```
import numpy as np
import time

print(np.random.randint(1, 7, 2))

# Simulacao iterativa
def Dado(n):
    deuCerto = 0 # Inicia quatidade de vezes que o evento oc
     for i in range(n): # Executa n vezes o sorteio dos dados
         d1 = np.random.randint(1, 7) # sorteia dado 1 (número inteiro
aleatório
         d2 = np.random.randint(1, 7) # sorteia dado 2 (número inteiro
aleatório
         if ((d1 == 3) & (d2 == 6)) | ((d1 == 6) & (d2 == 3)): # Testa
se o evento
         deuCerto = deuCerto + 1 # Incrementa quatidade de vezes que o
event
     return deuCerto/n

t1 = time.perf_counter()
probS = Dado(50000)
t2 = time.perf_counter()
print('Probabilidade simulada: {:.4f}'.format(probS))
print('Probabilidade teórica: {:.4f}'.format(2/36))
print('Tempo de simulação: {:.4f}'.format(t2-t1))

n=5 # Quantidade de sorteios
print(np.random.randint(0,2,n))

k = 0
moeda = 0
while (moeda != 1):
    k = k+1
    moeda = np.random.randint(0, 2)
    print(moeda)
print(k)

# Simulacao interativa
def Moeda(n):
 # Inicia quatidade de vezes que o evento ocorreu
    deuCerto = 0
    for i in range(n):
         k = 0
         moeda = 0
         while (moeda != 1):
             k = k+1
             moeda = np.random.randint(0, 2)
 #print(moeda)
 #print(k)
 # Executa n vezes o sorteio da moeda
```

```
 # Sortear da moeda até sair cara
 # Contar quantas vezes sorteou
 # Se a quantidade de sorteios for par, incrementa deuCerto
(quantidade de v
     if k % 2 == 0: # Substituir esse comando
    return deuCerto = d

import time
t1 = time.perf_counter()
probS = Moeda(50000)
t2 = time.perf_counter()
print('Probabilidade simulada: {:.4f}'.format(probS))
print('Probabilidade teórica: {:.4f}'.format(1/3))
print('Tempo de simulação: {:.4f}'.format(t2-t1))

def aniverS(tGrupo, nSim):
    deuCerto = 0
    for i in range(nSim):
 # sorteia grupo com tGrupo pessoas
        grupo = np.random.randint(1, 366, tGrupo)
    # se duas ou mais pessoa fazem aniver na mesma data
        if len(np.unique(grupo)) != len(grupo): # Substituir esse
comando
            deuCerto = deuCerto + 1
    return(deuCerto/nSim)

def aniverT(tGrupo):
    x = np.arange(365, 365 - tGrupo, -1, dtype = float)
 # print(x)
    return(1 - np.prod(x)/(365**tGrupo))

def aniverS(tGrupo, nSim):
    deuCerto = 0
    for i in range(nSim):
 # sorteia grupo com tGrupo pessoas
        grupo = np.random.randint(1, 366, tGrupo)
 # se duas ou mais pessoa fazem aniver na mesma data
        if True:
            deuCerto = deuCerto + 1
    return(deuCerto/nSim)

probT = aniverT(40)
t1 = time.perf_counter()

probS = aniverS(40, 10000)
t2 = time.perf_counter()
print('Probabilidade simulada: {:.4f}'.format(probS))
print('Probabilidade teórica: {:.4f}'.format(probT))
print('Tempo de simulação: {:.4f}'.format(t2-t1))
```