**Федеральное государственное образовательное**

**бюджетное учреждение**

**высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ**

**ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ**

**ФЕДЕРАЦИИ»**

**(Финансовый университет)**

**Факультет**

**информационных технологий и анализа больших данных**

**Кафедра «Бизнес-информатика»**

**Домашнее задание № 8**

«Решение задач систем массового обслуживания.»

Студенты группы БИ20-8:

Луканина Полина

Аверкин Никита

Филимонова Арина

Совин Владимир

Горшков Георгий

Киселева Евгения

Руководитель:

Аксенов Дмитрий Андреевич

**Москва 2022**

### **Оглавление**

[Оглавление 2](#_heading=h.30j0zll)

[1.](#_heading=h.1fob9te) Постановка задачи (физическая модель) 5

[2.](#_heading=h.3znysh7) Математическая модель 5

[3.1. Алгоритм в Python 8](#_heading=h.tyjcwt)

[3.1.1. Описание входных данных. 8](#_heading=h.3dy6vkm)

[3.1.2. Описание алгоритма решения 9](#_heading=h.1t3h5sf)

[3.1.3. Описание выходных данных 9](#_heading=h.2s8eyo1)

[4. Вариант использования 12](#_heading=h.17dp8vu)

[5. Архитектура решения 15](#_heading=h.3rdcrjn)

[5.1 Функции считывания информации 15](#_heading=h.26in1rg)

[5.2 Функции обработки информации 16](#_heading=h.35nkun2)

[5.3 Функции вывода информации 24](#_heading=h.44sinio)

[6. Тестирование 25](#_heading=h.2jxsxqh)

[6.1. Тестирование Датасета №1: 25](#_heading=h.z337ya)

[6.1.1 Метод Python: 25](#_heading=h.3j2qqm3)

[6.1.2 Метод Excel: 29](#_heading=h.1y810tw)

[6.2. Тестирование Датасета №2: 32](#_heading=h.4i7ojhp)

[6.2.1 Метод Python: 32](#_heading=h.2xcytpi)

[6.2.2 Метод Excel: 35](#_heading=h.1ci93xb)

[6.3. Тестирование Датасета №3: 38](#_heading=h.3whwml4)

[6.3.1 Метод Python: 38](#_heading=h.2bn6wsx)

[6.3.2 Метод Excel: 41](#_heading=h.qsh70q)

[6.4. Тестирование Датасета №4: 44](#_heading=h.3as4poj)

[6.4.1 Метод Python: 44](#_heading=h.1pxezwc)

[6.4.2 Метод Excel: 47](#_heading=h.49x2ik5)

[6.5. Тестирование Датасета №5: 50](#_heading=h.2p2csry)

[6.5.1 Метод Python: 50](#_heading=h.147n2zr)

[6.5.2 Метод Excel: 53](#_heading=h.3o7alnk)

[ 55](#_heading=h.23ckvvd)

[Изображение выглядит как стол

Автоматически созданное описание 55](#_heading=h.ihv636)

[7. Заключение. 55](#_heading=h.32hioqz)

# Постановка задачи (физическая модель)

К нам обратилась крупная сеть заправок. Они столкнулись с проблемой хаотичного наплыва клиентов. В связи с этим в пиковые нагрузки система не справляется, образуются очереди и это вызывает негатив клиентов.

Нам необходимо на основе статистических данных, которые предоставил нам заказчик рассчитать при каком количестве людей система начинает не справляться и образуется очередь; что нужно сделать (какие решения внедрить), чтобы при минимальных затратах получить решение проблемы. Критичность очереди составляет не более 3 человек (как на заправку, так и на оплату) и время обслуживания не более 3 минут.

# Математическая модель

**Структура системы массового обслуживания**

Математическая модель для всех систем массового обслуживания имеет

одинаковую структуру. Ее компонентами являются:

* Поступающие новые заявки
* Ожидающие в очереди заявки
* Обслуживающие устройства
* Обслуженные исходящие заявки

**Потоки заявок**

Входной поток заявок характеризуется скоростью поступления заявок в систему (интенсивностью входного потока заявок). Данный показатель показывает сколько заявок приходит в систему за определенный интервал времени.

Если известен средний интервал времени между потоком заявок, то используется формула:

Если нет, то необходимо определить количество заявок, пришедшее за определенный период времени.

где 𝑁 – количество заявок, пришедшее за период времени 𝑡, 𝑡вх – средний интервал времени между поступлением заявок.

Выходной поток заявок характеризуется скоростью обработки заявок системой (интенсивностью выходного потока заявок от одного прибора). Данный показатель показывает сколько заявок обрабатывает один прибор за единицу времени и рассчитывается по формуле. Если известно среднее время, необходимое на обработку 1 заявки одним прибором, то используется формула:

Если нет, то необходимо определить количество заявок, обработанное за период времени некоторым количеством приборов.

где 𝑁 – количество заявок, обработанное за период времени 𝑡 всеми 𝑛 приборами, 𝑡вых – среднее время на обработку одной заявки одним прибором.

**Вероятности состояний системы**

Вероятность того, что система бездействует, определяется по следующей формуле:

Для вероятностей наступления состояний обслуживания используется представленная ниже формула.

Для вероятностей наступления состояний очереди используются следующая формула:

**Показатель загруженности системы**

Величина 𝜌 является показателем загруженности системы и дает представление о том, несколько система справляется с потоком клиентов.

* 𝜌 ≪ 𝑛 (показатель загруженности сильно меньше количества касс)

– система недогружена, что выгодно для клиента, так как нет очередей, но невыгодна для владельца, так как есть лишние кассы и большой простой по времени;

* 𝜌 <𝑛 (показатель загруженности меньше количества касс) –

система сбалансирована для клиента, так как есть приемлемые очереди и допустимый простой касс;

* 𝜌 ≤ 𝑛 – (показатель загруженности приближен к количеству касс)

система сбалансирована для владельца, так как есть большие очереди клиентов, кассы заняты практически полностью;

* 𝜌> 𝑛 (показатель загруженности больше количества касс) –

система перегружена, что выгодно для владельца, так как заявок больше, чем можно обработать, но невыгодно для клиента по причине бесконечно растущей очереди

**Показатели для клиентов**

Вероятность отказа в обслуживании показывает вероятность, что новая заявка не помещается в систему (заняты все приборы и заполнена вся очередь – последняя «правая» вероятность):

Вероятность встать в очередь показывает вероятность того, что новая заявка встречает в системе очередь (заняты все приборы и есть место в очереди - сумма всех вероятностей очередей):

Средняя длина очереди показывает среднее количество заявок, ожидающих в очереди:

Среднее время ожидания в очереди рассчитывается по формуле Литтла:

**Показатели для владельцев**

Абсолютная пропускная способность показывает скорость обслуживания заявок (сколько заявок успевает обрабатываться системой в единицу времени):

Относительная пропускная способность показывает процент обслуженных заявок (какой процент заявок не успевает обрабатываться и получает отказ):

Среднее количество занятых приборов:

Коэффициент простоя показывает процент времени простоя обслуживающих приборов:

# 3.1. Алгоритм в Python

## 3.1.1. Описание входных данных.

Формат входных данных определяется тем, что в программу необходимо вбивать данные вручную.

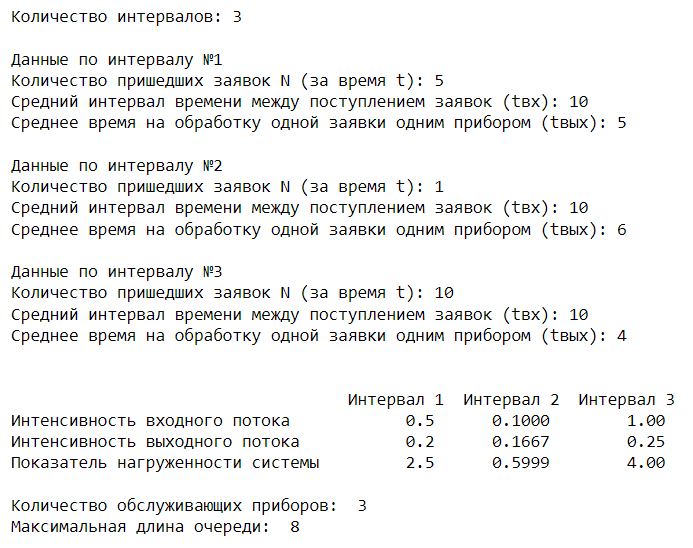


Рисунок 1. Пример входных данных в CSV файле

## 3.1.2. Описание алгоритма решения

После того как данные введены, программе необходимо преобразовать данные для дальнейшего использования.

