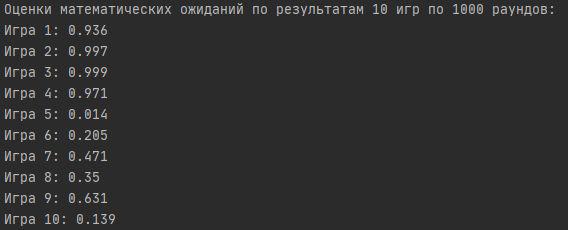
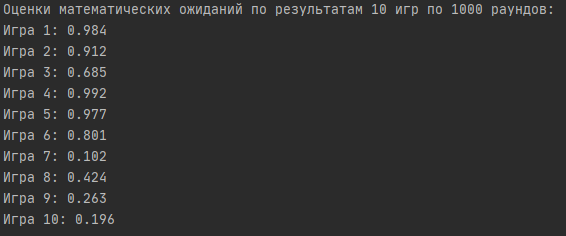
**Задача 3**. Два игрока по очереди бросают монету. Если выпадает «орел», то первый игрок добавляет в свой актив одно очко. Если выпадает «решка», то очко добавляется второму игроку. Разница очков показывает игрока, выигрывающего в текущий момент времени. Построить модель процесса бросания монеты и вычисления разницы очков. Провести 10 испытаний по 1000 исходов от подбрасываний монет в каждом и оценить математическое ожидание периода выигрывания первого игрока. Периодом выигрывания будем называть число шагов между подбрасываниями монеты, в каждом из которых один из игроков имеет большее число очков, чем его соперник.

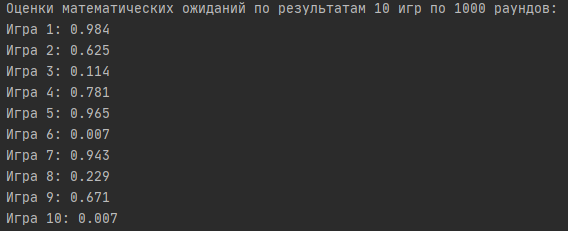
**Код:**

from enum import Enum  
import random  
  
  
GAMES = 10 # число испытаний  
ROUNDS = 1000 # число исходов  
  
  
class Coin(Enum):  
 HEAD = 0 # орёл  
 TAIL = 1 # решка  
  
  
def tossing():  
 return Coin(random.randint(0, 1))  
  
  
def game():  
 # число очков игроков  
 p1 = 0 # выигрывает, когда выпадает орёл  
 p2 = 0  
 winning\_period\_p1 = 0 # период выигрывания игрока 1  
 winning\_period\_p2 = 0  
 for \_ in range(ROUNDS):  
 if tossing() == Coin.HEAD:  
 p1 += 1  
 else:  
 p2 += 1  
 if p1 > p2:  
 winning\_period\_p1 += 1  
 elif p2 > p1:  
 winning\_period\_p2 += 1  
 return winning\_period\_p1  
  
  
def estimate\_expected\_value(sum\_x, n):  
 return sum\_x / n  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 results = {}  
 for game\_number in range(1, GAMES + 1):  
 results[game\_number] = estimate\_expected\_value(game(), ROUNDS)  
  
 print(f'Оценки математических ожиданий по результатам {GAMES} игр по {ROUNDS} раундам:')  
 for i in results:  
 print(f'Игра {i}: {results[i]}')

**Результат:**

****

****

****