10. Iteration: The `while`-Loop

Topics:

- Open-Ended repetition
- the `while` statement
- Random Walk Simulation

Open-Ended Iteration

So far, we have only addressed iterative problems in which we know (in advance) the required number of repetitions.

Not all iteration problems are like that.

Some iteration problems are open-ended.

Stir for 5 minutes   vs   Stir until fluffy.

Examples

Keep tossing a coin until the number of heads and the number of tails differs by 10.

Compute the square root of 2....

L = 2; W = 1

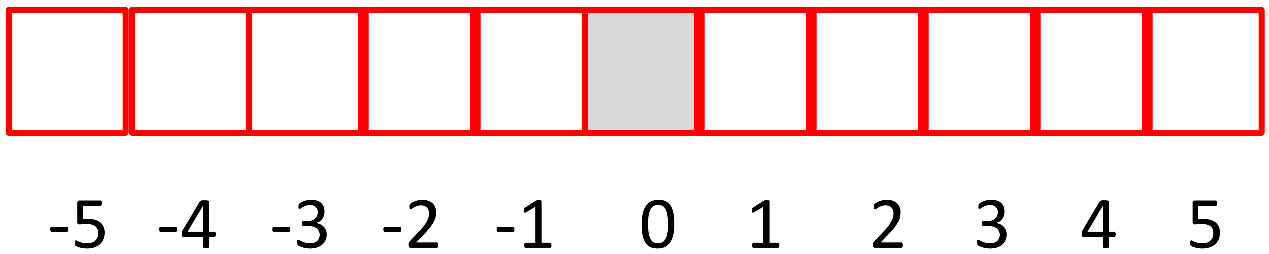
Repeat this until |L-W| <= .000001:

    L = (L + W)/2

    W = x/L

In both cases, we do not know the number of iterations that will be required

The Random Walk Idea



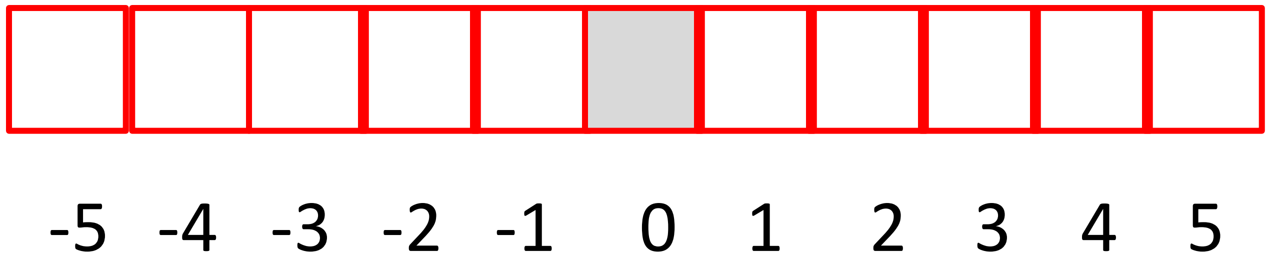
We have a "runway" made up of 1x1 tiles.

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.

We do not know in advance how many iterations we'll need,

The While Loop

We introduce an alternative to the for-loop called the while-loop.

The while loop is more flexible and is essential for ``open ended" iteration.

How Does a While-Loop Work?

A simple warm-up example:

Sum the first 5 whole numbers and display the summation process.

Two Solutions

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

```
s = 0
for k in range(1,6):
    s = s + k
    print k,s
```

The While-Loop Solution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

```
1   1
2   3
3   6
4  10
5  15
```

Observation: `k` is used for counting, `s` is used for the running sum, and the `while` is used to control the repetition of the indented code.

The Solution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

```
1   1
2   3
3   6
4  10
5  15
```

We call this the "loop body"

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 0

s -> 0

At the start, k and s are initialized

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 0

s -> 0

Is the boolean condition true?

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 0

s -> 0

Yes, so execute the loop body

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 1

s -> 1

```
1 1
```

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 1

s -> 1

1 1

Is the boolean condition true?

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 1

s -> 1

1 1

Yes, so execute the loop body

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 2

s -> 3

```
1  1
2  3
```

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 2

s -> 3

```
1  1
2  3
```

Is the boolean condition true?

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 2

s -> 3

1	1
2	3

Yes, so execute the loop body

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 3

s -> 6

```
1  1
2  3
3  6
```

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 3

s -> 6

```
1 1
2 3
3 6
```

Is the boolean condition true?

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 3

s -> 6

```
1 1
2 3
3 6
```

Yes, so execute the loop body

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 4

s -> 10

1	1
2	3
3	6
4	10

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 4

s -> 10

1	1
2	3
3	6
4	10

Is the boolean condition true?

