Random Numbers

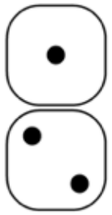
INTERMEDIATE PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

datacamp

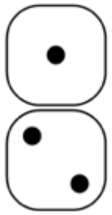
100 x

100 x



-1

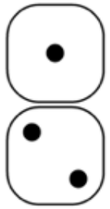
100 x



-1

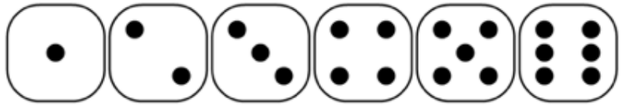
+1

100 x

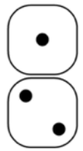


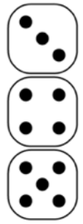
-1

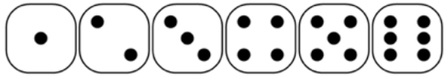
+1



+1 +2 +3 +4 +5 +6

100 x  -1

 +1

+1 +2 +3 +4 +5 +6

- Can't go below step 0
- 0.1 % chance of falling down the stairs
- Bet: you'll reach step 60

How to solve?

- Analytical
- Simulate the process
  - Hacker statistics!

Random generators

```
import numpy as np
np.random.rand()      # Pseudo-random numbers
```

```
0.9535543896720104    # Mathematical formula
```

```
np.random.seed(123)    # Starting from a seed
np.random.rand()
```

```
0.6964691855978616
```

```
np.random.rand()
```

```
0.28613933495037946
```

Random generators

```
np.random.seed(123)
np.random.rand()
```

```
0.696469185597861    # Same seed: same random numbers!
```

```
np.random.rand()     # Ensures "reproducibility"
```

```
0.2861393349503794 6
```

Coin toss

`game.py`

```
import numpy as np
np.random.seed(123)
coin = np.random.randint(0,2)  # Randomly generate 0 or 1
print(coin)
```

```
0
```

Coin toss

`game.py`

```
import numpy as np
np.random.seed(123)
coin = np.random.randint(0,2)  # Randomly generate 0 or 1
print(coin)
if coin == 0:
    print("heads")
else:
    print("tails")
```

```
0
heads
```

Let's practice!