Шаг 1: расчет показателей интенсивности входного и выходного потока заявок.

Шаг 2: расчет количества обслуживающих приборов и максимальной длину очереди.

Шаг 3: расчет вероятностей нахождения системы в каждой из состояний.

Шаг 4: построение гистограмма по вероятностям.

Шаг 5: расчет расчета средней длины очереди.

Шаг 6: расчет среднего времени ожидания в очереди.

Шаг 7: расчет параметров системы, важных для владельца.

## 3.1.3. Описание выходных данных

В конце программа выдаст вероятности нахождения системы в каждом состоянии временного интервала, гистограммы по этим интервалам, характеристику для клиентов и характеристику для владельцев.

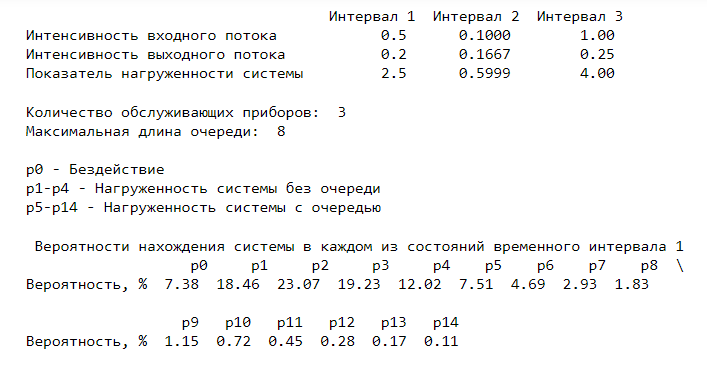


Рисунок 2.Пример выходных данных

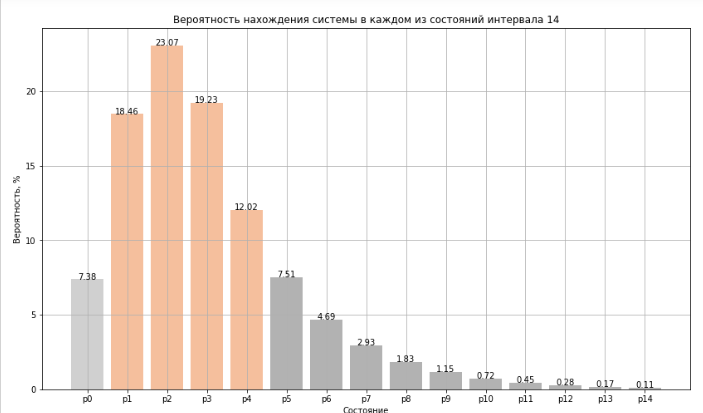


Рисунок 3. Пример выходных данных

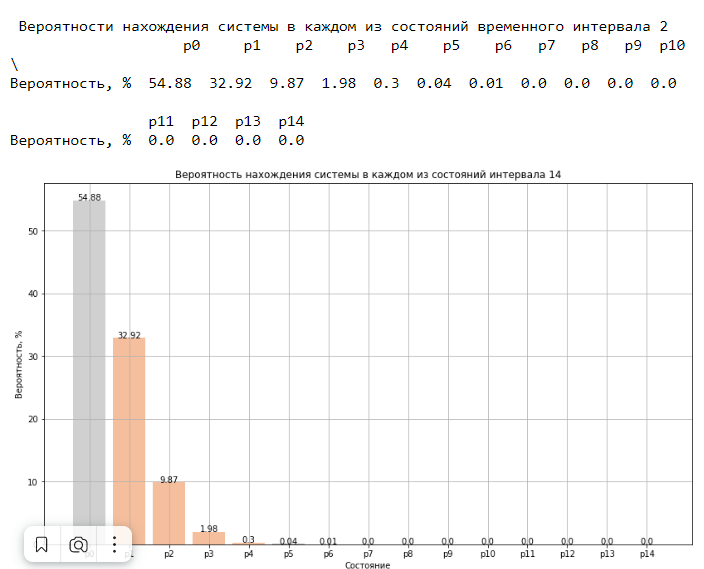


Рисунок 4. Пример выходных данных

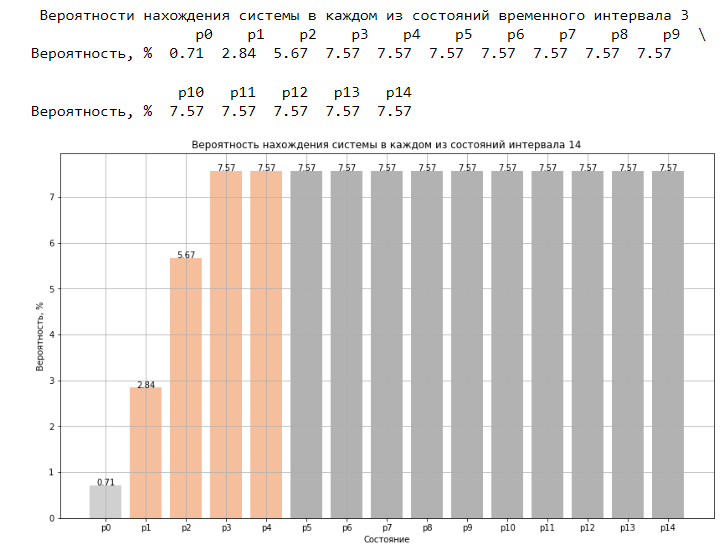


Рисунок 5. Пример выходных данных

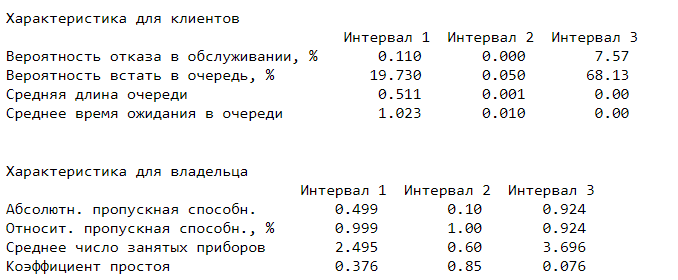


Рисунок 6. Пример выходных данных

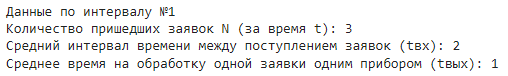
# 4. Вариант использования

Существует единственный вариант использования кода – это ручной ввод данных. Данный вариант использования включает в себя ввод данных ручным способом. Для того, чтобы внести данные необходимо запустить программу и задать количество интервалов, которые вам необходимы.

****

Рисунок 7. Установка интервалов

После этого появляется окно, в котором вам необходимо ввести, сколько пришло заявок, средний интервал между ними и сколько времени нужно, чтобы обработать одну заявку. Пример:



*Рисунок 8. Пример вводных данных*

Так нужно прописать для всех интервалов, которые вы указали пунктом выше.

В конце нужно указать количество обслуживающих приборов и максимально допустимую длину очереди.



Рисунок 9. Пример ввода данных

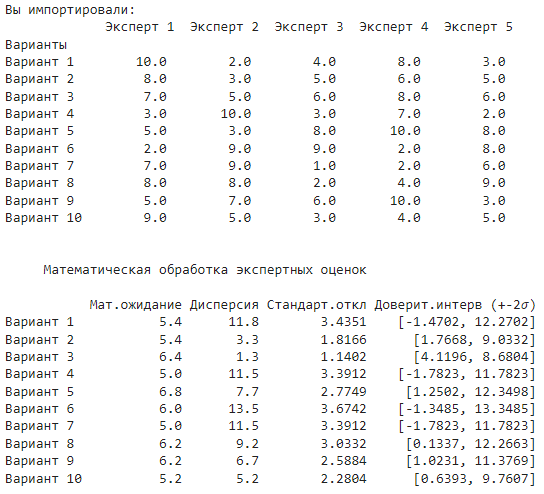


Рисунок 10. Решение

При выборе 3 типа задачи необходимо ввести количество экспертов и после этого для каждого подгрузить соответствующие файлы.

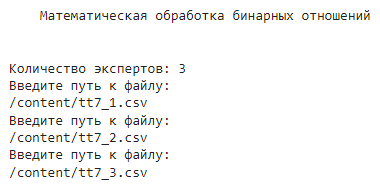


Рисунок 11. 3 тип задачи

# 5. Архитектура решения

Для решения задачи использовались методы (функции), которые можно разделить на 3 принципиальных кода.