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 4

s -> 10

```
1  1
2  3
3  6
4  10
```

Yes, so execute the loop body

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 5

s -> 15

```
1  1
2  3
3  6
4  10
5  15
```

Trace the Execution

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

k -> 5

s -> 15

```
1  1
2  3
3  6
4  10
5  15
```

Is the boolean condition true?
NO! The loop is over.

The While-Loop Mechanism

while *A Boolean Expression* **:**

The Loop Body

The Boolean expression is checked. If it is true, then the loop body is executed. The process is repeated until the Boolean expression is false. At that point the iteration terminates.

The Broader Context

Code that comes before the loop

while *A Boolean Expression* **:**

The Loop Body

Code that comes after the loop

Every variable involved in the Boolean expression must be initialized.

The Broader Context

Code that comes before the loop

while *A Boolean Expression* **:**

The Loop Body

Code that comes after the loop

After the loop terminates the next statement after the loop is executed.

The Broader Context

Code that comes before the loop

while *A Boolean Expression* **:**

The Loop Body

Code that comes after the loop

Indentation defines the loop body

Back to Our Example

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
    print k,s
```

```
1  1
2  3
3  6
4  10
5  15
```

Let's move the print statement outside the loop body

Back to Our Example

```
k = 0
s = 0
while k < 5:
    k = k + 1
    s = s + k
print k,s
```

```
5  15
```

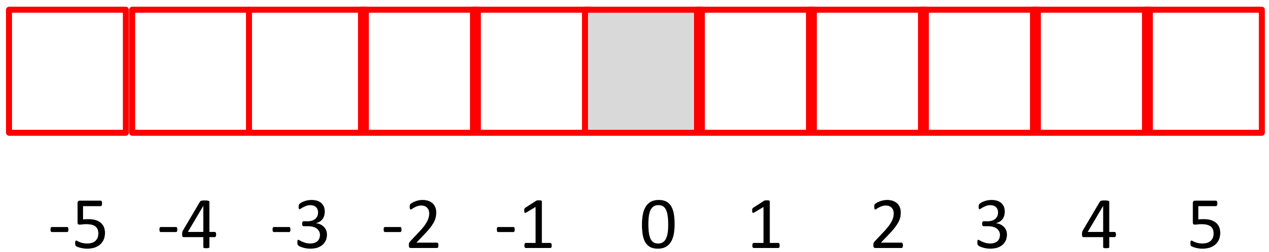
Only the final value of k and s are reported.

Random Walks

A very important type of random simulation.

A good example to showcase the while loop.

The Random Walk Idea



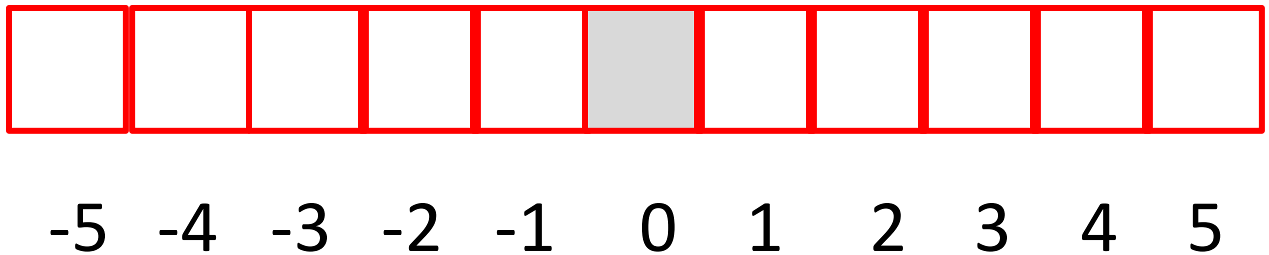
We have a “runway” made up of 1x1 tiles.

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



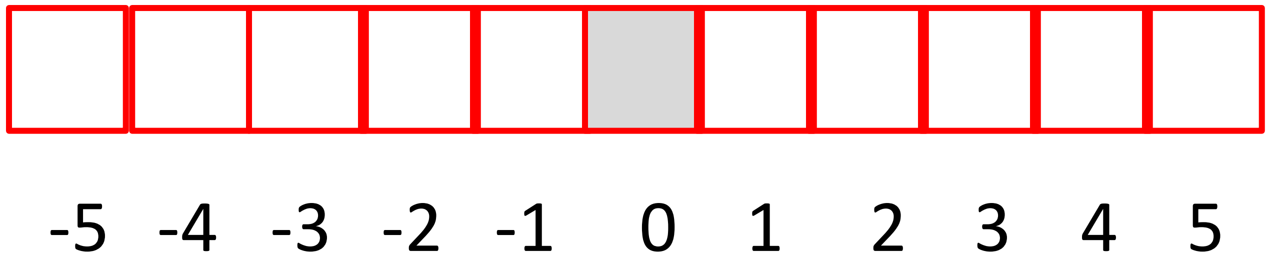
Starting at the center tile, a robot hops from tile to tile according to a coin flip.

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 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

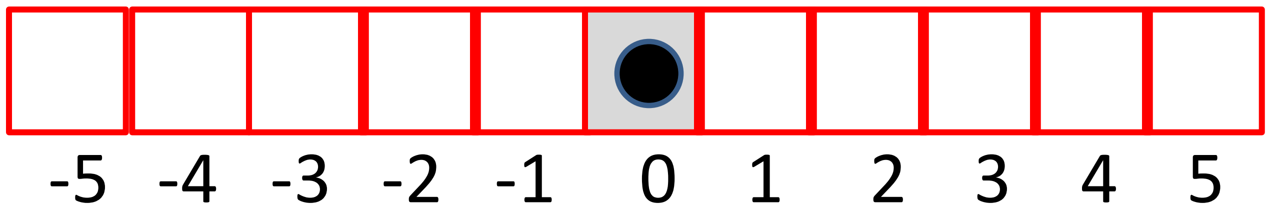
Hop left

The While Loop

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

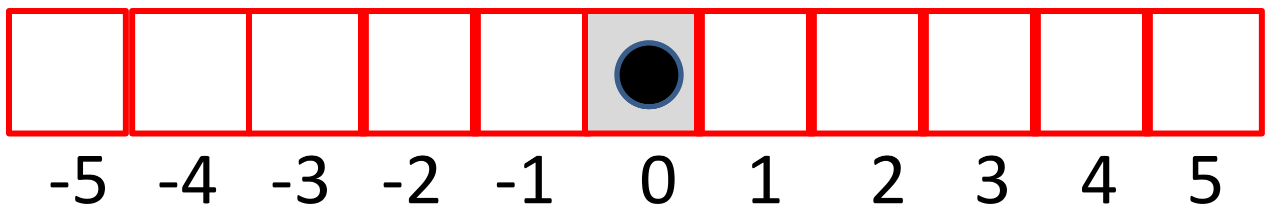
```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

