

Министерство образования и науки РФ  
Санкт-Петербургский политехнический университет Петра Великого  
Институт компьютерных наук и технологий  
Высшая школа программной инженерии

## **КУРСОВАЯ РАБОТА**

по дисциплине "Архитектура программных систем"

### **Моделирование системы массового обслуживания**

Выполнил студент гр. 3530904/00104

Смирнов Е. А.

Руководитель

Н. Г. Смирнов

## Оглавление

Введение.....	3
Исходные данные .....	4
Описание СМО.....	4
Формализованная схема .....	5
Временная диаграмма .....	6
Вывод законов распределения .....	7
1. Генерация случайной величины.....	7
2. Обслуживание заявки на приборе .....	7
Пример системы, удовлетворяющей формализованному описанию.....	8
Документация ПО .....	10
Модульная структура и описание модулей .....	12
Результаты работы имитационной модели.....	15
Определение количества реализаций .....	15
Рекомендации по выбору конфигурации системы .....	16
Вывод.....	18
Приложение .....	19

## Введение

Целью практической курсовой является создание модели ВС или ее компонентов на некотором уровне детализации, описывающей и имитирующей ее структуру и функциональность.

Каждый реальный объект ВС обладает огромной сложностью, определяемой множеством состояний, множеством внутренних и внешних связей, множеством анализируемых характеристик. Модель дает приближенное описание объекта с целью получения требуемых результатов с определенной точностью и достоверностью. Степень приближения модели к описываемому объекту может быть различной и зависит от требований задачи.

Существуют различные типы моделей ВС: аналитические, аналоговые, физические и имитационные. В данной работе будет использоваться имитационная модель ВС. Одним из подходов к построению имитационной модели является построение ее в виде системы массового обслуживания (СМО).

## Исходные данные

Задание:

ИБ	ИЗ2	ПЗ1	Д1ОЗ2	Д1ОО4	Д2П2	Д2Б1	ОР1	ОД2
----	-----	-----	-------	-------	------	------	-----	-----

### Описание СМО

Источник

ИБ	ИЗ2
бесконечный источник	равномерный закон распределения источника

Приборы и буфер:

ПЗ1	Д1ОЗ2
экспоненциальный закон распределения времени обслуживания	заполнение буферной памяти: в порядке поступления

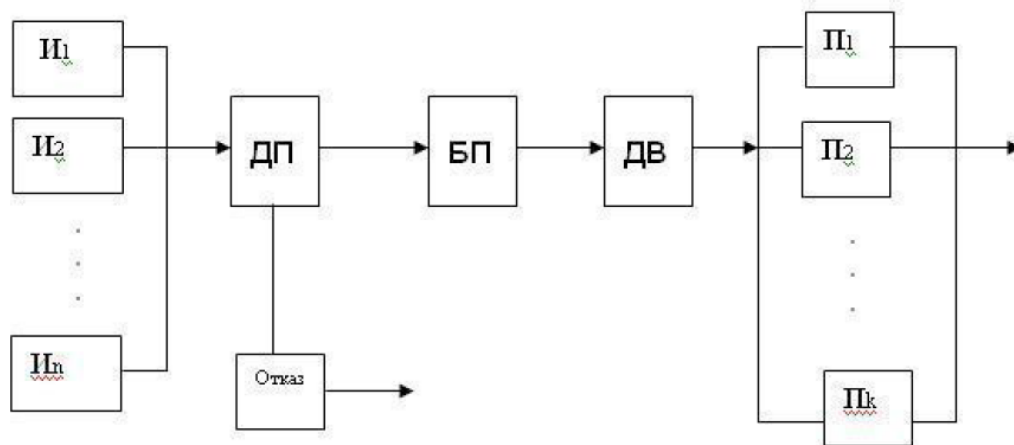
Дисциплины выбора прибора/ выбора места в буфере/ отказа:

Д1ОО4	Д2П2	Д2Б1
дисциплина отказа: последняя добавленная заявка в буфер	дисциплина а выбора прибора: по кольцу	дисциплина выбора заявок на обслуживание: LIFO

