МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №5

по дисциплине «Теория надежности»

тема: «Без ожидания»

Выполнил: ст. группы ВТ-32

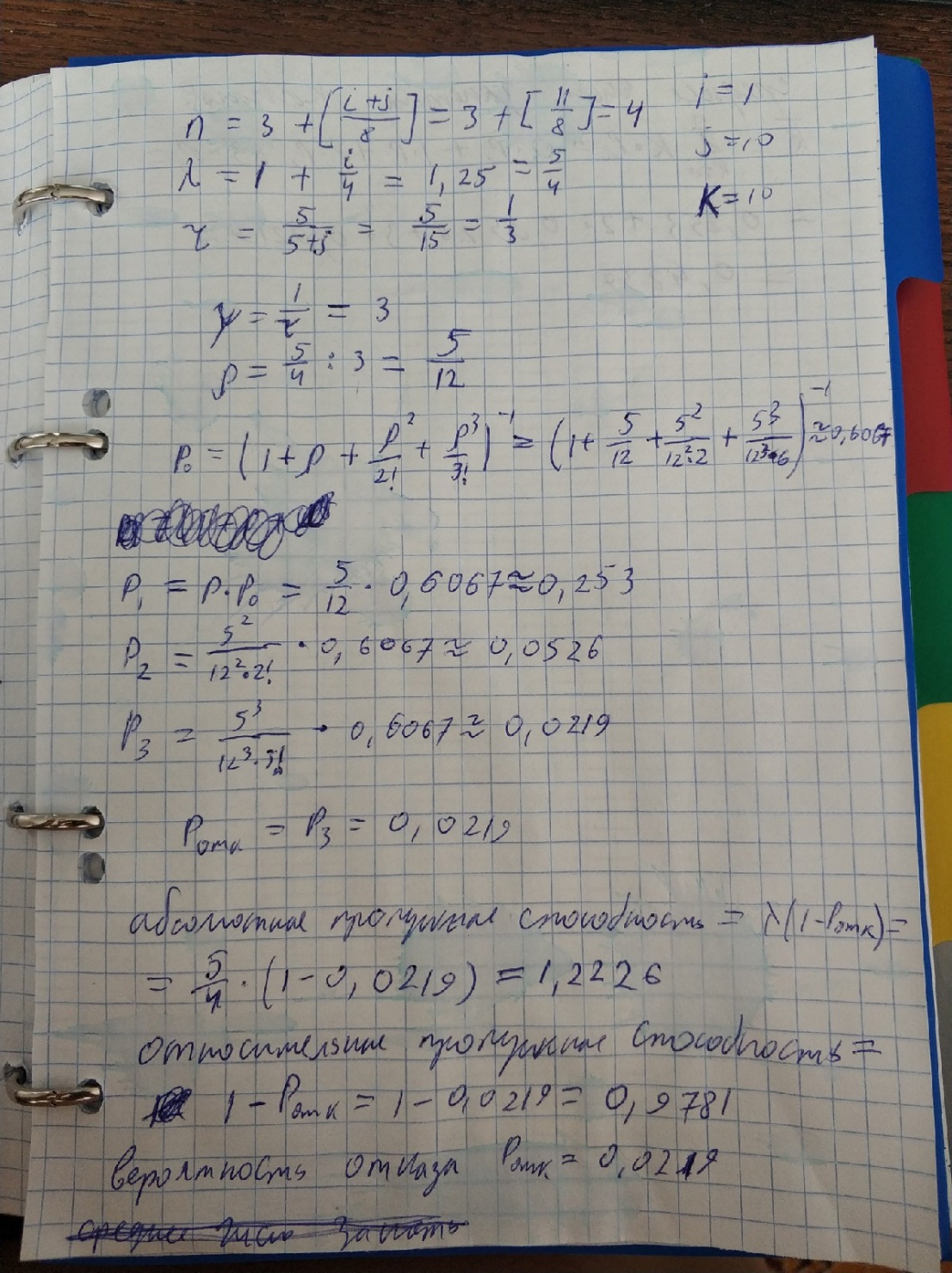
Воскобойников И. С.

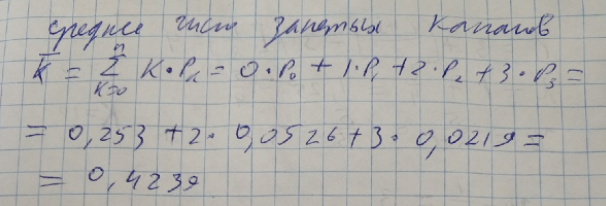
Проверил: Кабалянц П. С.

Белгород 2021 г.

**Задание:** На сервере есть n каналов передачи сообщений. Среднее время обработки сообщений τ минут. На сервер поступают сообщения в среднем количестве λ сообщений в минуту. 1) Определить основные характеристики сервера (абсолютную пропускную способность, т.е. среднее число сообщений, обрабатываемых в единицу времени; относительную пропускную способность, т.е. среднюю долю пришедших сообщений, обрабатываемых системой; вероятность отказа в обработке; среднее число занятых каналов), если сообщение получает отказ при занятости всех каналов. Параметры варианта определяются по формулам: n=3+[(i+j)/8], λ=1+i/4, τ=5/(5+j). Здесь квадратные скобки означают взятие целой части, а i,j - последние цифры зачетки.  
2) Написать программу, которая имитирует поведение сервера и вычисляет его основные характеристики. Сравните результаты.

**Выполнение:**





from random import random  
from math import factorial, exp, pow, sqrt, log  
*# Последние 3 цифры зачетки*i = 1  
j = 10  
k = 10  
  
def random\_request\_time():  
 random\_value = random()  
 *# время через которое придет одна заявка  
 # Экспоненциальное распределение* return - 1 / (5. / 4.) \* log(random\_value)  
  
def random\_respone\_time():  
 random\_value = random()  
 *# время через которое придет одна заявка  
 # Экспоненциальное распределение* return - 1 / (3. / 1.) \* log(random\_value)  
  
def step(current\_time, count\_request, count\_not, busy):  
 current\_time += random\_request\_time()  
 count\_request += 1  
 if busy[0] < current\_time:  
 busy[0] = current\_time + random\_respone\_time()  
 elif busy[1] < current\_time:  
 busy[1] = current\_time + random\_respone\_time()  
 elif busy[2] < current\_time:  
 busy[2] = current\_time + random\_respone\_time()  
 else:  
 count\_not += 1  
 return current\_time, count\_request, count\_not  
  
  
def run(n):  
 busy = [0, 0, 0]  
 count\_request = 0  
 count\_not = 0  
 current\_time = 0  
 while count\_request < n:  
 current\_time, count\_request, count\_not = step(current\_time, count\_request,count\_not,busy)  
 return count\_not,count\_request  
  
  
  
n = 1000  
count\_not, count\_request = run(n)  
  
p\_practice = count\_not/count\_request  
p\_teor = 0.0219  
print(**f"Практическая вероятность отказа =** {p\_practice}**"**)  
print(**f"Теоретическая вероятность отказа =** {p\_teor}**"**)  
print(**f"|K| =** {abs((p\_practice - p\_teor) / (sqrt(p\_practice \* (1-p\_practice) / n)))} **< 1.96**\n**"  
"Согласно критерию сравнения долей гипотезы согласуются и мы правильно посчитали вручную."**)