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



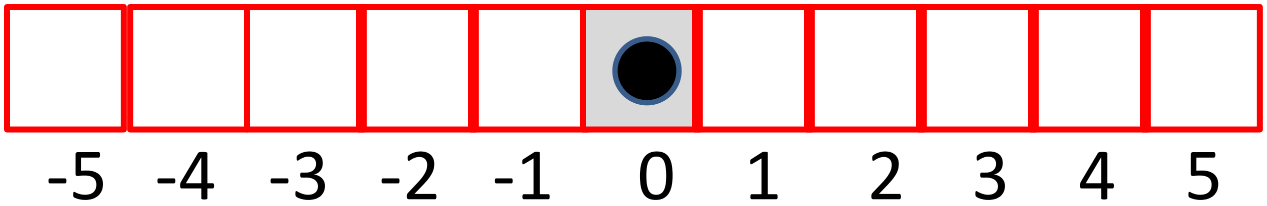
Assume r = 0

Coin = Heads

Hop Right

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

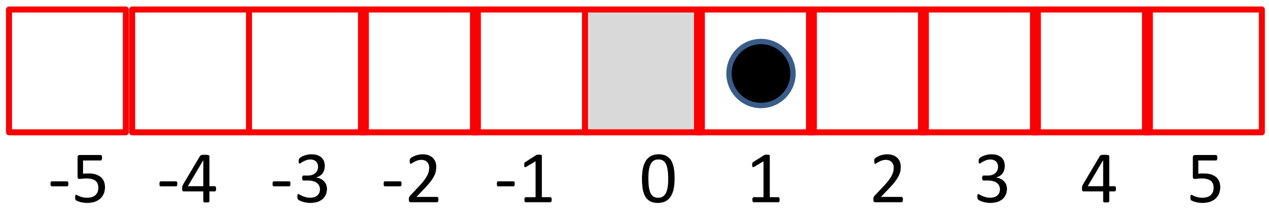
Understanding the While Loop



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



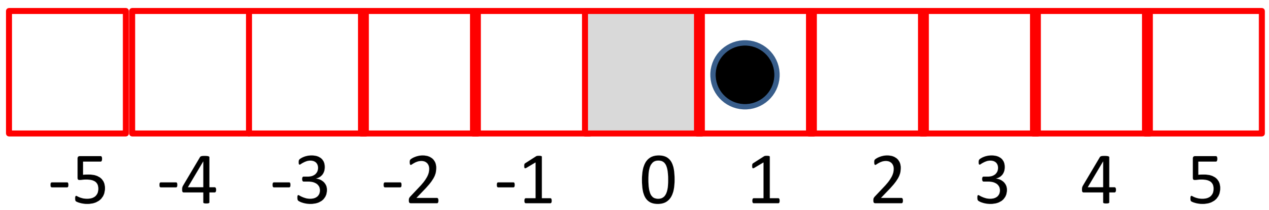

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

abs(x) < 5 is true.

