 5 is true.

Robot not at boundary.

Loop continues

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



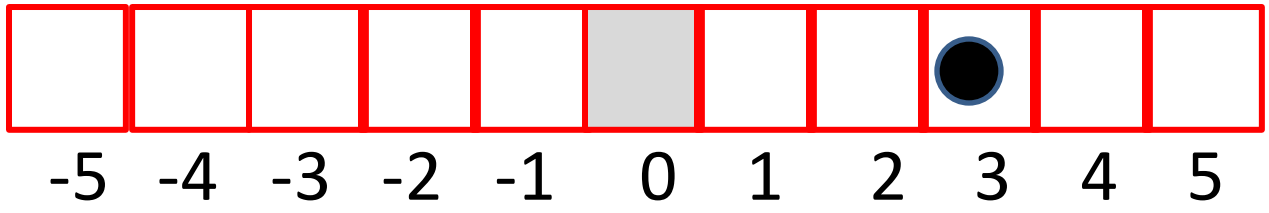
Assume r = 0

Coin = Heads

Hop Right

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

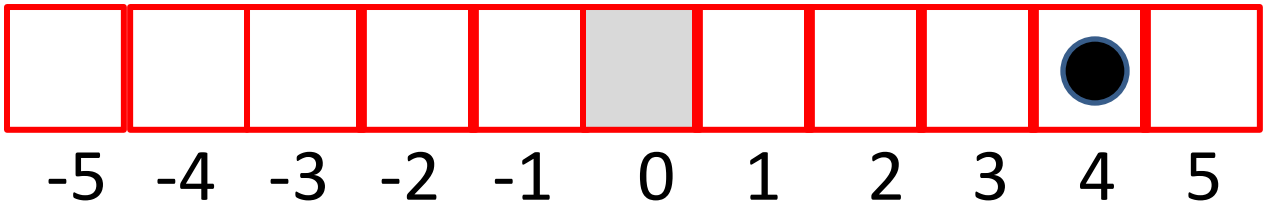
Understanding the While Loop



The value of **x** is increased from 3 to 4.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



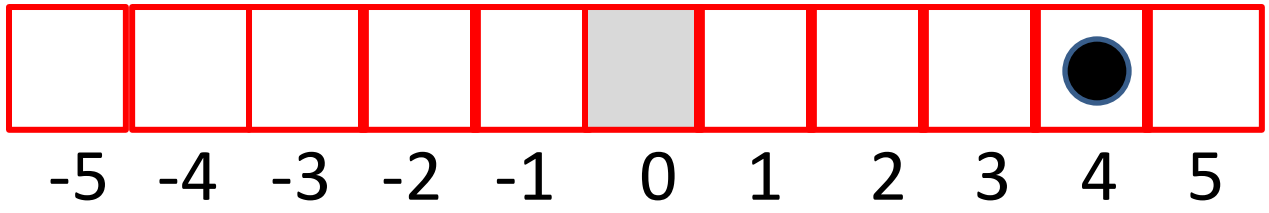
abs(x) < 5 is true.

Robot not at boundary.

Loop continues

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop



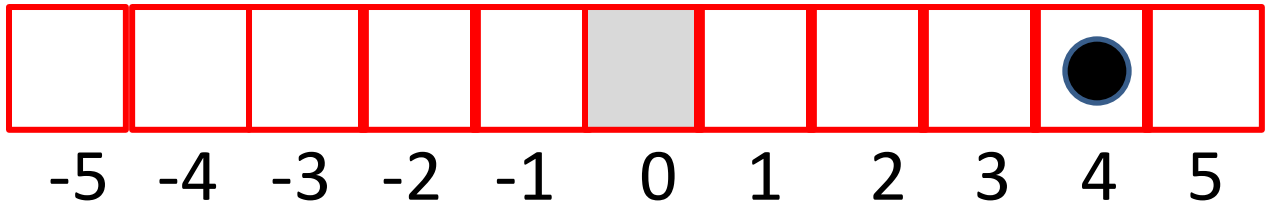
Assume r = 0

Coin = Heads

Hop Right

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

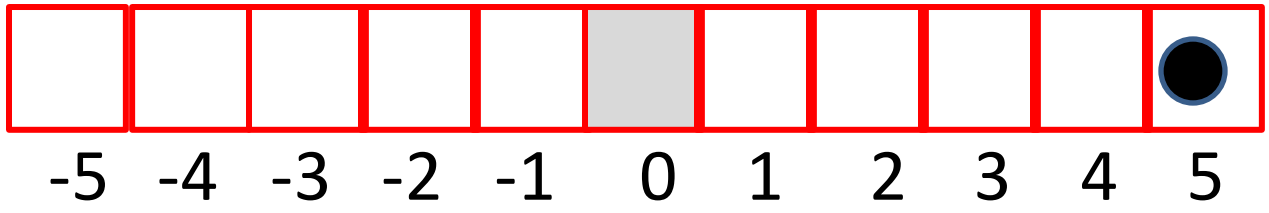
Understanding the While Loop



The value of **x** is increased from 4 to 5.

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

Understanding the While Loop




```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

abs(x) < 5 is False.

Robot is on the boundary.

Loop TERMINATES

Sample Output

```
  L        Ave
------------
5           24
10          93
15         219
20         399
25         649
30         917
35        1259
40        1594

Averages based on 1000 trials.
```

Looks like doubling L increases the average by a factor of 4.

Insight through Computing!