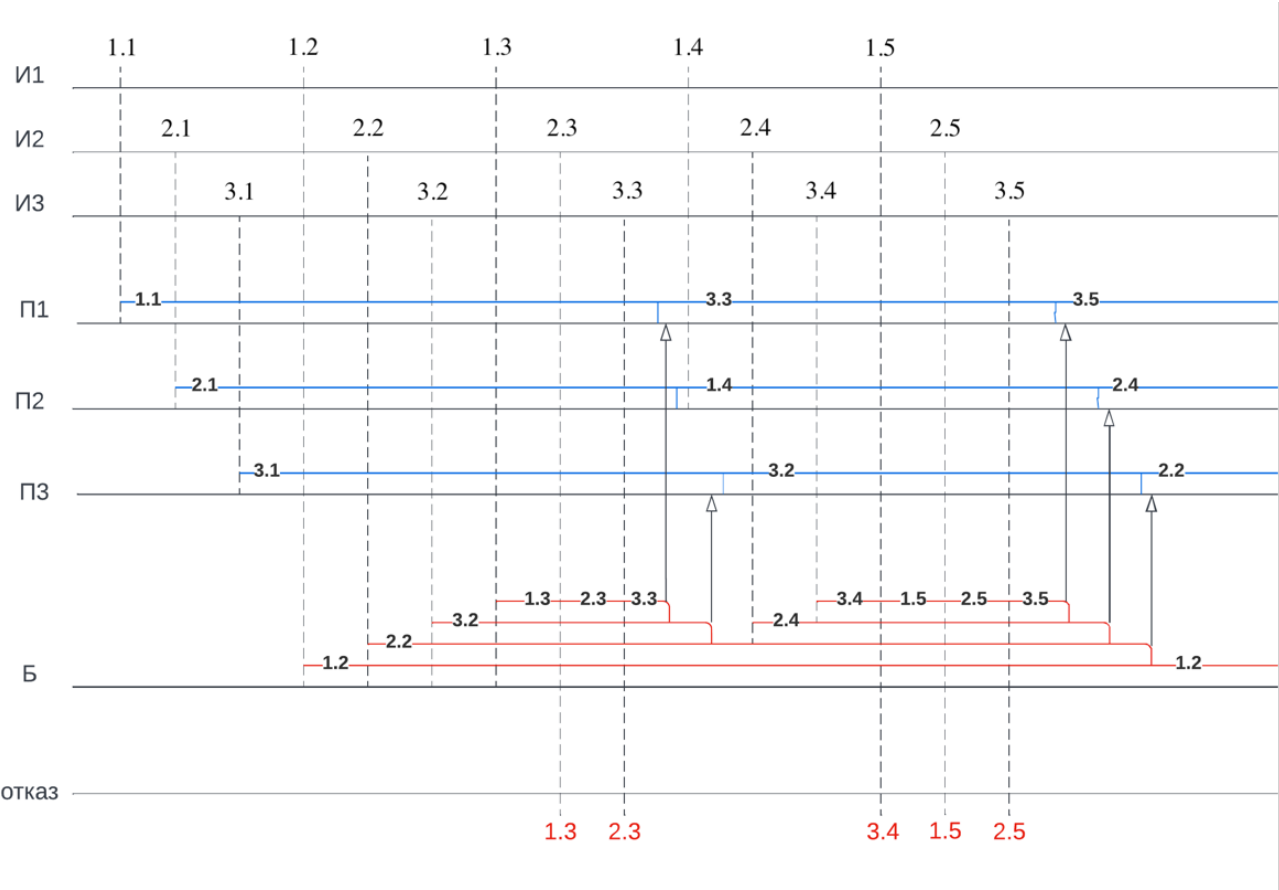
Виды отображения результатов:

ОР1	ОД2
сводная таблица результатов	формализованная схема модели, текущее состояние

## Формализованная схема



# Временная диаграмма



## Вывод законов распределения

Генерация заявки и обслуживание заявки определяются математическими функциями *равномерной случайной величины*.

### 1. Генерация случайной величины

Равномерный закон распределения:

$$F(x) = \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a \leq x \leq b \\ 1, & x \geq b \end{cases}$$

Выразим  $x$ :

$$x = F(x)(b - a) + a$$

Параметры  $a$  и  $b$  – ограничивают случайную величину, чтобы она достоверно приняла одно из значений отрезка.

### 2. Обслуживание заявки на приборе

Экспоненциальный закон распределения:

$$F(x) = \begin{cases} 1 - e^{-\lambda x}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

Выразим  $x$ :

$$\begin{aligned} 1 - F(x) &= e^{-\lambda x} \\ \ln(1 - F(x)) &= -\lambda x \\ x &= -\frac{1}{\lambda} \ln(1 - F(x)) \end{aligned}$$

Параметр  $\lambda > 0$  говорит об интенсивности величины во времени.