Robot not at boundary.

Loop continues.

Understanding the While Loop



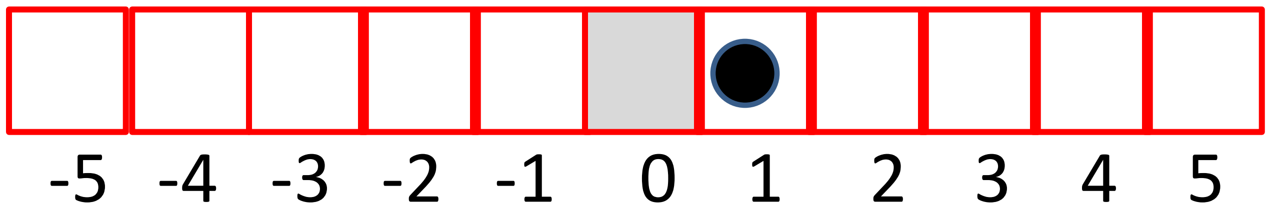
Assume r = 1

Coin = Tails

Hop Left

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

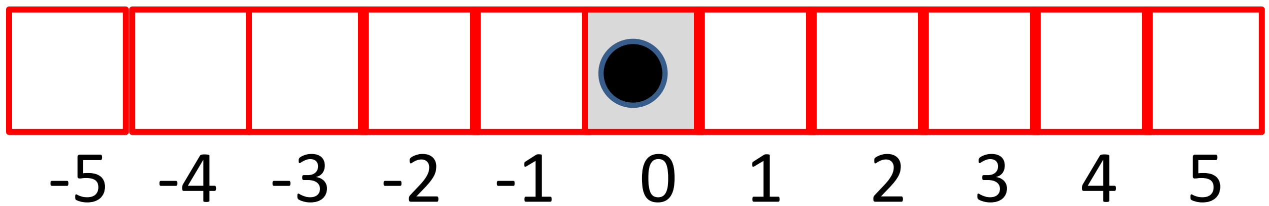
Understanding the While Loop



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



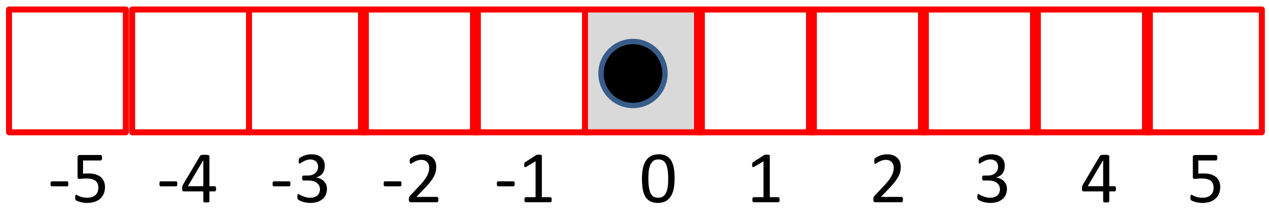
abs(x) < 5 is true.

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



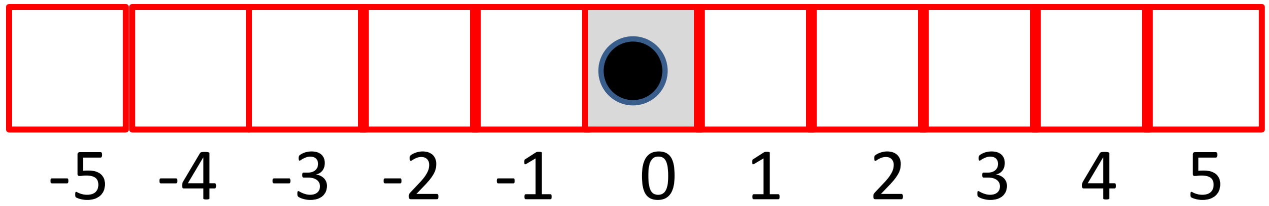
Assume r = 0

Coin = Heads

Hop Right

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

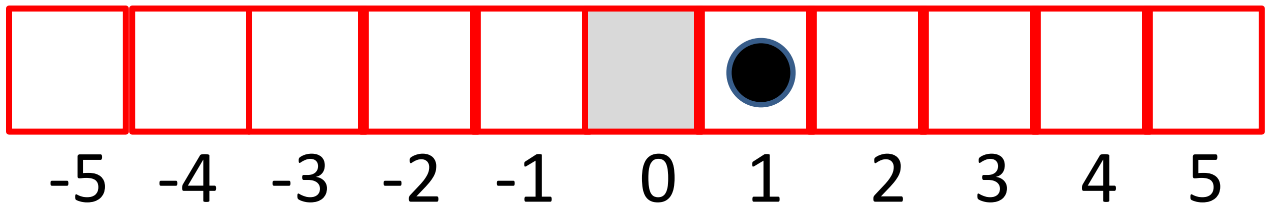
Understanding the While Loop



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



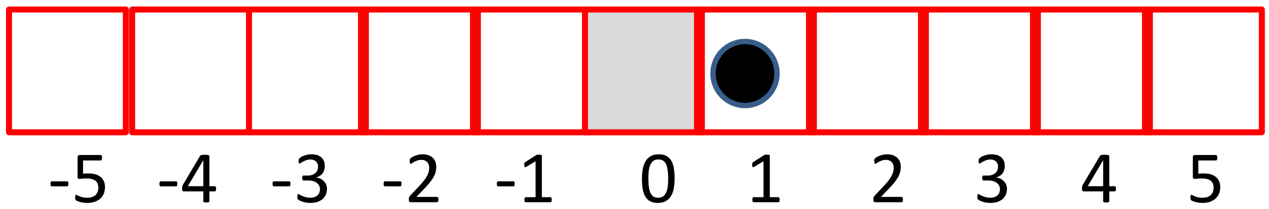
abs(x) < 5 is true.

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



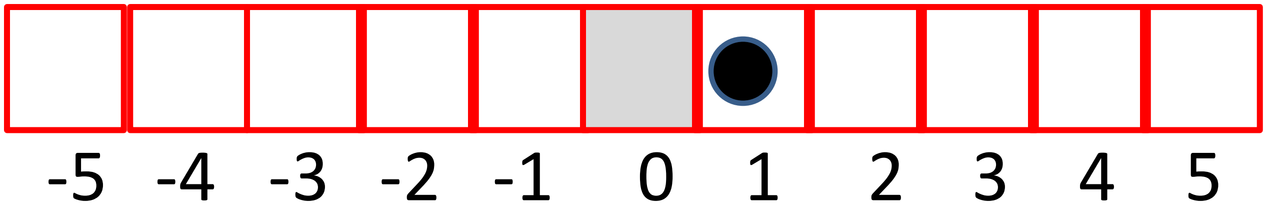
Assume r = 0

Coin = Heads

Hop Right

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

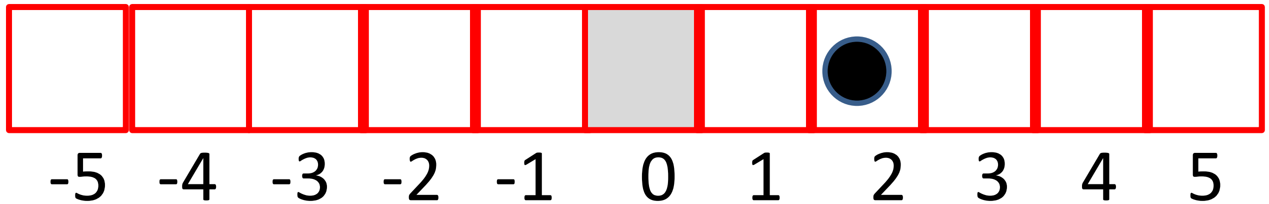
Understanding the While Loop




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

The value of x is increased from 1 to 2.

Understanding the While Loop



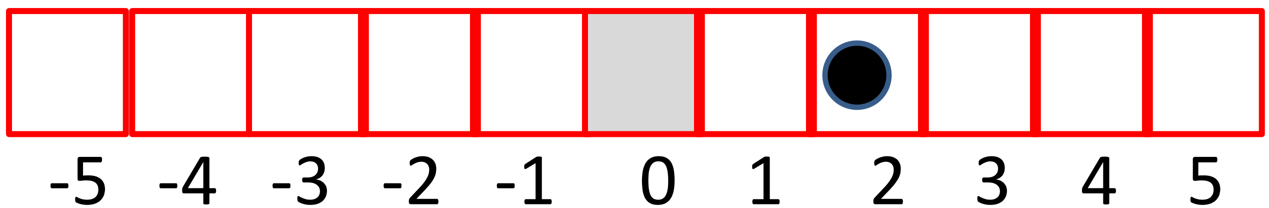
abs(x) < 5 is true.