INTERMEDIATE PYTHON

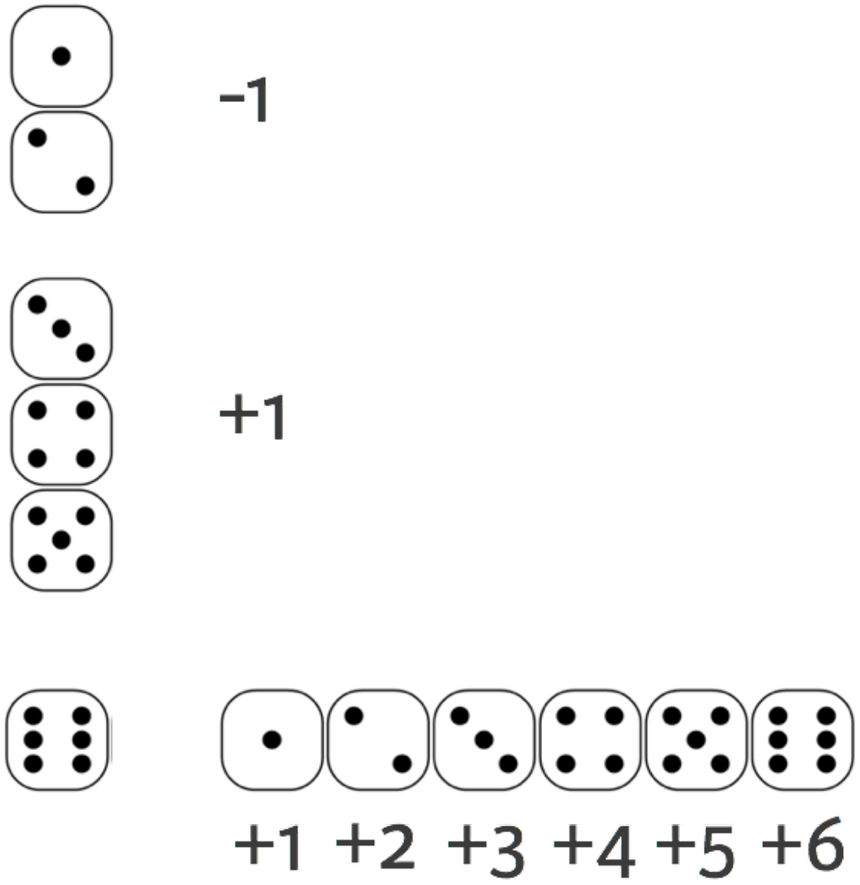
Random Walk

INTERMEDIATE PYTHON

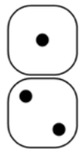
Hugo Bowne-Anderson
Data Scientist at DataCamp

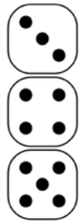
datacamp

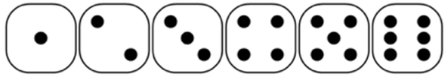
Random Step



Random Walk

100 x  -1

 +1

+1 +2 +3 +4 +5 +6

Known in Science

- Path of molecules
- Gambler's financial status

Heads or Tails

`headtails.py`

```
import numpy as np
np.random.seed(123)
outcomes = []
for x in range(10) :
    coin = np.random.randint(0, 2)
    if coin == 0 :
        outcomes.append("heads")
    else :
        outcomes.append("tails")
print(outcomes)
```

```
['heads', 'tails', 'heads', 'heads', 'heads',
 'heads', 'heads', 'tails', 'tails', 'heads']
```

Heads or Tails: Random Walk

`headtailsrw.py`

```
import numpy as np
np.random.seed(123)
tails = [0]
for x in range(10) :
    coin = np.random.randint(0, 2)
    tails.append(tails[x] + coin)
print(tails)
```

```
[0, 0, 1, 1, 1, 1, 1, 1, 2, 3, 3]
```

5. Heads or Tails: Random Walk
You could turn this example into a random walk by tracking the -total- number of tails while you're simulating the game. In this case, you start by creating a list, tails, that already contains the number 0, because at the start, you haven't thrown any tails. Then you again start a for loop that runs 10 times, using the range function. In there, you again generate a random number. Instead of the if-else structure, you can simplify things. If coin is 0, so heads, the number of tails you've thrown shouldn't change. If a 1 is generated, the number of tails should be incremented with 1. This means that you can simply add coin to the previous number of tails, and add this count to the list with append. Finally, you again print the list tails. After running this script, a list with 11 elements will be printed out. The final element in this list tells you how often tails was thrown.

Step to Walk

```
outcomes
```

```
['heads', 'tails', 'heads', 'heads', 'heads',
 'heads', 'heads', 'tails', 'tails', 'heads']
```

```
tails
```

```
[0, 0, 1, 1, 1, 1, 1, 1, 2, 3, 3]
```

6. Step to Walk
If you compare the output of the first script to the output of the second script, you can see that the numbers in the tails list are incremented by one each time you threw tails. This is exactly how a bunch of random steps are converted into a random walk.

Let's practice!

INTERMEDIATE PYTHON

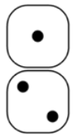
Distribution

INTERMEDIATE PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

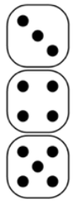
datacamp

Distribution

100 x  -1

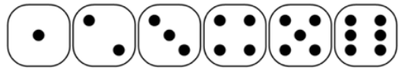
Each random walk has an end point

Simulate 10,000 times: 10,000 end points

 +1

Distribution!

Calculate chances!