## Пример системы, удовлетворяющей формализованному описанию

Техническая система	<b>Винодельная компания:</b> производство, хранение и реализация вина.
Источники	<p>Источниками являются винодельные <b>бочки</b>, где созревает вино. Скорость производства партии вина <i>составляет</i> от 3 <i>месяца</i> до 6 <i>месяцев</i>, смотря какое вино будет производиться. Затем партия изготовленного вина отправляется на склад, откуда они попадают в магазины.</p> <p>На данный момент компания обладает 10 <i>бочками</i>, но она способна купить еще 20 бочек.</p>
Приборы	Прибором является <b>магазин</b> . Началом работы является доставка одной партии вина в магазин. Результатом работы является реализация всех бутылок вина из партии. Минимальный срок, за которой магазин продать партию вина - 1 месяц (перед новым годом), максимальный срок реализации - 9 месяцев.
Буфер	Буфером является <b>склад</b> . Склад вмещает 100 партий вина на полки. Компания может купить еще место для 100 партий.
Дисциплина постановки в буфер	В <b>порядке поступления</b> , ни одна из полок складе не является приоритетной.
Дисциплина выбора из буфера	<b>LIFO</b> , так как новые вина дешевле стоят и их легче продать, то их первыми нужно реализовать, а старые вина будут только дорожать. Чем дольше вино пролежит на складе(в буфере), тем большую цену за него можно получить.
Дисциплина отказа	<b>Последняя пришедшая</b> партия в буфере идёт в отказ, так как ее стоимость и ценность меньше тех, чем тех, которые хранятся на складе уже некоторое время.
Дисциплина постановки на обслуживание	Магазин получает новую партию, как только распродает всю партию вина; поставка в магазин <b>по кольцу</b>



**Ограничения и требуемые характеристики:**

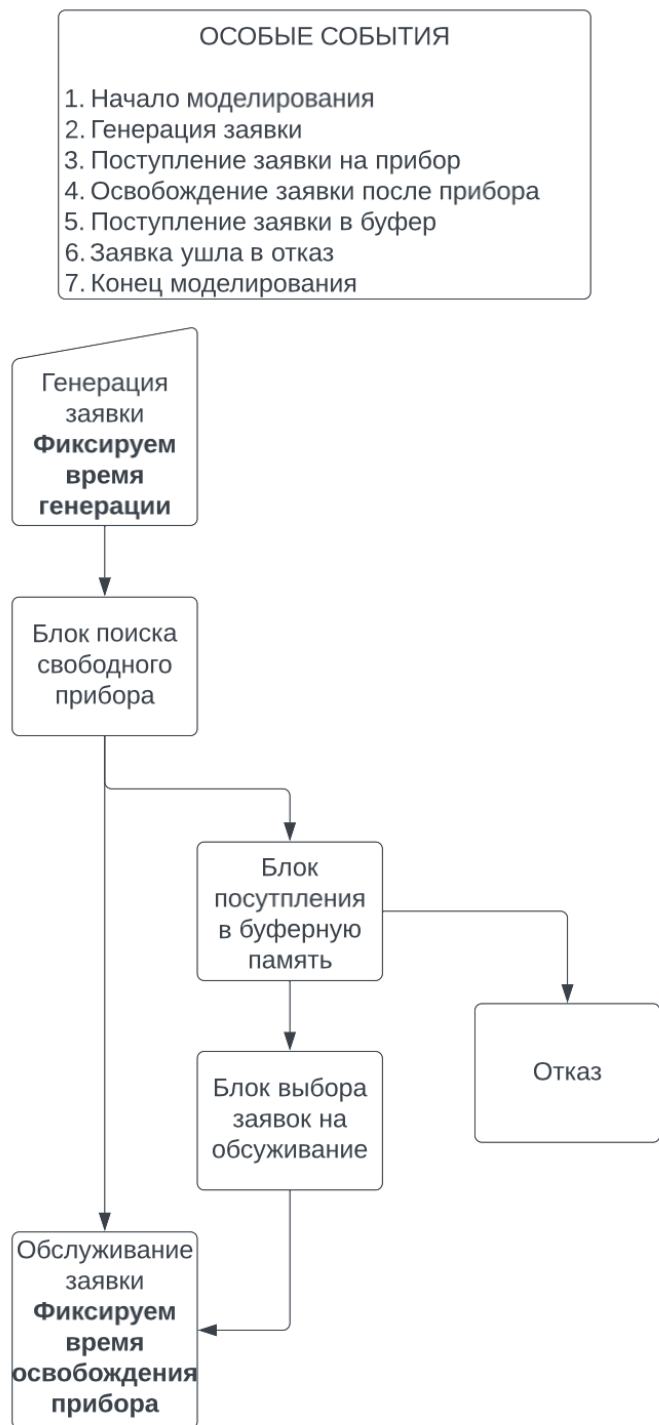
Вероятность отказа должна составлять не более 5%