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



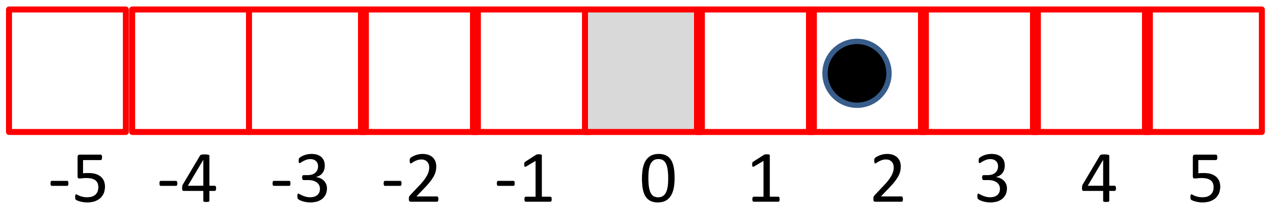
Assume r = 0

Coin = Heads

Hop Right

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

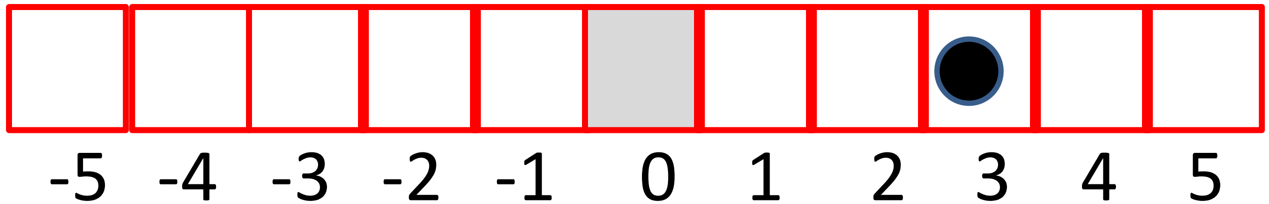
Understanding the While Loop



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



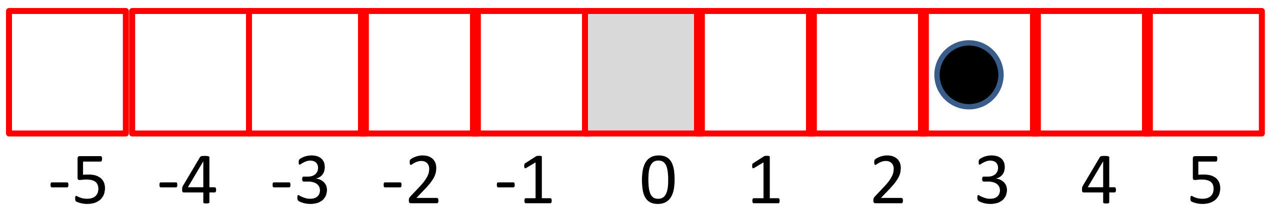
abs(x) < 5 is true.

