Objectives:

- To compute the relative frequency of an event.
- To be familiar with properties of the relative frequency.

Submission Instructions

After you have completed the notebook, export it as pdf for submission. You can do this in one of two ways:

1. Go to File, click Download as, click PDF via LaTeX (.pdf).

2. Go to File, click Download as, click HTML (.html), then convert the html file to pdf file.

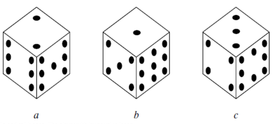
Import useful libraries

In [1]:
```
import numpy as np
from numpy import random
import matplotlib.pyplot as plt
```

Description

Suppose you are playing a game. In this game,

1. There are 3 fair dice a, b and c as shown below.



a b c

2. The number of dots on the 3 dice are different and the number on one face of each die is the same as the number on the opposite face. 3. Each of the two players selects one die and roll it once. The player with lower value loses.

Suppose that the two players, Alice and Bob, selected die a and b, respectively. Below, we will simulate this game $n = 10000$ times and estimate probabilities.

The class below creates a die, and an associated function roll, which returns the result of one roll of the die. To create the die, you pass in a list containing the number of dots on each side of the die. If you do not pass in anything, it defaults to a fair six sided die.

In [2]:
```
class die:
    '''class to simulate the rolling of a die'''

    def __init__(self, dots = [1, 2, 3, 4, 5, 6]):
        '''initialize the die'''
        self.dots = dots

    def roll(self):
        '''roll the die'''
        return random.choice(self.dots)
```

The following code creates a fair six sided die, and rolls it 10 times. Note that python starts indexing from 0, so range(10) returns a list of integers from 0 to 9.

In [3]:
```
# create fair die
fair_die = die()

# roll it 10 times
for i in range(10):
    print(f'The result of roll {i} is {fair_die.roll()}')
```

```
The result of roll 0 is 6
The result of roll 1 is 2
The result of roll 2 is 5
The result of roll 3 is 3
The result of roll 4 is 3
The result of roll 5 is 6
The result of roll 6 is 3
The result of roll 7 is 2
The result of roll 8 is 4
The result of roll 9 is 3
```

The following code creates Alice's and Bob's dice, rolls each die once, and checks whether Alice or Bob wins. You can re-run the cell several times to play multiple games.

In [4]:
```
#create Alice's and Bob's dice
Alice = die([2, 6, 8])
Bob = die([3, 5, 7])

# roll the dice
Alice_roll = Alice.roll()
Bob_roll = Bob.roll()

# print the results
print(f'Alice rolled {Alice_roll}.')
print(f'Bob rolled {Bob_roll}.')
if Alice_roll > Bob_roll:
    print ('Alice wins!')
else:
    print('Bob wins!')
```

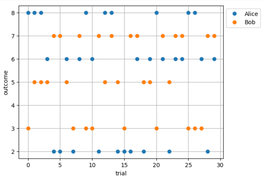
```
Alice rolled 8.
Bob rolled 7.
Alice wins!
```

The following code plots the results of 30 rolls of Alice's and Bob's dice.

In [5]:
```
# roll each die 30 times
trials = range(30)
# method 1
Alice_rolls = []
for i in trials:
    Alice_rolls.append(Alice.roll())
# method 2
Bob_rolls = [Bob.roll() for i in trials]
# note there are many possible methods to do this

# create plot
fig, ax = plt.subplots() # create figure to hold plots
ax.plot(trials, Alice_rolls, 'o', label='Alice') # plot Alice's rolls
ax.plot(trials, Bob_rolls, 'o', label='Bob') # plot Bob's rolls

# label the plots
plt.xlabel('trial')
plt.ylabel('outcome')
plt.grid()
plt.legend(bbox_to_anchor = (1,1))
plt.show()
```



The following code rolls Alice's die 10,000 times, and computes the relative frequency of each of the three outcomes. Note that the resulting probabilities are all close to 1/3?

In [6]:
```
total = 10000 # total number of times to roll the die

# roll the die "total" times
trials = range(total)
rolls = [Alice.roll() for i in trials]

# the total number of outcomes is the number of faces with different dots
number_of_outcomes = np.size(Alice.dots)
# for each outcome
for i in range(number_of_outcomes):
    # count the number of times the outcome occurs
    count = np.sum( np.equal( rolls, Alice.dots[i] ) )
    # calculate the relative frequency
    rf = count/total
    # print result
    print(f'The relative frequency of the outcome {Alice.dots[i]} is {rf}.')
```

```
The relative frequency of the outcome 2 is 0.3325.
The relative frequency of the outcome 6 is 0.3297.
The relative frequency of the outcome 8 is 0.3378.
```

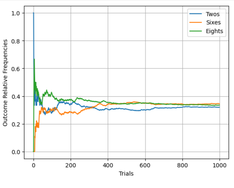
Part a: Convergence of relative frequency

Generate plots of the relative frequencies of the three outcomes of Alice's die versus the number of trials, which ranges from 1 to 1000