+1 +2 +3 +4 +5 +6

Random Walk

`headtailsrw.py`

```
import numpy as np
np.random.seed(123)
tails = [0]
for x in range(10) :
    coin = np.random.randint(0, 2)
    tails.append(tails[x] + coin)
```

100 runs

`distribution.py`

```
import numpy as np
np.random.seed(123)
final_tails = []
for x in range(100) :
    tails = [0]
    for x in range(10) :
        coin = np.random.randint(0, 2)
        tails.append(tails[x] + coin)
    final_tails.append(tails[-1])
print(final_tails)
```

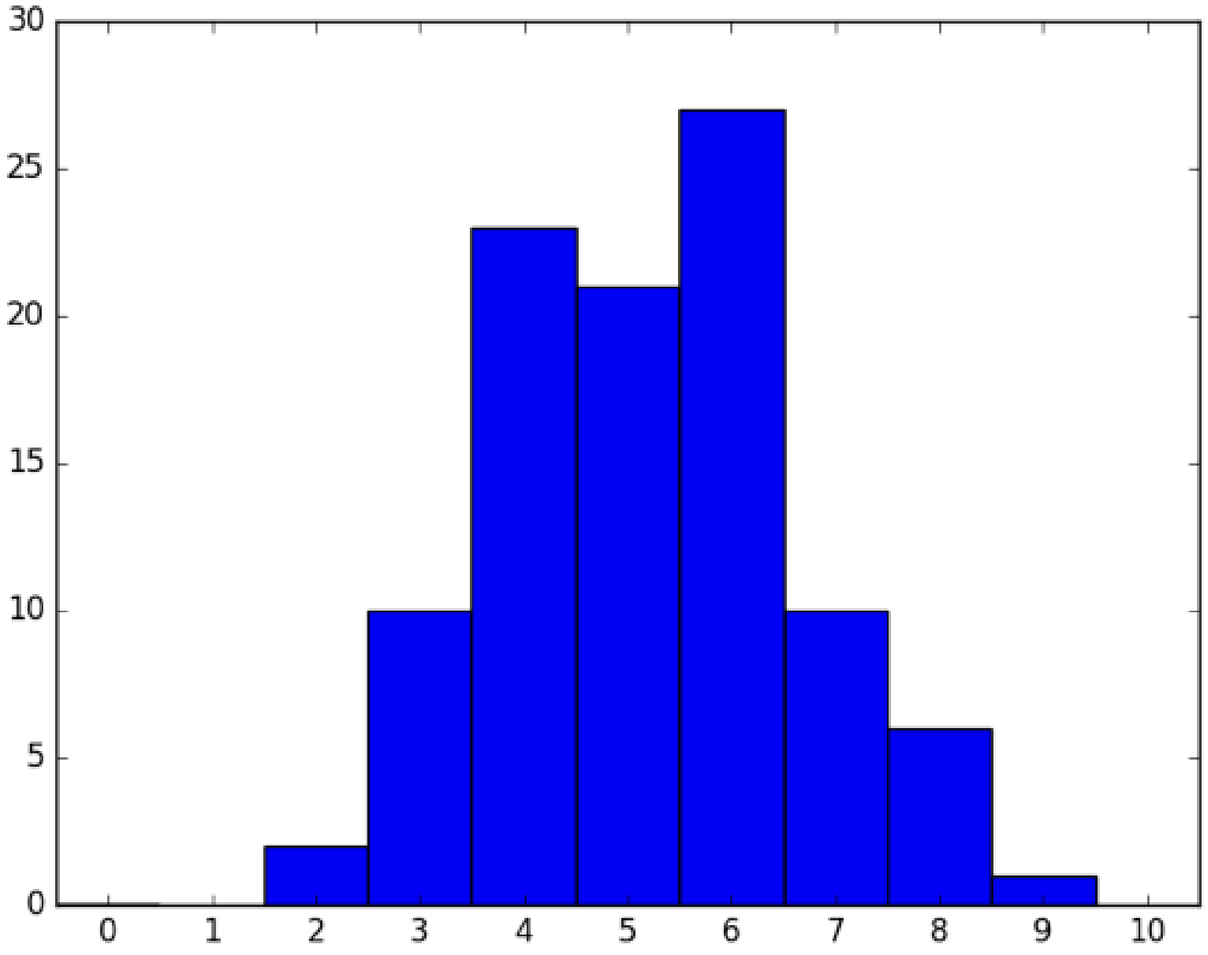
```
[3, 6, 4, 5, 4, 5, 3, 5, 4, 6, 6, 8, 6, 4, 7, 5, 7, 4, 3, 3, ..., 4]
```