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



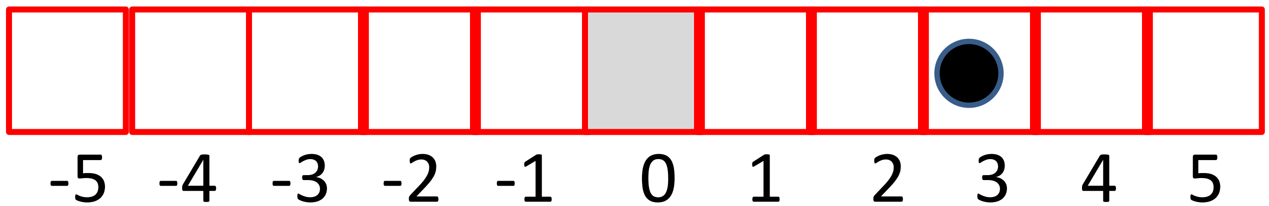
Assume r = 1

Coin = Tails

Hop Left

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

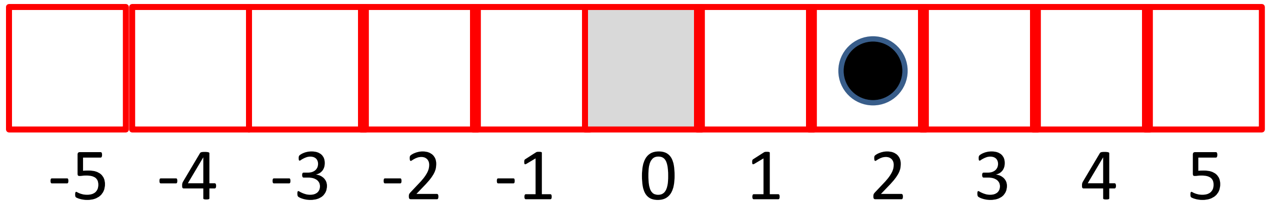
Understanding the While Loop



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



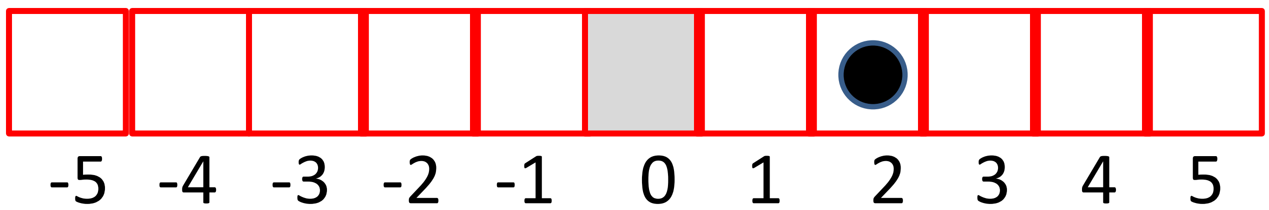

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

abs(x) < 5 is true.

Robot not at boundary.

Loop continues

Understandingthe While Loop



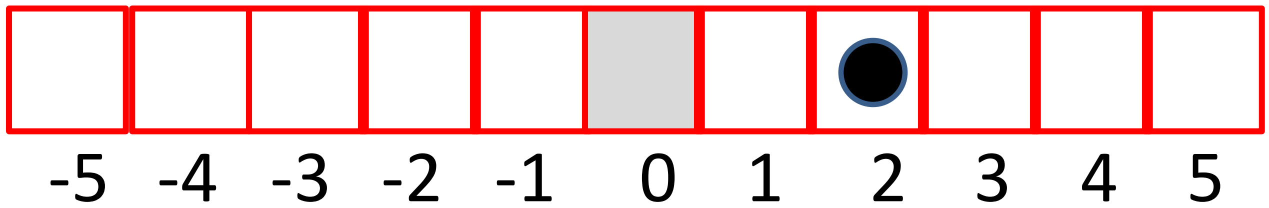
Assume r = 1

Coin = Heads

Hop Right

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

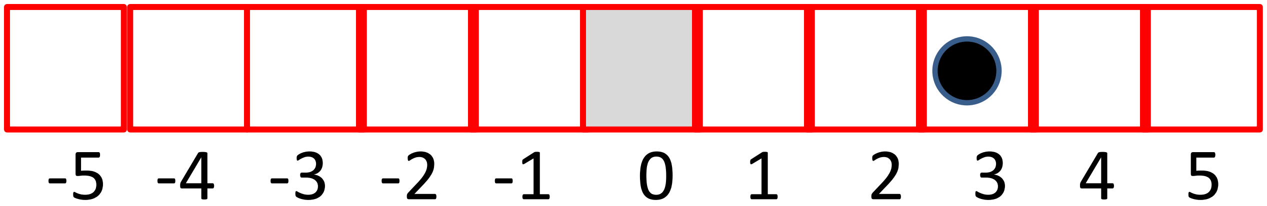
Understanding the While Loop



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



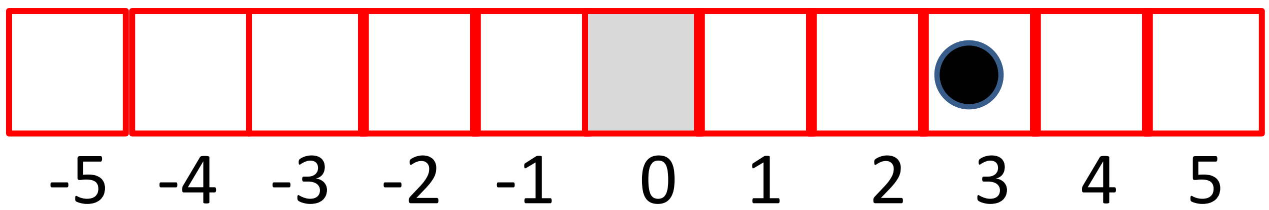

```
x = 0
while abs(x) < 5:
    r = randi(0,1)
    if r == 0:
        x = x+1
    else:
        x = x-1
```

abs(x) < 5 is true.

Robot not at boundary.

