CSCE 110: Programming I

David Kebo Hougninou

Basics of Python

Simulation
Example: flip a fair coin

A fair coin has a 50% chance of landing heads up and a 50% chance of landing tails up.

Suppose you flip the coin 3 times independently.

What is the probability that the coin lands heads up, then tails up, then heads up?

Solution

Let’s calculate the probability of this event:

Probability of landing heads up \( P_{\text{heads}} = \frac{1}{2} \)

Probability of landing tails up \( P_{\text{tails}} = \frac{1}{2} \)

\[ P_{\text{heads, tails, heads}} = P_{\text{heads}} \times P_{\text{tails}} \times P_{\text{heads}} \]

\[ = \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \]

\[ = \frac{1}{8} \]
Flip a fair coin

A fair coin has a 50% chance of landing heads up and a 50% chance of landing tails up.

Suppose you flip the coin 3 times independently.

What is the probability that the coin lands heads up exactly twice?

Solution

Let's calculate the probability of this event:
Probability of landing heads up $P_{\text{heads}} = \frac{1}{2}$
Probability of landing tails up $P_{\text{tails}} = \frac{1}{2}$
Solution

There are 8 possible events overall:
H H H, H H T, H T H, T H H, H T T, T H T, T T H, T T T

The probability of one event is 1/8

There are 3 events where the coin lands heads up exactly twice:
H H H, H H T, H T H, T H H, H T T, T H T, T T H, T T T

\[ P = P_{HHT} + P_{HTH} + P_{THH} = 3 \times \frac{1}{8} = \frac{3}{8} \]

What is a simulation?

Simulation is the imitation of the operation of a real-world process or system over time.

\[ P_{\text{heads}} = ? \]
\[ P_{\text{tails}} = ? \]

We can build a system to simulate the flip of a coin (based on a model) and then count the number of heads and tails.

Simulations can demand more than one run (several runs).
What is probability

Probability is the branch of mathematics that studies the possible outcomes of events.

The word probability means the chance that an event (or multiple events) will occur on a linear scale from 0 (impossibility) to 1 (certainty), or as a percentage from 0 to 100%.

e.g.:
If there are 5 pens on my desk, 4 of which are blue, then the chances of picking a blue pen is 4 out of 5.

Example

Write a program that simulates the number of heads and tails that appear using a fair coin.

The input is the number of flips
1. How many variables do we need?
2. What is the output of the program?
Example: flowchart

```
number_tosses = int(input())
count = 0; heads = 0 ; tails = 0

if count < number:
    result = flip_coin()
    if result == 'heads':
        heads += 1
    else:
        tails += 1
    count += 1
else:
    print("Number of tosses", count)
    print("Number of heads", heads)
    print("Number of tails", tails)
```

Solution

```
import random

def flip_coin():
    return random.choice(['heads', 'tails'])

def main():
    coin_tosses = int(input("Enter the number of coin tosses: "))
    heads = 0
    tails = 0
    for i in range(0, coin_tosses):
        result = flip_coin()
        if result == 'heads':
            heads += 1
        else:
            tails += 1
    heads_percent = round(float(heads) / coin_tosses * 100, 2)
    tails_percent = round(float(tails) / coin_tosses * 100, 2)
    print(f"Number of tosses: {coin_tosses}")
    print(f"Number of heads: {heads}, {heads_percent}%")
    print(f"Number of tails: {tails}, {tails_percent}%")

main()
```
Example

Write a program that simulates the number of heads and tails that appear using an **unfair** coin.

The number of flips is an input data.

\[ P_{\text{heads}} > P_{\text{tails}} \text{ or } P_{\text{heads}} < P_{\text{tails}} \]

What is the output of the program?

Flowchart
Example: rolling a die

Write a program that calculates on average, the number of times to roll a die before all six numbers turn up.
Solution

Simulate the roll of a die: `random.randint(1,6)`
Save the numbers that turned up in a list
Test if all numbers have turned up: `sum(list)`
Saving the number of rolls: `number_of_rolls`
Computing the average: `number_of_rolls / number_of_trials`

Before all faces have turned up, we have:

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

When 2 have turned up, we have:

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

When 4 have turned up, we have:

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

After all faces have turned up, we have:

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>
Solution

```python
import random

def roll_die():
    return random.randint(1, 6)

def roll_all_numbers():
    rolled_value = [0]*7  # rolled_value[0] is not used
    number_rolls = 0
    while sum(rolled_value) != 6:
        number = roll_die()
        rolled_value[number] = 1
        number_rolls += 1
    return number_rolls

def conduct_simulation(reps):
    total_rolls = 0
    for i in range(0, reps):
        total_rolls += roll_all_numbers()
    print(f"Repetitions: {reps}"
    print(f"Average rolls: {round(float(total_rolls) / reps, 1)} \n")

def main():
    for experiment in (1, 10, 100, 10**3, 10**4, 10**5):
        conduct_simulation(experiment)

main()
```

Solution using sets

```python
import random

def roll_die():
    return random.randint(1, 6)

def roll_all_numbers():
    rolled_values = set()
    number_rolls = 0
    while len(rolled_values) != 6:
        number = roll_die()
        rolled_values.add(number)
        number_rolls += 1
    return number_rolls

def conduct_simulation(reps):
    total_rolls = 0
    for i in range(0, reps):
        total_rolls += roll_all_numbers()
    print(f"Repetitions: {reps}"
    print(f"Average rolls: {round(float(total_rolls) / reps, 1)} \n")

def main():
    for experiment in (1, 10, 100, 10000, 100000):
        conduct_simulation(experiment)

main()
```
Example: rolling a die

Let's assume we roll six dice at the same time.

On average, how many times do you need to roll six dice before each of them has a different number?

Solution using sets

```python
import random

def roll_six_dice():
    roll = []
    for die in range(6):
        roll += [random.randint(1, 6)]
    return roll

def roll_all_numbers():
    number_rolls = 0
    roll = set()
    while len(roll) != 6:
        roll = set(roll_six_dice())
        number_rolls += 1
    return number_rolls

def conduct_simulation(repetitions):
    total_rolls = 0
    for i in range(repetitions):
        total_rolls += roll_all_numbers()
    print(f"Average rolls: {round(float(total_rolls) / repetitions, 1)}")

def main():
    for experiment in [10, 10**2, 10**3, 10**4, 10**5, 10**6]:
        conduct_simulation(experiment)
main()
```