## 5.1 Функции считывания информации

**Функция input\_data:**

**Входные данные:**

* Нет входных данных.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* value - принимает все значения, введенные пользователем.
* i – название вводимого параметра.

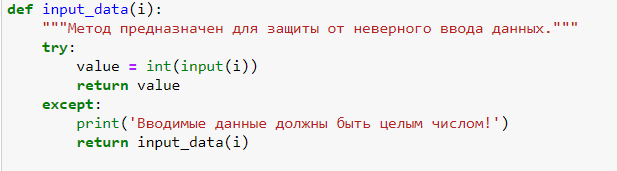
****

Рисунок 12. Фрагмент кода

## 5.2 Функции обработки информации

После того, как вы введете все необходимые данные, программа их получит и начнет первичную обработку.

Функция task\_1 предназначена для расчета всех показателей и включает в себя 2 функции: intensivn\_input, в которой рассчитывается интенсивность входного и выходного потока заявок и показатель загруженности системы; queue, в которой рассчитывается количество обслуживающих приборов и максимальную длину очереди.

Функция task\_2 предназначена для расчета показателей и включает в себя 3 функции: p, в которой рассчитывается вероятность нахождения системы в каждом из состояний; color, в которой устанавливается цвета столбцов в гистограммах; graph, в которой осуществляется построение гистограммы по вероятностям.

Функция task\_3 предназначена для расчета параметров важных для клиента и включает в себя 2 функции: average\_len\_queue, в которой рассчитывается средняя длина очереди; average\_time\_waitin, в которой рассчитывается среднее время ожидания в очереди.

Функция task\_4 предназначена для расчета параметров системы важных для владельца, таких как абсолютной пропускной способности, относительной пропускной способностью, среднее число занятых приборов, коэффициент простоя.

**Функция task\_1:**

Она включает в себя две функции:

Функция intensivn\_input:

**Входные данные:**

* data – словарь с данными, которые ввел пользователь.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* list\_intensivn\_input – интенсивность входного потока заявок.
* list\_intensivn\_output – интенсивность выходного заявок.
* system\_load – показатель загруженности системы.
* data – словарь с данными, которые ввел пользователь.

Функция queue:

**Входные данные:**

* system\_load – показатель загруженности системы.

**Выходные данные:**

* load – словарь с данными, которые ввел пользователь.
* system\_load – показатель загруженности системы.
* sum\_service\_device – количество обслуживающих приборов.
* sum\_queue – максимальная длина очереди.

**Переменные, затрагиваемые в ходе работы:**

* sum\_service\_device – количество обслуживающих приборов.
* sum\_queue – максимальная длина очереди.
* system\_load – показатель загруженности системы.
* name\_rows – название столбцов в Dataframe.
* name\_rows – название строк в Dataframe.

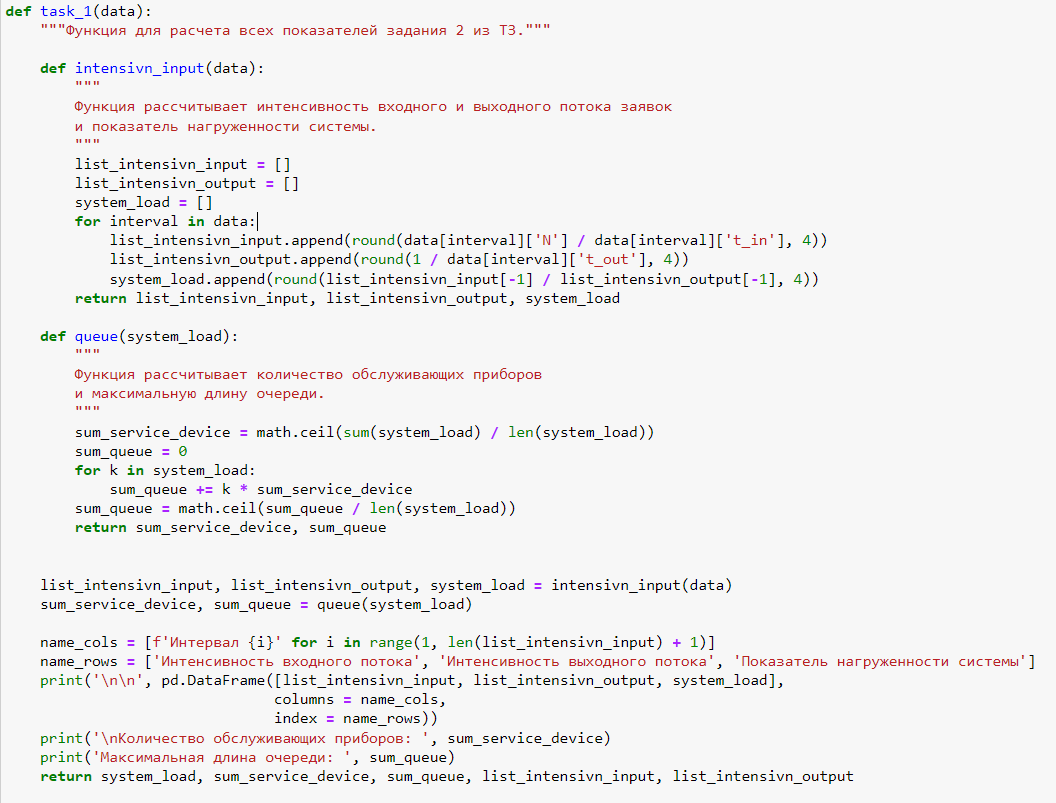


Рисунок 13. Фрагмент кода

**Функция task\_2:**

Она включает в себя три функции:

Функция p:

**Входные данные:**

* load – словарь с данными, которые ввел пользователь.
* device – количество используемых приборов.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* load – словарь с данными, которые ввел пользователь.
* list\_load\_1 –список состояния без очереди.
* list\_load\_2 – список состояния с очередью.
* list\_load – вспомогательные вычисления.
* value – значения состояний.
* list\_p – список вероятностей состояний.
* df – Dataframe с вероятностями состояний.

Функция color:

**Входные данные:**

* device – количество используемых приборов.
* sum\_queue – максимальная длина очереди.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* device – количество используемых приборов.
* sum\_queue – максимальная длина очереди.
* color – список цветов гистограммы.

Функция graph:

**Входные данные:**

* device – количество используемых приборов.
* sum\_queue – максимальная длина очереди.
* list\_p – список вероятностей состояний.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* device – количество используемых приборов.
* sum\_queue – максимальная длина очереди.
* list\_p – список вероятностей состояний.
* color – список цветов гистограммы.
* sum\_service\_device – количество обслуживающих приборов.

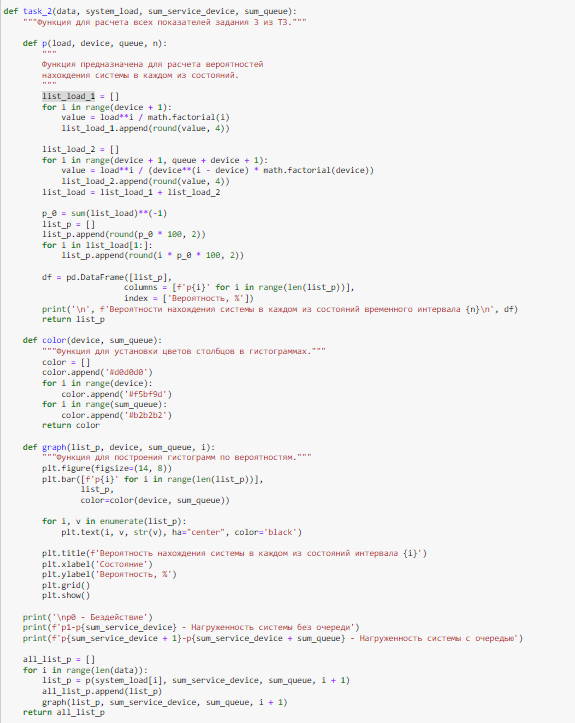


Рисунок 14. Фрагмент кода

**Функция task\_3:**

Она включает в себя две функции:

Функция average\_len\_queue:

**Входные данные:**

* list\_p – список вероятностей состояний.
* system\_load – показатель загруженности системы.
* sum\_queue – максимальная длина очереди.
* device – количество используемых приборов.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* list\_p – список вероятностей состояний.
* system\_load – показатель загруженности системы.
* sum\_queue – максимальная длина очереди.
* device – количество используемых приборов.
* value1 – средняя длина очереди.