In [13]:
```
trial_range = range(1000)
# 2s
outcome_one = 0 # number of times 2s show up
outcome_one_freq = [] # array of freq
# 6s
outcome_two = 0 # number of times 6s show up
outcome_two_freq = [] # array of freq
# 8s
outcome_three= 0 # number of times 8s show up
outcome_three_freq = [] # array of freq
for i in trial_range:
    # roll Dice
    alice_temp = Alice.roll()
    #record outcomes
    if alice_temp == Alice.dots[0]:
        outcome_one+=1
    elif alice_temp == Alice.dots[1]:
        outcome_two+=1
    elif alice_temp == Alice.dots[2]:
        outcome_three+=1
    #compute and store outcome relative frequencies
    outcome_one_freq.append(outcome_one/(n+1))
    outcome_two_freq.append(outcome_two/(n+1))
    outcome_three_freq.append(outcome_three/(n+1))

# create plot
fig, ax = plt.subplots() # create figure to hold plots
ax.plot(trial_range, outcome_one_freq, label="Twos") # plot 2
ax.plot(trial_range, outcome_two_freq, label="Sixes") # plot 6
ax.plot(trial_range, outcome_three_freq, label='Eights') # plot 8

# label the plots
plt.xlabel('Trials')
plt.ylabel('Outcome Relative Frequencies')
plt.grid()
plt.legend(bbox_to_anchor = (1,1))
plt.show()
```



Part b: Play the game repeatedly.

Play the game 10,000 times and calculate the relative frequency that Alice wins. Who is more likely to win this game, Alice or Bob?

In [16]:
```
# your code below
times = 10000
run_times = range(times)
alice_freq = [] # array of freq
alice_wins = 0 # number of times alice wins
bob_freq = [] # array of freq
bob_wins = 0 # number of times bob wins
for i in run_times:
    # roll dice once for both players
    alice_temp = Alice.roll()
    bob_temp = Bob.roll()
    # compare values, record wins for each player
    if alice_temp > bob_temp:
        alice_wins +=1 # number of times alice wins
    elif bob_temp>alice_temp:
        bob_wins+=1
    # compute and store relative freq for both players
    alice_freq.append(alice_wins/(i+1))
    bob_freq.append(bob_wins/(i+1))

#     debug
#     print(alice_temp)
#     print(bob_temp)
#     print(alice_wins)
#     print(bob_wins)
#     print(alice_freq[j])
# print(len(alice_freq))


# # create plot
# fig, ax = plt.subplots() # create figure to hold plots
# ax.plot(run_times, alice_freq, label='Alices Relative Frequency') # plot relative frequency of Alice wins
# ax.plot(run_times, bob_freq, label='Bobs Relative Frequency') # plot bob wins

# # label the plots
# plt.xlabel('Trials')
# plt.ylabel('Relative Frequency')
# plt.grid()
# plt.legend(bbox_to_anchor = (1,1))
# plt.show()

print("Relative Frequency for Alice is: " + str(alice_freq[times-1]))
```

```
Relative Frequency for Alice is: 0.5534
```

Part b answer:

In [9]:
```
print("Alice is more likely to win.")
```

```
Alice is more likely to win.
```

Part c: Play the game with two die rolls.

Now assume that Alice and Bob both get two rolls of the die per game and add the two outcomes. The winner is whoever's sum is larger.

Simulate this game 10,000 times and compute the relative frequency that Alice wins. Who is more likely to win this game, Alice or Bob?

In [15]:
```
# your code below
# your code below
run_times1 = range(times)
alice_freq1 = [] # array of freq
alice_wins1 = 0 # number of times alice wins
bob_freq1 = [] # array of freq
bob_wins1 = 0 # number of times bob wins
draws_freq =[] # array of freq
draws=0 # number of times draw
for k in run_times1:
    # sum the two dice rolls
    alice_temp1 = Alice.roll() + Alice.roll()
    bob_temp1 = Bob.roll() + Bob.roll()
    # compare values, record wins for each player, draw possible
    if alice_temp1 > bob_temp1:
        alice_wins1 +=1
    elif bob_temp1>alice_temp1:
        bob_wins1+=1
    else:
        draws+=1
    # compute and store relative frequencies
    draws_freq.append(draws/(k+1))
    alice_freq1.append(alice_wins1/(k+1))
    bob_freq1.append(bob_wins1/(k+1))

# debug
# print(len(draws_freq))
# print(draws)

# # create plot
# fig, ax = plt.subplots() # create figure to hold plots
# ax.plot(run_times1, alice_freq1, label='Alice Win') # plot alice wins
# ax.plot(run_times1, bob_freq1, label='Bob wins') # plot bob wins
# ax.plot(run_times1, draws_freq, label='Draws') # plot draws

# # label the plots
# plt.xlabel('trial')
# plt.ylabel('outcome')
# plt.grid()
# plt.legend(bbox_to_anchor = (1,1))
# plt.show()

print("Relative Frequency for Alice is : " + str(alice_freq1[times-1]))
```

```
Relative Frequency for Alice is : 0.4774
```

Part c answer:

In [11]:
```
print("Alice is more likely to win.")
```

```
Alice is more likely to win.
```