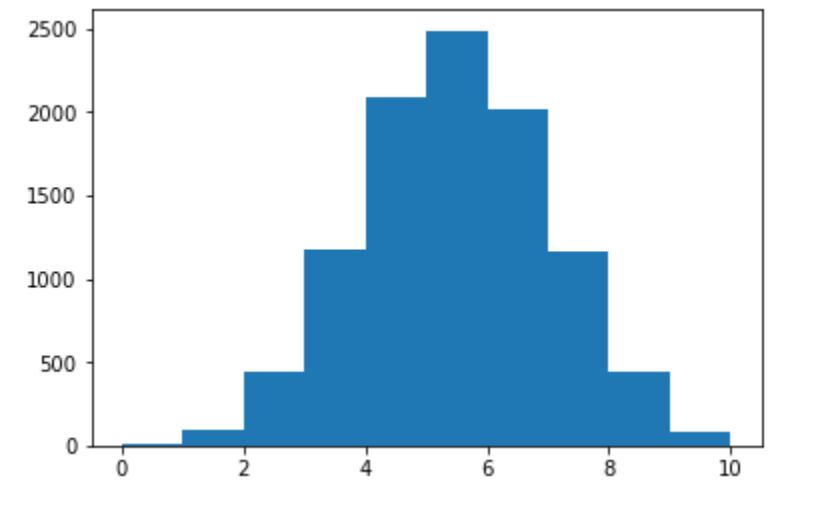
Simulation process

We can use randomness to simulate a game. In python, we can use random package, a sub-package of numpy.

The below code given distribution of tails for a coin flipped 10 times and process is repeated 10000 times

```
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)
plt.show()
```

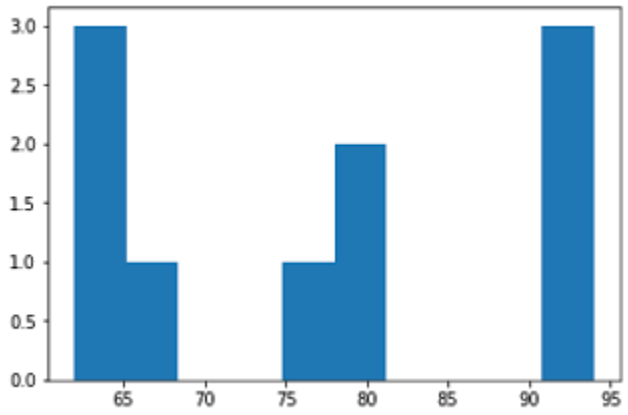


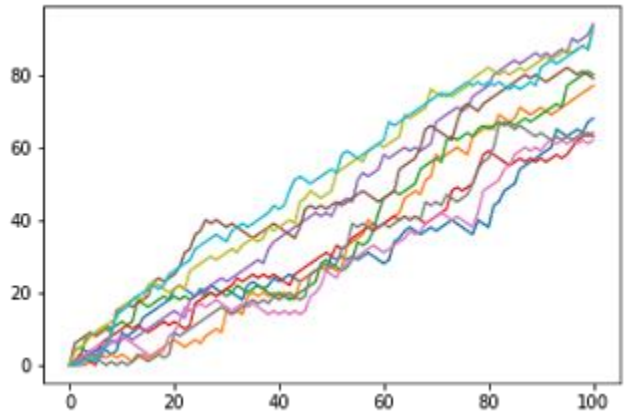
 # we use randint(), also a function of the random package, to generate integers randomly. Here 0 is head and # tail is one. Note that in the code below 0 is included and 2 is not.

In the below problem, I have tried to figure out what are the odds that one reaches 80 steps high on the State Building with each step is based on the result of a dice. Instructions (if dice <=2 , step -1, else if dice <= 5. step +1, else throw dice again) Approach- To get an idea about how big our chances are reaching 80 steps, we can repeatedly simulate the random walk and collect the results.

```
# What are the odds that one reach 60 steps high on the State Buildi
ng
# if dice <=2 , step -1, elseif dice <= 5. step +1, else throw dice again
import matplotlib.pyplot as plt
import numpy as np
np.random.seed(1)
all_walks = []
for i in range(10) :
    random_walk = [0]
    for x in range(100) :
        step = random_walk[-1]
        dice = np.random.randint(1,7)
        if dice <= 2:
            step = max(0, step - 1)
        elif dice <= 5:
            step = step + 1
        else:
            step = step + np.random.randint(1,7)
        random_walk.append(step)
    all_walks.append(random_walk)

# Convert all_walks to Numpy array: np_aw
np_aw=np.array(all_walks)
# Plot np_aw and show
plt.plot(np_aw)
plt.show()
# Clear the figure
plt.clf()

# Transpose np_aw: np_aw_t
np_aw_t = np.transpose(np_aw)

# Plot np_aw_t and show
```

```
plt.plot(np_aw_t)
plt.show()

# look -How important is to transpose numpy array
# Select last row from np_aw_t: ends
ends =np_aw_t[-1]

# Plot histogram of ends, display plot
plt.hist(ends)
plt.show()
print(ends)
np.mean(ends >60)
```






```
[68 77 80 63 94 79 62 64 93 93]
```

Out[11]:

```
1.0
```