Функция average\_time\_waiting:

**Входные данные:**

* length– средняя длина очереди.
* list\_p – список вероятностей состояний.
* intensivn – данные, введенные пользователем.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* length– средняя длина очереди.
* list\_p – список вероятностей состояний.
* intensivn – данные, введенные пользователем.
* name\_rows – название столбцов в Dataframe.
* name\_rows – название строк в Dataframe.

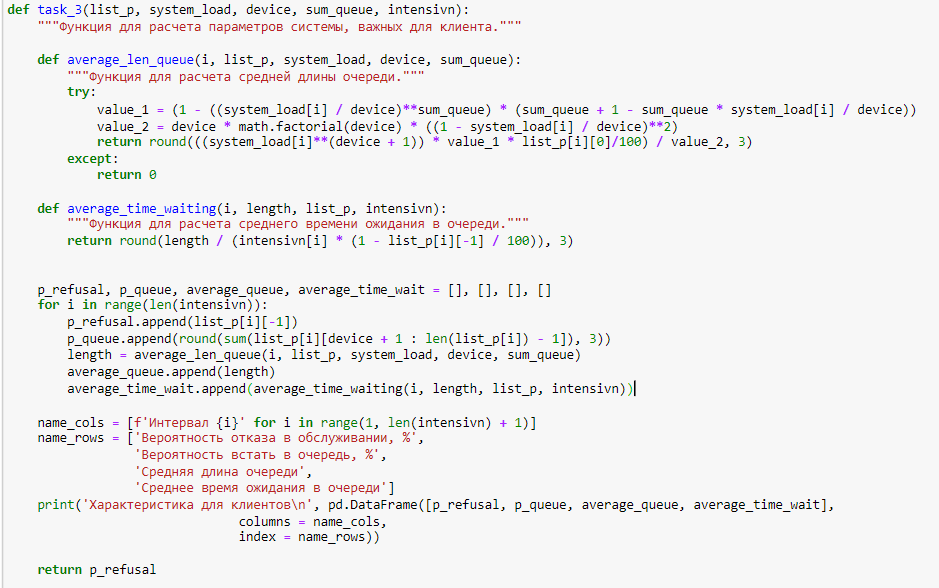


Рисунок 15. Фрагмент кода

**Функция task\_4:**

**Входные данные:**

* p\_refusal – вероятность отказа в обслуживания.
* device – количество используемых приборов.

**Выходные данные:**

* Нет выходных данных.

**Переменные, затрагиваемые в ходе работы:**

* p\_refusal – вероятность отказа в обслуживания.
* device – количество используемых приборов.
* name\_cols – название столбцов в Dataframe.
* name\_rows – название строк в Dataframe.
* absol\_throughput – абсолютная пропускная способность.
* relat\_throughput – относительная пропускная способность.
* busy\_device – среднее число занятых приборов.
* K – коэффициент простоя.



Рисунок 16. Фрагмент кода

## 5.3 Функции вывода информации

Метод вывода информации

Что делает: осуществляет вывод необходимой информации

В данном методе осуществляется непосредственно вызов функций с помощью метода print ().

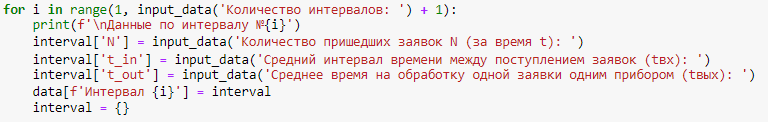
****

Рисунок 17. Часть кода, отвечающая за вывод данных

# 6. Тестирование

Проведём тестирование нашей программы и сравним полученные показатели, чтобы сделать вывод о предпочтительном варианте использования нашей программы или Excel под условия заказчика.

## 6.1. Тестирование Датасета №1:

### 6.1.1 Метод Python:

Вводим необходимое количество интервалов, определяем количество пришедших заявок, средний интервал времени между поступлением заявок и среднее время на обработку одной заявки одним прибором, повторяем процедуру по количеству введённых интервалов.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18. Входные данные в питон

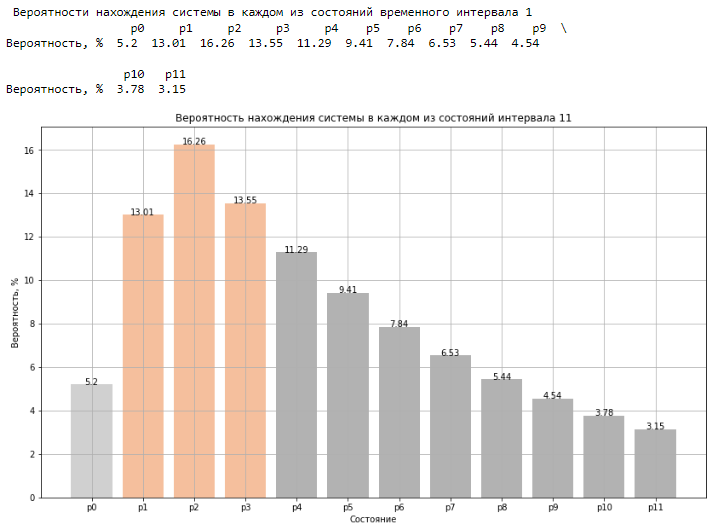


Рисунок 19. Выходные данные в питоне

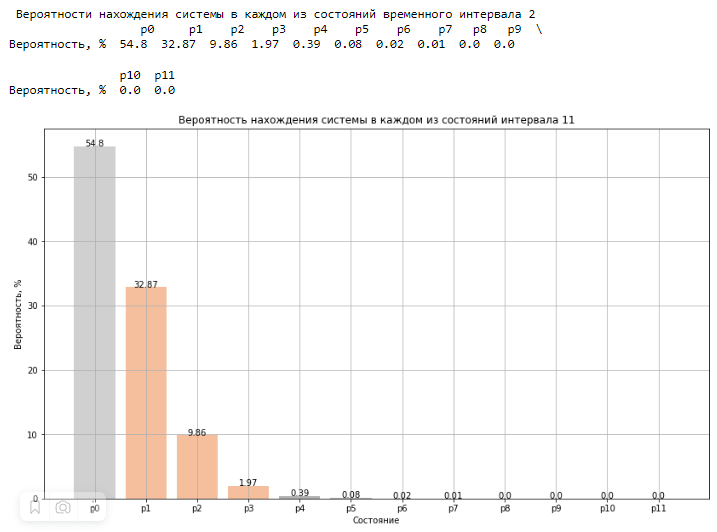


Рисунок 20. Выходные данные в питоне

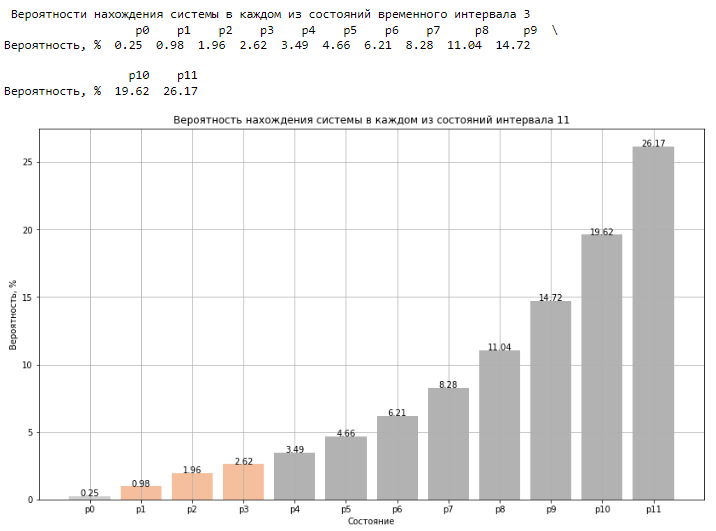


Рисунок 21. Выходные данные в питоне

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 22. Выходные данные в питоне