Загрузка магазинов(приборов) более 90%

<b>Кол-во источников</b>	10-20 бочек
<b>Размер заявки</b>	1 партия вина
<b>Размер буфера</b>	100 мест - 200 мест
<b>Количество магазинов</b>	1-5 магазина
<b>Реализация вина</b>	$\lambda = 0.9$
<b>Скорость производства на заводе</b>	$a = 3$ и $b = 6$

# Документация ПО

## Блок-схема СМО



Блок поиска свободного прибора происходит по указателю. Указатель устанавливается на тот прибор, который стоит следующим по порядку после последнего занятого. Если система проходит все приборы, тогда заявка отправляется в буфер.

Блок поступления в буфер и блок выбора заявки являются алгоритмом LIFO. Если в буфере заканчивается место, то последняя заявка уходит в отказ, а поступившая занимает ее место.

Блоки поиска свободного прибора, поступления в буферную память, выбора заявки на обслуживание – содержат в себе свой **алгоритм**, описанный в блок схемах – ниже.

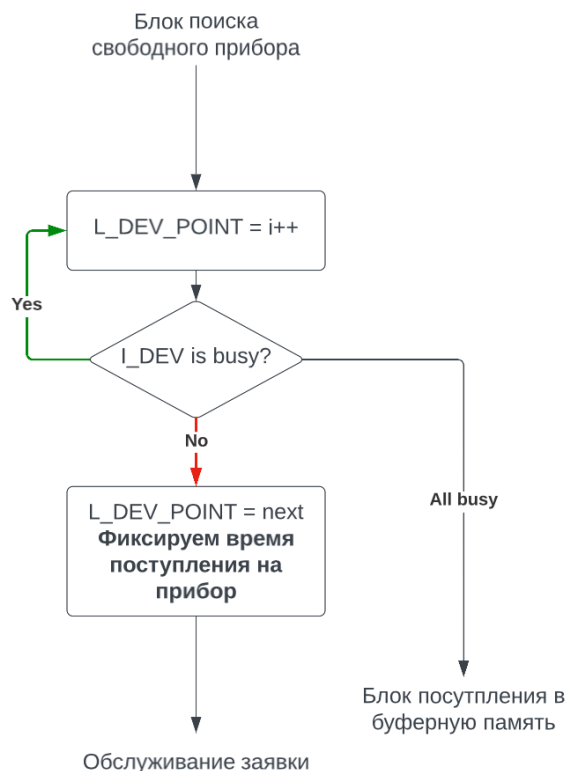


Рис 1. Блок схема поиска свободного прибора



Рис 2. Блок схема выбора заявки

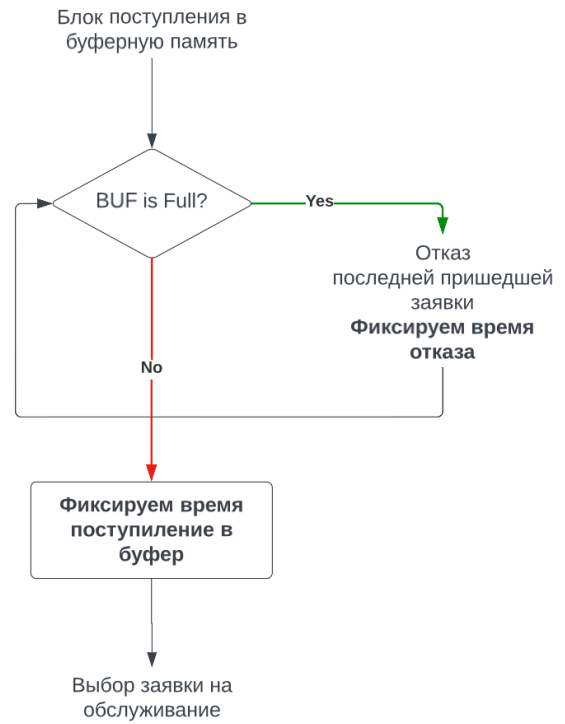


Рис 3. Блок схема поступления в буфер

## Модульная структура и описание модулей

Разработка производилась в среде IntelliJ IDEA на языке Kotlin с использованием графической библиотеки JavaFX. Приложением используется объектно-ориентированную парадигму программирования и содержит набор классов:

class **Bid** - класс заявки

class **Worker** - класс обслуживания заявки

class **Generator** - класс генератора заявки

class **Buffer** - класс хранения заявки в буфере

class **SpecialEvent** - класс наступившего события

class **DispatcherWorker** - класс координирования и выбора устройства обслуживания

class **DispatcherBuffer** - класс координирования мест в буфере

class **MainDispatcher** - класс "обертка" для всех остальных объектов:

- функция **buildModel()** - моделирует систему массового обслуживания
- функция **getResults()** - получение списка событий

Работа приложения начинается с функции **main**, она запускает десктопное приложение, после чего приложение ожидает ввода данных и нажатия кнопок пользователя.