Loop continues

Understanding the While Loop



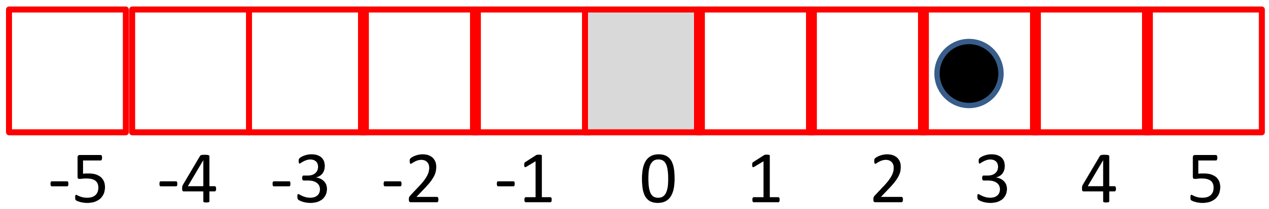
Assume r = 0

Coin = Heads

Hop Right

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

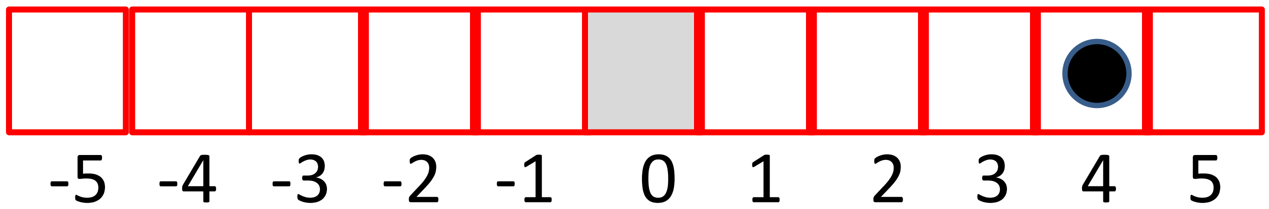
Understanding the While Loop



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



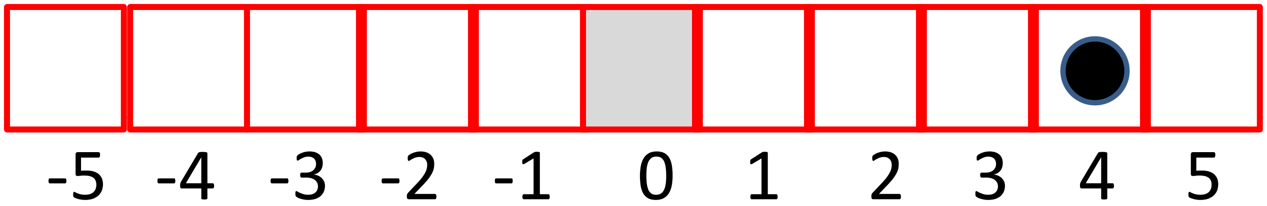
abs(x) < 5 is true.

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



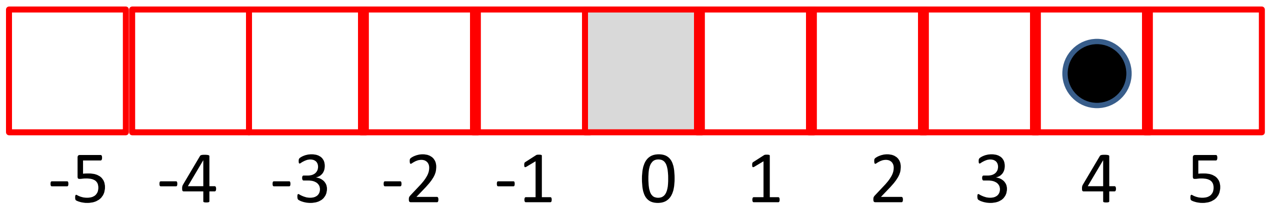
Assume r = 0

Coin = Heads

Hop Right

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

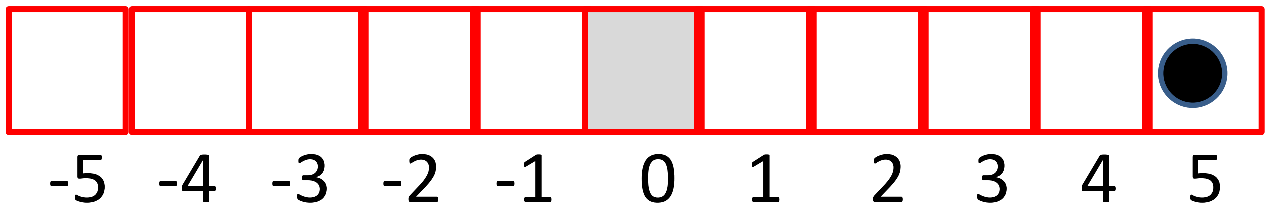
Understanding the While Loop



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



abs(x) < 5 is False.

Robot is on the boundary.

Loop TERMINATES

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

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
```

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!