Histogram, 100 runs

`distribution.py`

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(123)
final_tails = []
for x in range(100) :
    tails = [0]
    for x in range(10) :
        coin = np.random.randint(0, 2)
        tails.append(tails[x] + coin)
    final_tails.append(tails[-1])
plt.hist(final_tails, bins = 10)
plot.show()
```

Histogram, 100 runs

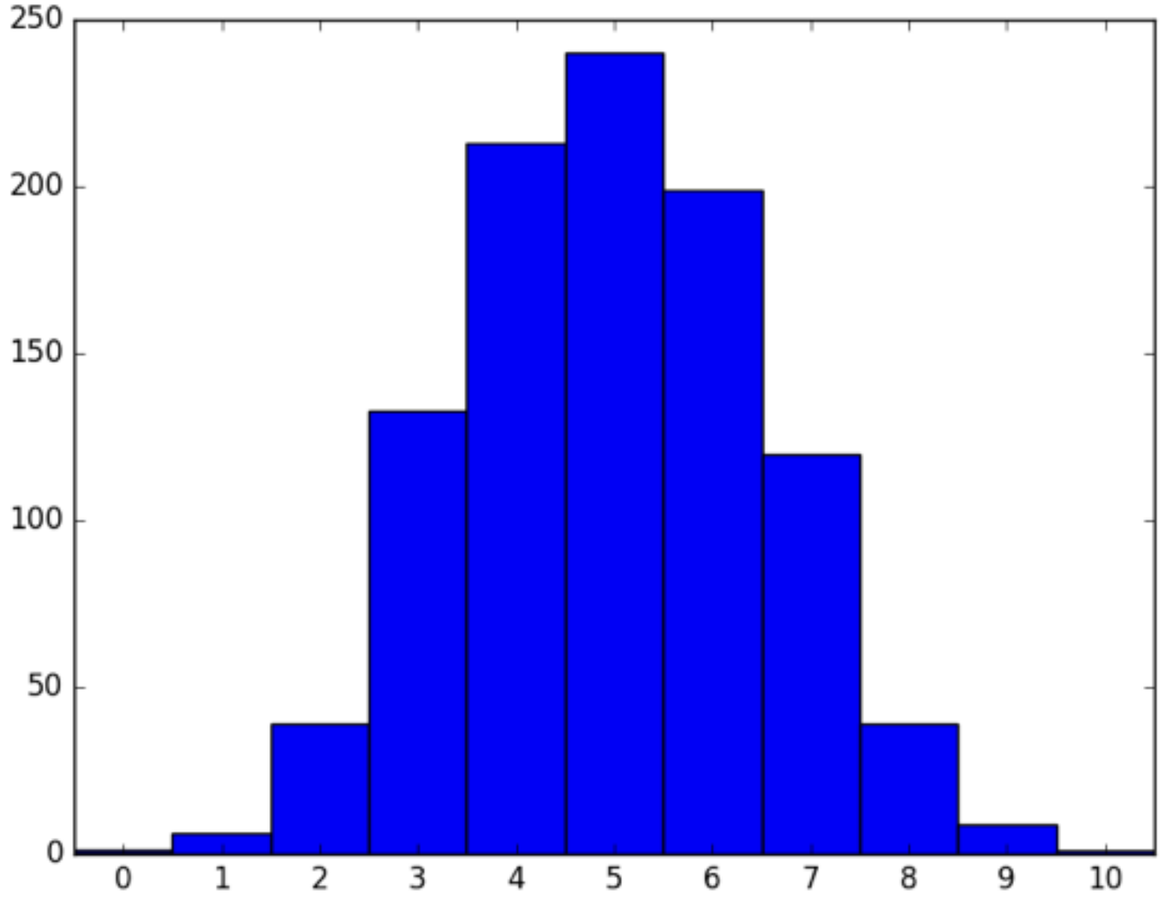


Histogram, 1,000 runs

`distribution.py`

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(123)
final_tails = []
for x in range(1000) : # <--
    tails = [0]
    for x in range(10) :
        coin = np.random.randint(0, 2)
        tails.append(tails[x] + coin)
    final_tails.append(tails[-1])
plt.hist(final_tails, bins = 10)
plot.show()
```

