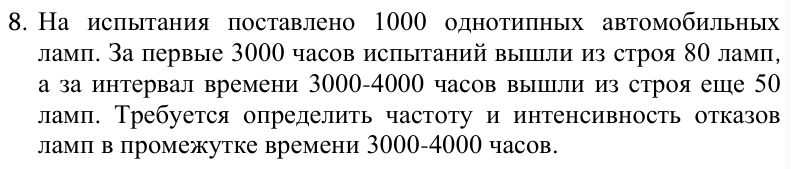
# Лабораторная работа №13-14

Студент: Болдинов Алексей ЭФМО-02-24

**Вариант – 3**

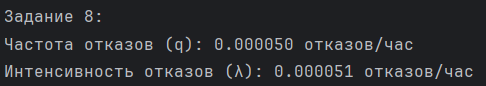
**Задание 1**:



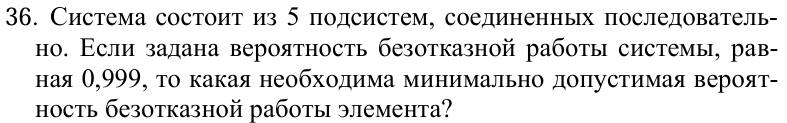
Код для решения поставленной задачи:

# Данные задачи  
total\_lamps = 1000 # Общее количество ламп  
failed\_3000\_4000 = 50 # Число отказов за интервал 3000-4000 часов  
time\_interval = 1000 # Интервал времени (4000 - 3000)  
  
# Расчёт частоты отказов  
failure\_frequency = failed\_3000\_4000 / (total\_lamps \* time\_interval)  
  
# Расчёт среднего числа работающих ламп на интервале  
average\_working\_lamps = total\_lamps - failed\_3000\_4000 / 2  
  
# Расчёт интенсивности отказов  
failure\_intensity = failed\_3000\_4000 / (time\_interval \* average\_working\_lamps)  
  
# Вывод результатов  
print(f"Частота отказов (q): {failure\_frequency:.6f} отказов/час")  
print(f"Интенсивность отказов (λ): {failure\_intensity:.6f} отказов/час")

Вывод программы:



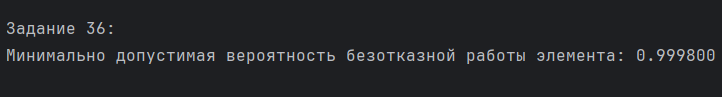
**Задание 2**:



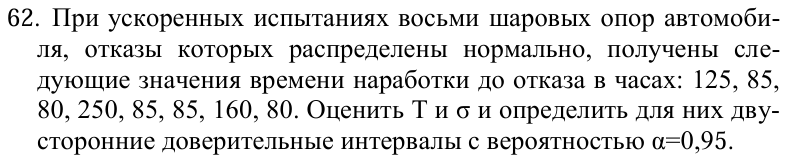
Код для решения поставленной задачи:

# Данные задачи  
P\_s = 0.999 # Вероятность безотказной работы системы  
n = 5 # Количество подсистем  
  
# Вычисление вероятности безотказной работы одной подсистемы  
P\_e = P\_s \*\* (1 / n)  
  
# Вывод результата  
print(f"Минимально допустимая вероятность безотказной работы элемента: {P\_e:.6f}")

Вывод программы:



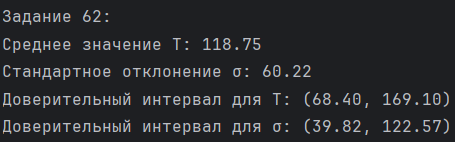
**Задание 3**:



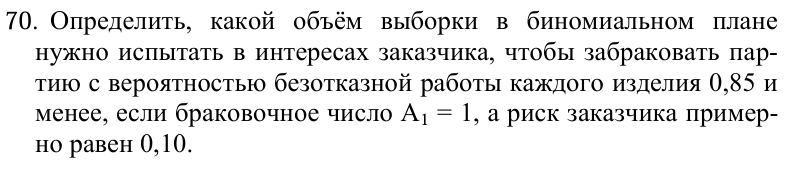
Код для решения поставленной задачи:

import numpy as np  
from scipy.stats import t, chi2  
  
# Данные выборки  
data = [125, 85, 80, 250, 85, 85, 160, 80]  
n = len(data)  
alpha = 0.95 # Уровень доверия  
  
# Вычисляем среднее T и стандартное отклонение σ  
T = np.mean(data)  
sigma = np.std(data, ddof=1)  
  
# Доверительный интервал для T  
t\_critical = t.ppf(1 - (1 - alpha) / 2, df=n - 1)  
T\_low = T - t\_critical \* sigma / np.sqrt(n)  
T\_high = T + t\_critical \* sigma / np.sqrt(n)  
  
# Доверительный интервал для σ  
chi2\_low = chi2.ppf((1 - alpha) / 2, df=n - 1)  
chi2\_high = chi2.ppf(1 - (1 - alpha) / 2, df=n - 1)  
sigma\_low = np.sqrt((n - 1) \* sigma \*\* 2 / chi2\_high)  
sigma\_high = np.sqrt((n - 1) \* sigma \*\* 2 / chi2\_low)  
  
# Результаты  
print(f"Среднее значение T: {T:.2f}")  
print(f"Стандартное отклонение σ: {sigma:.2f}")  
print(f"Доверительный интервал для T: ({T\_low:.2f}, {T\_high:.2f})")  
print(f"Доверительный интервал для σ: ({sigma\_low:.2f}, {sigma\_high:.2f})")

Вывод программы:



**Задание 4**:



Код для решения поставленной задачи:

from scipy.special import comb  
from scipy.stats import binom  
  
# Данные задачи  
p = 0.85 # вероятность безотказной работы  
A1 = 1 # браковочное число  
risk\_customer = 0.10 # риск заказчика  
  
# Функция для вычисления вероятности браковки  
def calculate\_rejection\_probability(n, p, A1):  
 rejection\_probability = sum(comb(n, k) \* (p \*\* k) \* ((1 - p) \*\* (n - k)) for k in range(A1 + 1))  
 return rejection\_probability  
  
# Поиск минимального n  
n = 1  
while True:  
 P\_reject = calculate\_rejection\_probability(n, p, A1)  
 if P\_reject >= risk\_customer:  
 break  
 n += 1  
  
# Результат  
print(f"Минимальный объем выборки: {n}")  
print(f"Вероятность браковки для n={n}: {P\_reject:.4f}")

Вывод программы:

