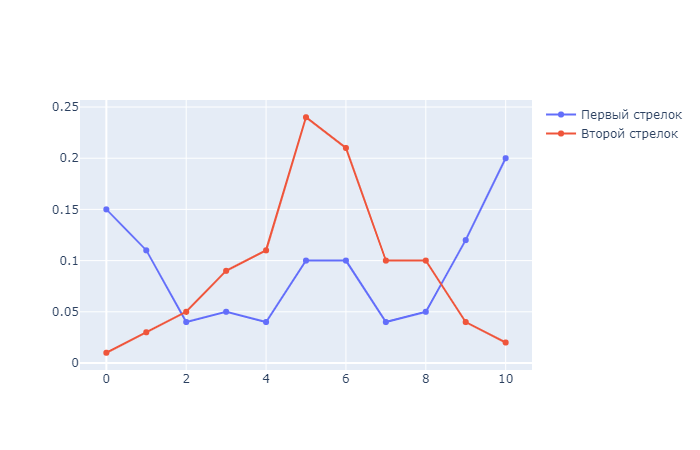
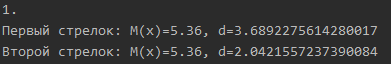
**Лабораторная работа №2**

Задача №1

Постановка задачи: Построить многоугольник распределения вероятностей, найти математическое ожидание и среднее квадратичное отклонение по известным законам распределения вероятности попадания в мишень для двух стрелков Х и Y.

Результаты:





Код программы:

import math  
import plotly.graph\_objects as go  
  
  
def expected\_value(x: list, p: list, square=False) -> float:  
 if square: # Square  
 x = [x\_i\*x\_i for x\_i in x]  
 m\_x = sum((x\_i \* p\_i for x\_i, p\_i in zip(x, p)))  
 return m\_x  
  
  
def dispersion(x: list, p: list) -> float:  
 m\_x = expected\_value(x, p)  
 m\_x2 = expected\_value(x, p, True)  
 d\_x = m\_x2 - m\_x\*m\_x  
 return d\_x  
  
  
def standart\_deviation(x: list, p: list) -> float:  
 d\_x = dispersion(x, p)  
 d = math.sqrt(d\_x)  
 return d  
  
  
def task1():  
 p = [0.15, 0.11, 0.04, 0.05, 0.04, 0.1, 0.1, 0.04, 0.05, 0.12, 0.2]  
 q = [0.01, 0.03, 0.05, 0.09, 0.11, 0.24, 0.21, 0.1, 0.1, 0.04, 0.02]  
 x = list(range(len(p))) # 0..10  
  
 m\_x1 = expected\_value(x, p)  
 d\_x1 = standart\_deviation(x, p)  
 m\_x2 = expected\_value(x, q)  
 d\_x2 = standart\_deviation(x, q)  
 print(f'1.\n'  
 f'Первый стрелок: M(x)={m\_x1}, d={d\_x1}\n'  
 f'Второй стрелок: M(x)={m\_x2}, d={d\_x2}')  
  
 fig = go.Figure()  
 fig.add\_trace(go.Scatter(x=x, y=p,  
 mode='lines+markers',  
 name='Первый стрелок'))  
 fig.add\_trace(go.Scatter(x=x, y=q,  
 mode='lines+markers',  
 name='Второй стрелок'))  
 fig.show()

v.draw\_empiric\_dist\_func(3, 1, 2, postfix='рабочих')  
 v.draw\_hist(3, 1, 3)  
  
 ContinuousVS.show()

Задача №2

Постановка задачи:

В лотерее разыгрывается: автомобиль стоимостью 5000 ден. ед., 4 телевизора стоимостью 250 ден. ед., 5 видеомагнитофонов стоимостью 200 ден. ед. Всего продается 1000 билетов. Вычислить математическое ожидание случайной величины X – средний выигрыш на билет. Определить, какова должна быть стоимость билетов, чтобы устроители лотерее не остались в проигрыше.

Результаты:



Код программы:

def task2():  
 prizes = [(1, 5000), (4, 250), (5, 200)]  
 n = 1000 # Всего билетов  
  
 t = sum((n for n, \_ in prizes)) # Всего призов  
  
 p = [p\_i / t for p\_i, \_ in prizes] # Вероятности  
 x = [x\_i for \_, x\_i in prizes] # Возможные значения  
  
 m\_x = expected\_value(x, p)  
 d = standart\_deviation(x, p)  
  
 price = (m\_x + d) / n  
 print(f'2.\n'  
 f'Математическое ожидание: {m\_x}\n'  
 f'Стоимость билета: {price}')

Задача №3

Постановка задачи: Найти математическое ожидание и дисперсию величины, заданной следующим распределением.



Результаты:



Код программы:

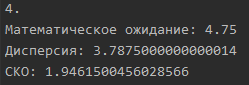
def task3():  
 x = [2, 4, 7, 10, 12]  
 p = [0.1, 0.2, 0.4, 0.2, 0.1]  
  
 m\_x = expected\_value(x, p)  
 d\_x = dispersion(x, p)  
 print(f'3.\n'  
 f'Математическое ожидание: {m\_x}\n'  
 f'Дисперсия: {d\_x}')

Задача №4

Постановка задачи: Найти математическое ожидание, дисперсию и среднее квадратичное отклонение величины, заданной следующим распределением.



Результаты:



Код программы:

def task4():  
 x = [2, 4, 5, 6, 8, 9]  
 p = [0.2, 0.25, 0.3, 0.1, 0.1, 0.05]  
  
 m\_x = expected\_value(x, p)  
 d\_x = dispersion(x, p)  
 d = standart\_deviation(x, p)  
 print(f'4.\n'  
 f'Математическое ожидание: {m\_x}\n'  
 f'Дисперсия: {d\_x}\n'  
 f'СКО: {d}')