Histogram, 1,000 runs

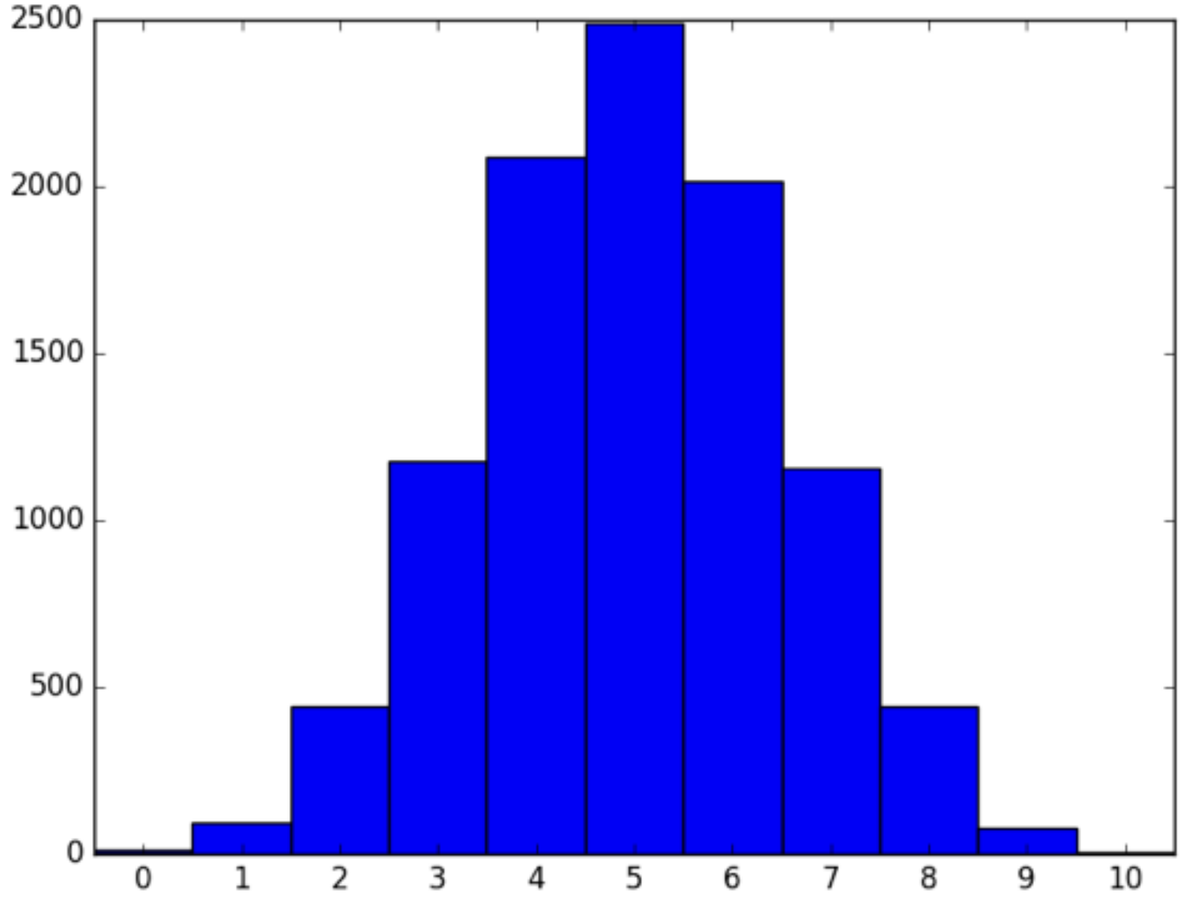


Histogram, 10,000 runs

`distribution.py`

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(123)
final_tails = []
for x in range(10000) : # <--
    tails = [0]
    for x in range(10) :
        coin = np.random.randint(0, 2)
        tails.append(tails[x] + coin)
    final_tails.append(tails[-1])
plt.hist(final_tails, bins = 10)
plot.show()
```

Histogram, 10,000 runs



Let's practice!

INTERMEDIATE PYTHON