### 6.1.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 23*. Тест в экселе*

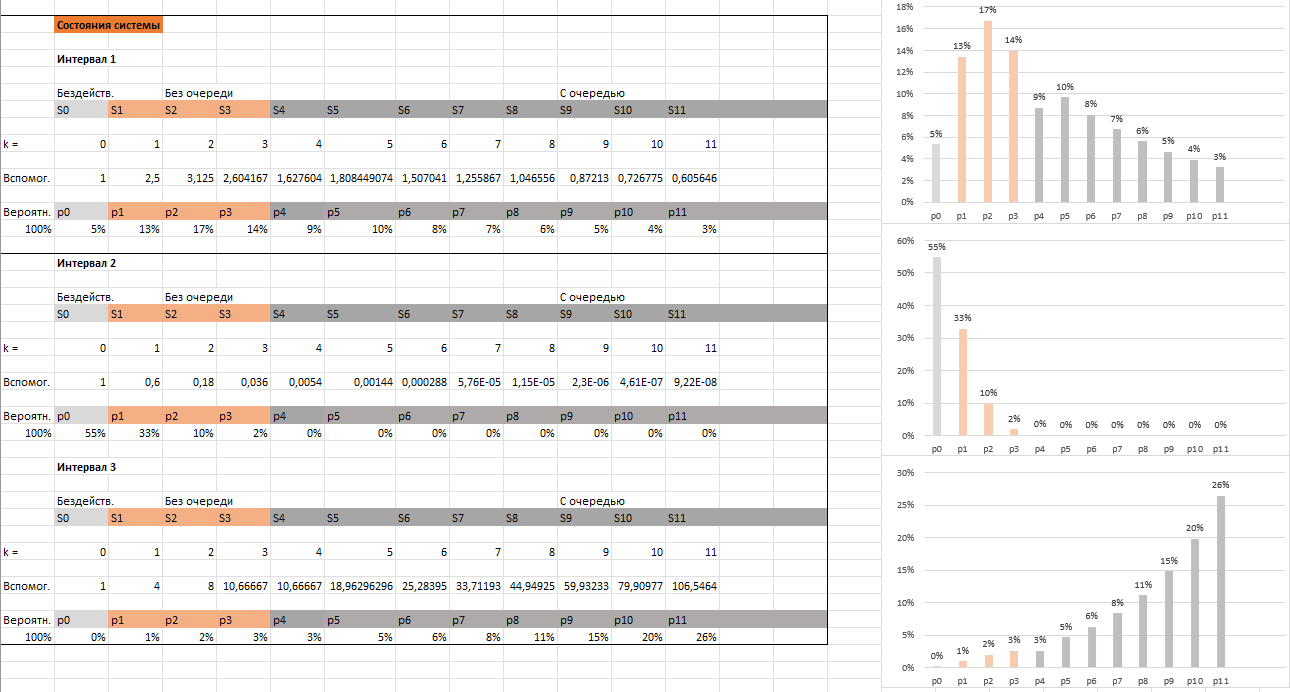


Рисунок 24. Тест в экселе

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 25. Тест в экселе

## 6.2. Тестирование Датасета №2:

### 6.2.1 Метод Python:

Вводим необходимое количество интервалов, определяем количество пришедших заявок, средний интервал времени между поступлением заявок и среднее время на обработку одной заявки одним прибором, повторяем процедуру по количеству введённых интервалов.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 26. Входные данные в питон

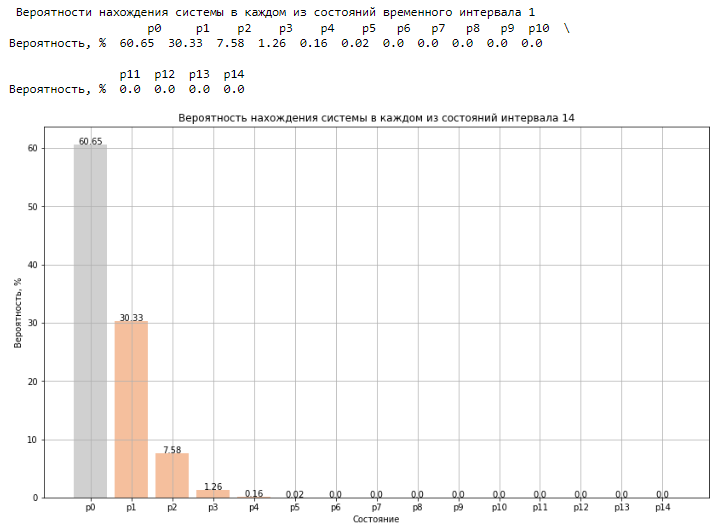


Рисунок 27. Выходные данные в питоне

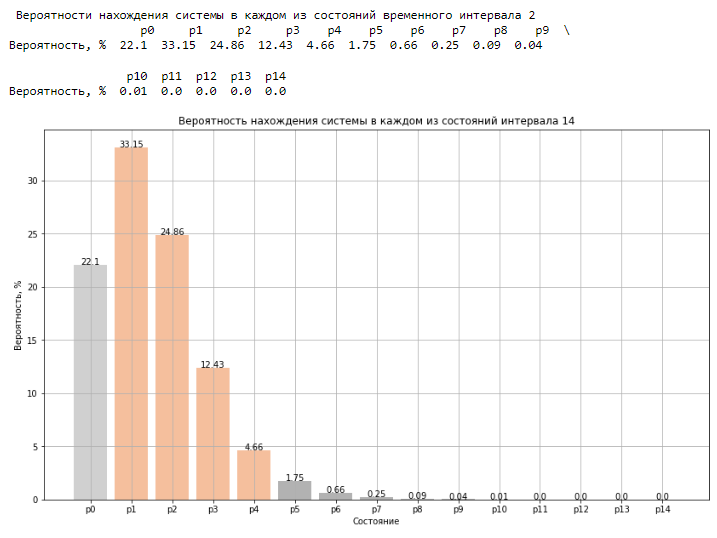


Рисунок 28. Выходные данные в питоне

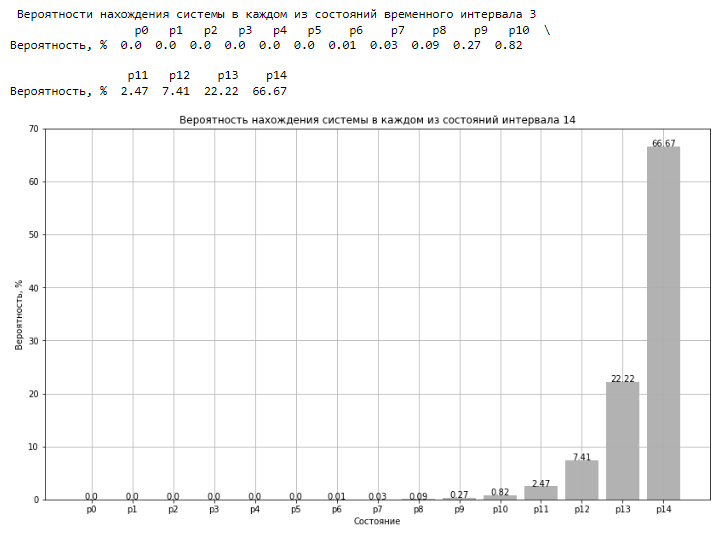


Рисунок 29. Выходные данные в питоне

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 30. Выходные данные в питоне

### 6.2.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 31. Тест в экселе

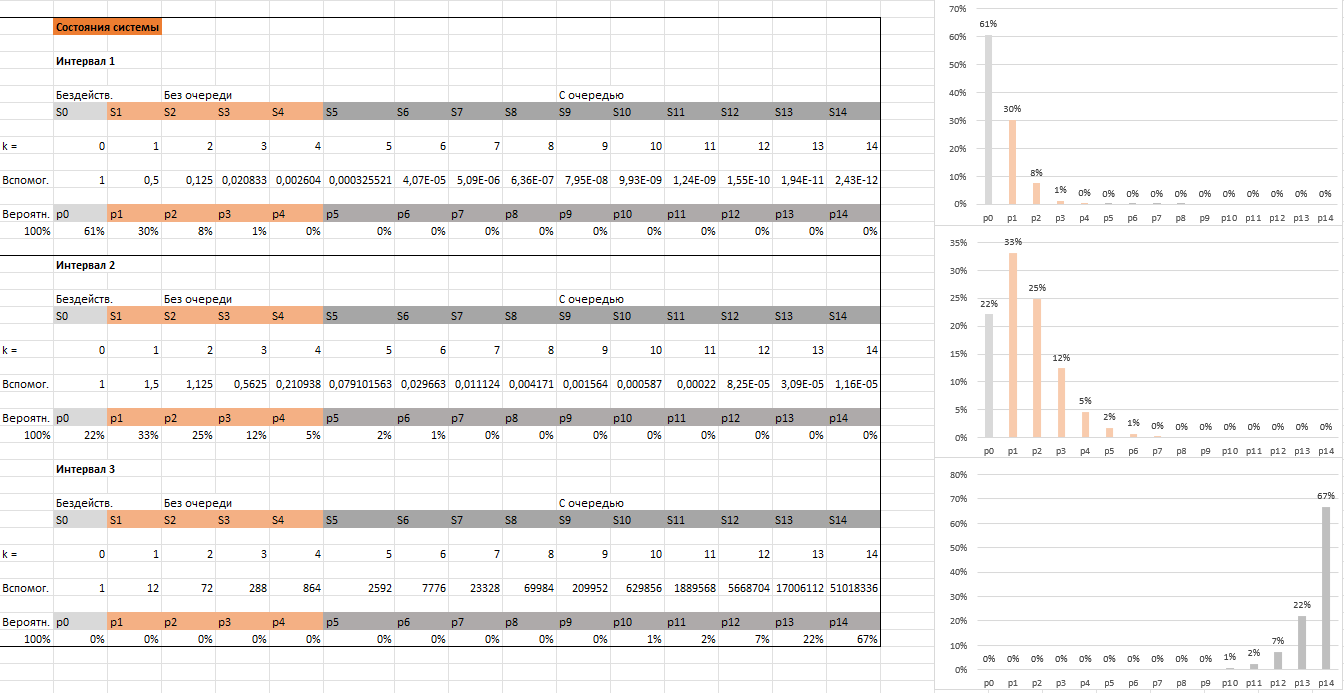


Рисунок 32. Тест в экселе

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 33. Тест в экселе

## 6.3. Тестирование Датасета №3:

### 6.3.1 Метод Python:

Вводим необходимое количество интервалов, определяем количество пришедших заявок, средний интервал времени между поступлением заявок и среднее время на обработку одной заявки одним прибором, повторяем процедуру по количеству введённых интервалов.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 34. Входные данные в питон

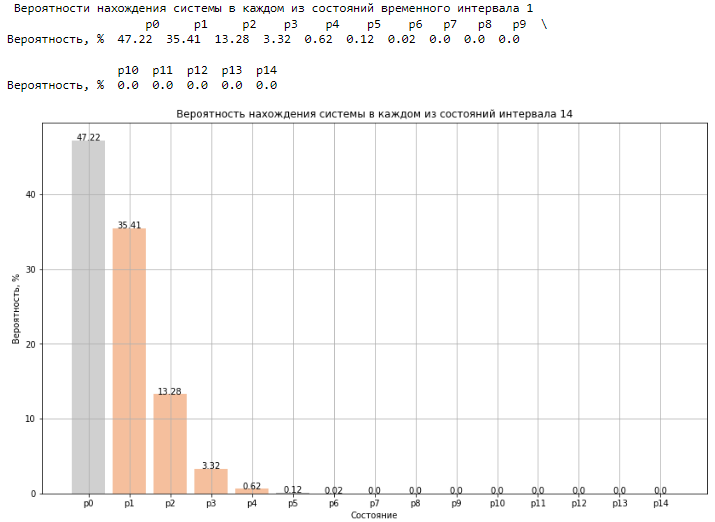


Рисунок 35. Выходные данные в питоне

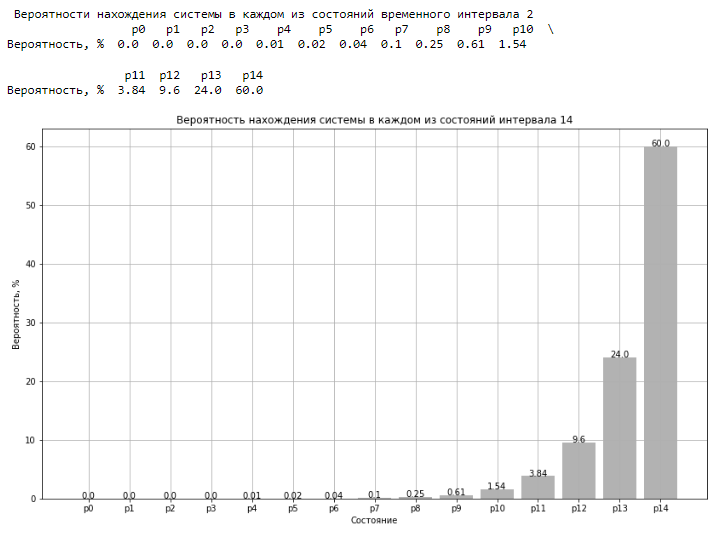


Рисунок 36. Выходные данные в питоне

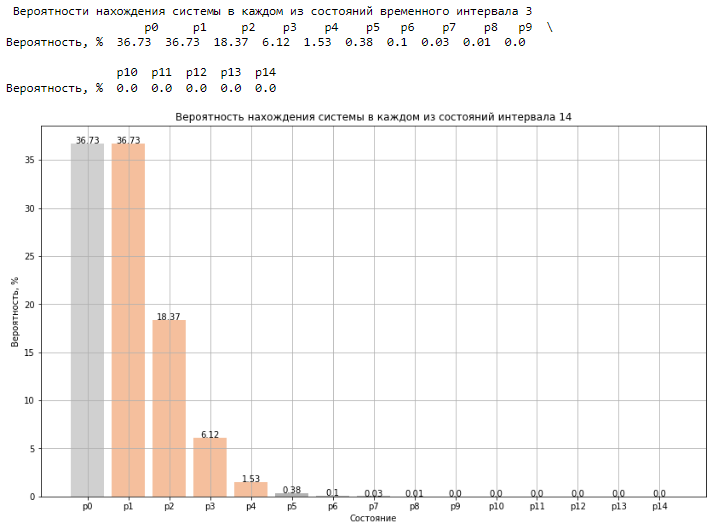


Рисунок 37. Выходные данные в питоне

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 38. Выходные данные в питоне

### 6.3.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 39. Тест в экселе

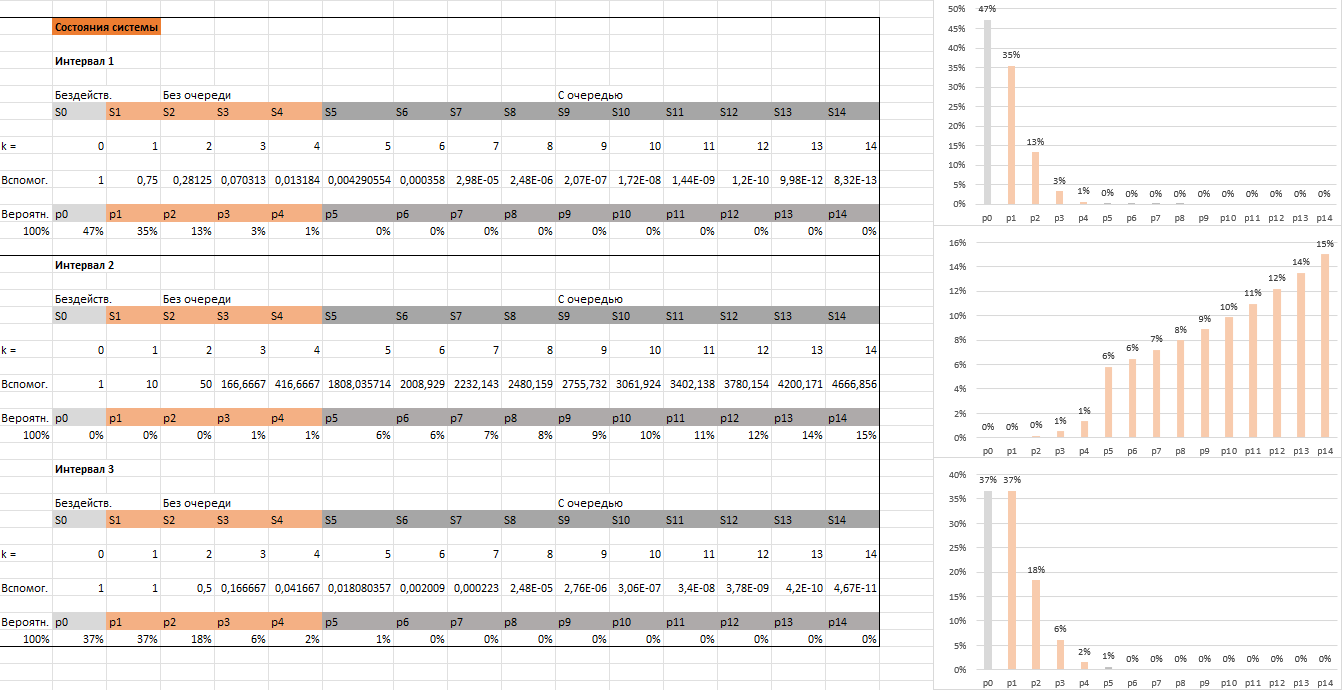


Рисунок 40. Тест в экселе

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 41. Тест в экселе

## 6.4. Тестирование Датасета №4:

### 6.4.1 Метод Python:

Вводим необходимое количество интервалов, определяем количество пришедших заявок, средний интервал времени между поступлением заявок и среднее время на обработку одной заявки одним прибором, повторяем процедуру по количеству введённых интервалов.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 42. Входные данные в питон

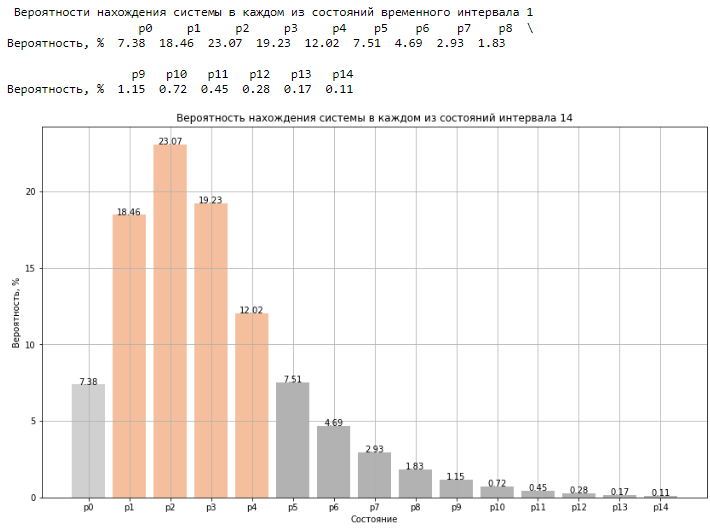


Рисунок 43. Выходные данные в питоне

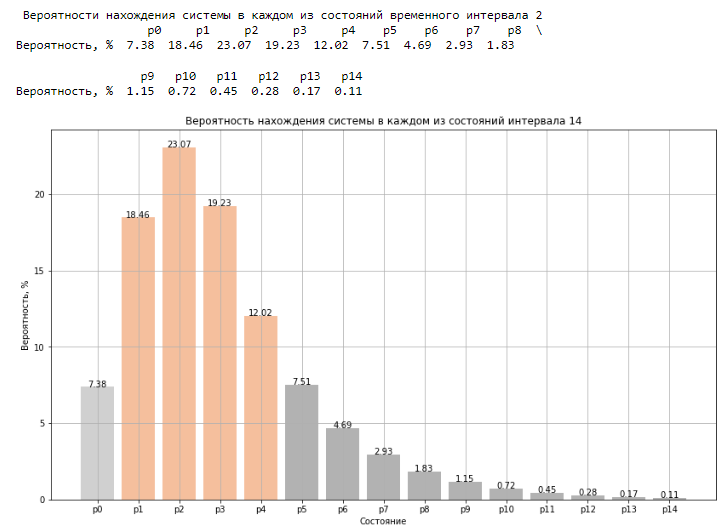


Рисунок 44. Выходные данные в питоне

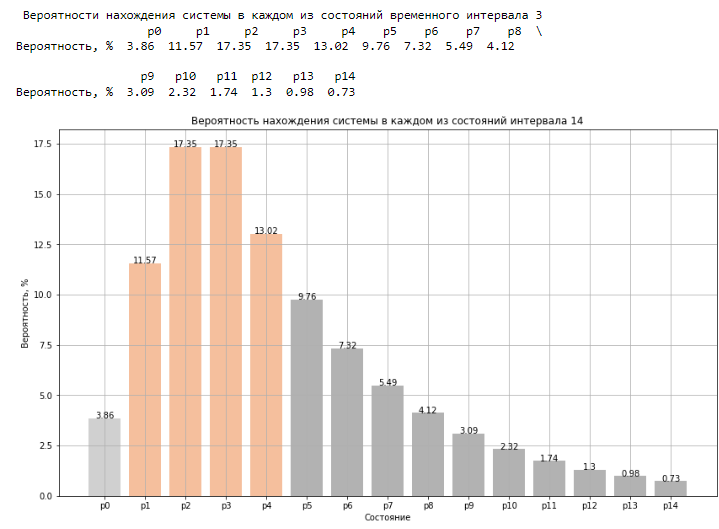


Рисунок 45. Выходные данные в питоне

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 46. Выходные данные в питоне

### 6.4.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:



Рисунок 47. Тест в экселе

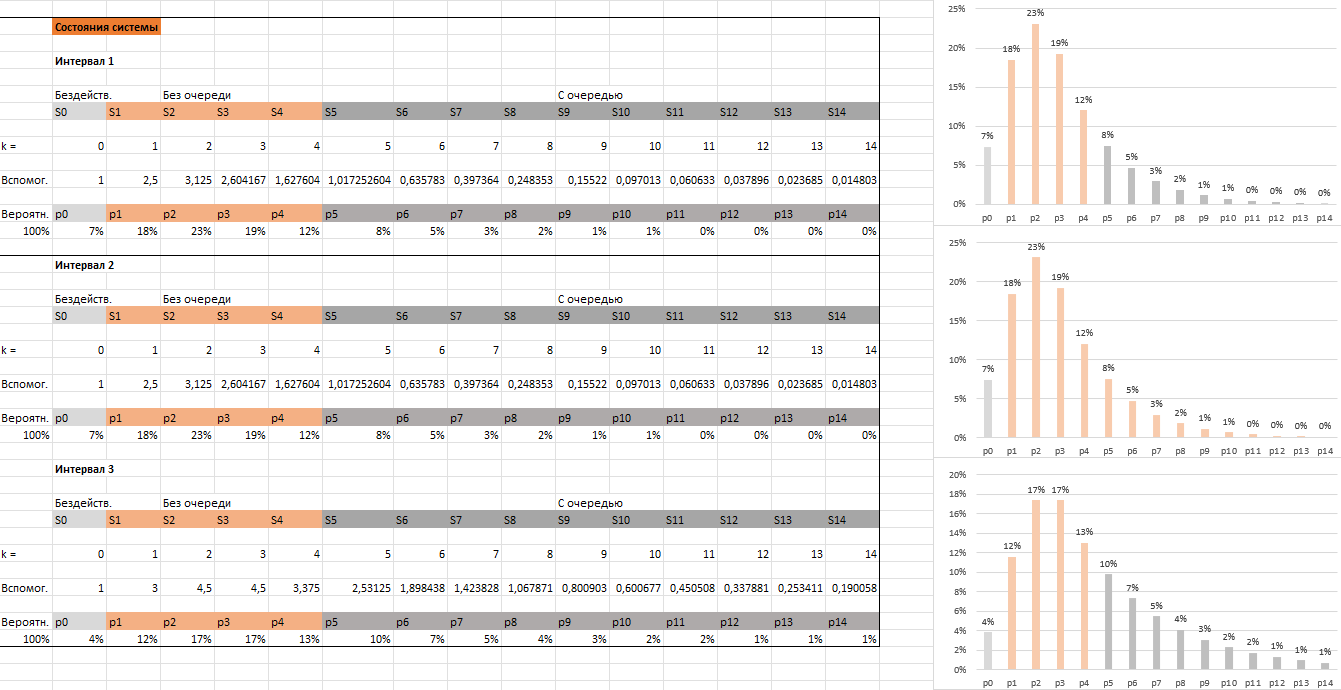


Рисунок 48. Тест в экселе

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 49. Тест в экселе

## 6.5. Тестирование Датасета №5:

### 6.5.1 Метод Python:

Вводим необходимое количество интервалов, определяем количество пришедших заявок, средний интервал времени между поступлением заявок и среднее время на обработку одной заявки одним прибором, повторяем процедуру по количеству введённых интервалов.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 50. Входные данные в питон

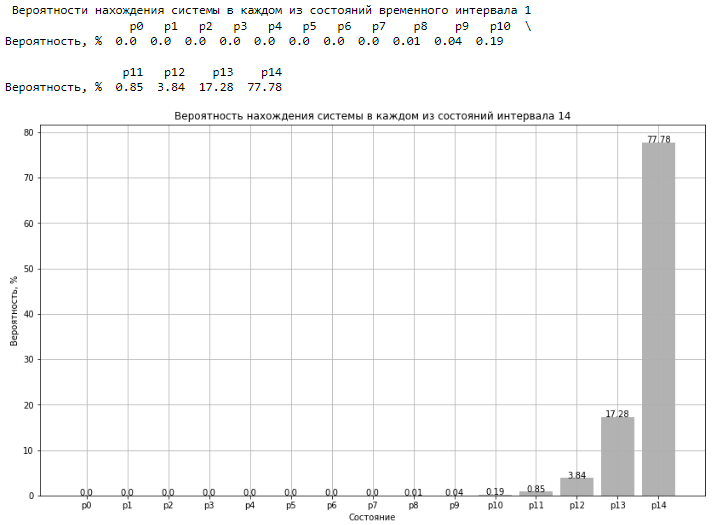


Рисунок 51. Выходные данные в питоне

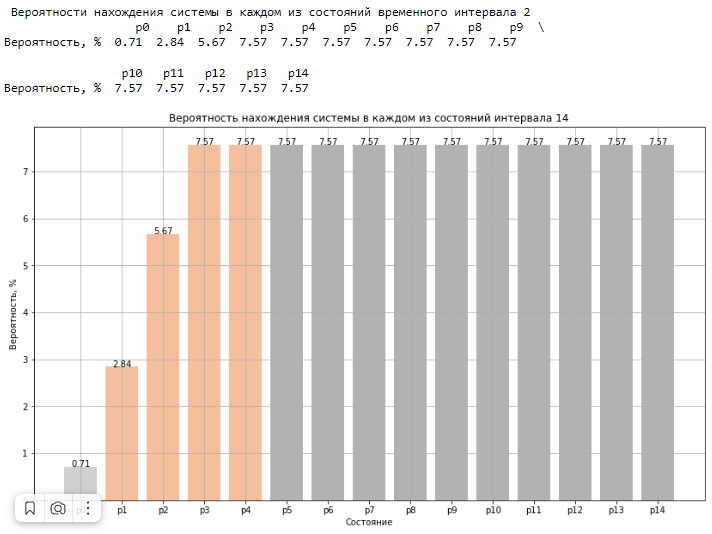


Рисунок 52. Выходные данные в питоне

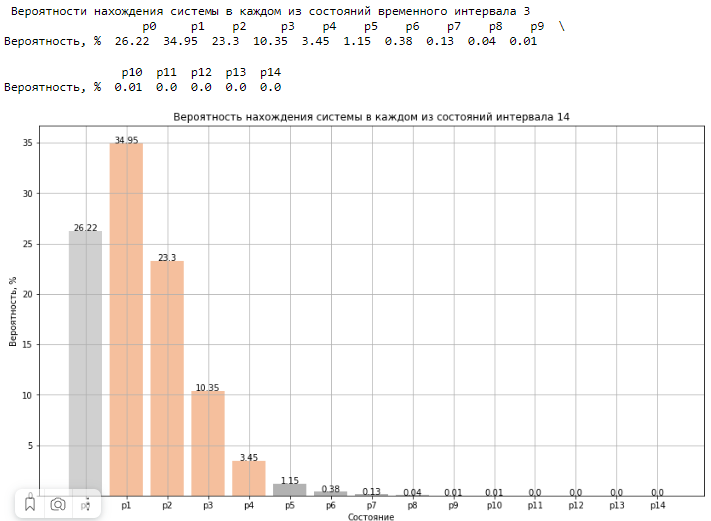


Рисунок 53. Выходные данные в питоне

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 54. Выходные данные в питоне

### 6.5.2 Метод Excel:

Вводим данные датасета в необходимые поля и получаем результат:

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 55. Тест в экселе

# 

Рисунок 56. Тест в экселе

# Изображение выглядит как стол Автоматически созданное описание

Рисунок 57. Тест в экселе

# 7. Заключение.

Представленный нами код решает поставленную задачу.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 58. Входные данные в питон

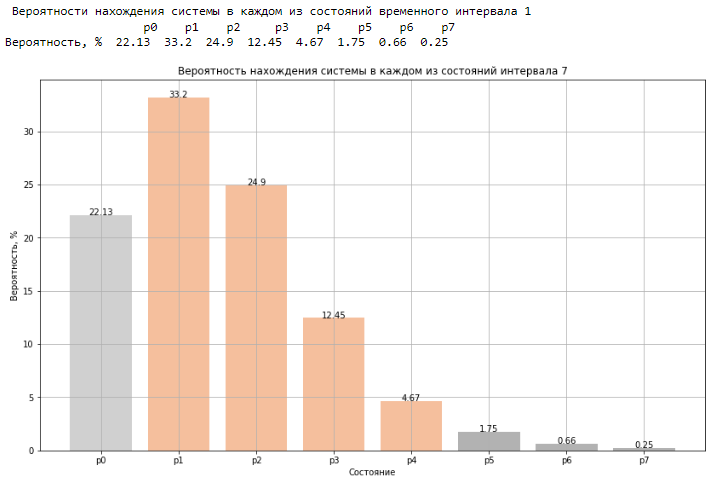


Рисунок 59. Выходные данные в питоне

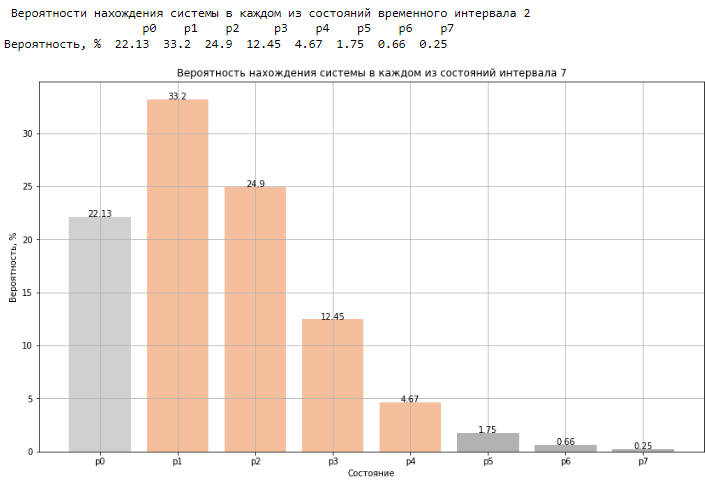


Рисунок 60. Выходные данные в питоне

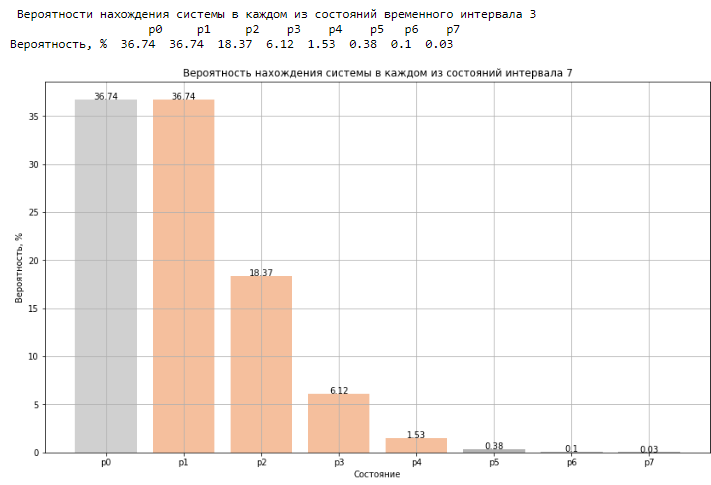


Рисунок 61. Выходные данные в питоне

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 62. Выходные данные в питоне

Наиболее загруженными являются утренние и дневные интервалы (1 и 2), для решения этой проблемы необходимо внедрить новую кассу обслуживания и работника, который будет работать на пол ставки в первую половину дня.

На основании тестирования данного алгоритма можно сделать вывод о том, что Python выводит самое оптимальное решение наиболее быстро и наглядно. Теперь сравним два алгоритма по критериям: эффективности, скорости использования алгоритма, простоты использования, надёжности в разрезе человеческого фактора и точности предоставляемого решения.

*Таблица 1. Сравнение решения в Excel и Pythоn*

| **Критерий** | **Python** | **Excel** |
| --- | --- | --- |
| Эффективность | Высокая | Высокая |
| Скорость использования алгоритма | Средняя | Средняя |
| Простота использования | Высокая | Средняя |
| Надёжность | Высокая | Средняя |
| Точность | Высокая | Высокая |

Мы считаем, что представленный рукописный код на языке Python лучше, потому что удобнее, быстрее и проще. Улучшением кода может послужить добавление времени выполнения запроса, более детальной выводимой информации, импортного ввода данных и генерацию случайных данных.