### Первое окно

Создание конфигурации СМО. Пользователь вводит кол-во заявок, кол-во обслуживающих устройств, размер буфер и коэффициенты: альфа, бета, лямбда.

Нажимая на кнопку "Start Modelling", систему моделирует систему и открывает новое окно. Для изменения конфигурации СМО, нужно открыть приложение заново.

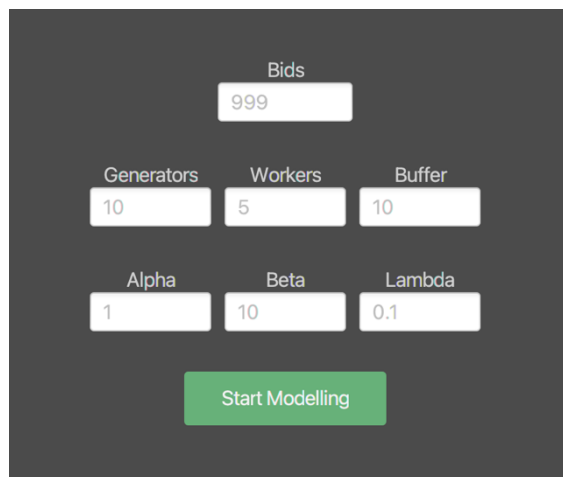


Рисунок 1. Создание конфигурации

## Второе окно

Выбор режима просмотра состояния СМО. Из одного режима в другой можно возвращаться с помощью кнопки "Close".

1. Кнопка "Step by Step" открывает окно пошагового режима.
2. Кнопка "Table Details " открывает окно автоматического режима; отображение двух таблиц.

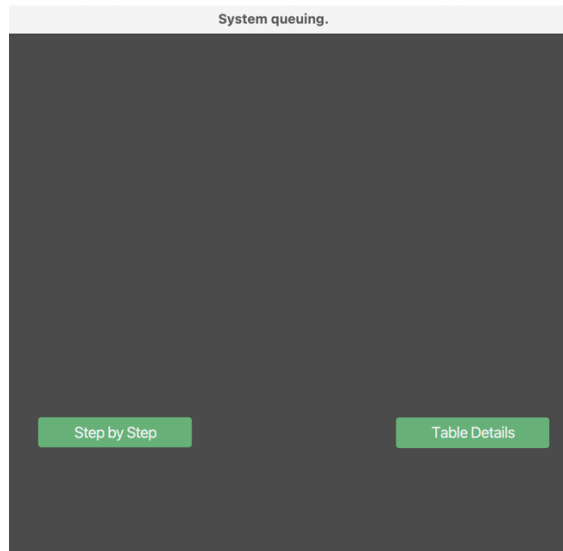


Рисунок 2. Выбор режима

## Третье окно

Пошаговый режим. Нажимая кнопки "Next Step" и "Prev Step" графический интерфейс будет менять свое состояние согласно списку событий СМО.



Рисунок 3. Пример пошагового режима



## Результаты работы имитационной модели

### Определение количества реализаций

Требования:

1. Доверительная вероятность = 0.9 (90%)
2. Количество заявок, необходимое для получения нужной точности при заданной доверительной вероятности, можно оценивать по формуле:

$$N = \frac{t_{\alpha}^2(1-p)}{p\delta^2}, \text{ где}$$

- $p$  - вероятность отказа заявки
- $t_{\alpha} = 1.643$
- $\alpha = 0.9$
- $\delta = 0.1$

Найдем  $N$  итерационным методом начиная с  $N_0 = 100$  - найдем  $p_0$  и по нему найдем  $N_1$  по формуле. Через него находим  $p_1$ : если  $|p_0 - p_1| < 10\% (0.1)$  от  $p_0$ , то  $N_0$  удовлетворяет заданной точности. По результатам работы программы получено, что для достижения заданной точности достаточно рассмотреть **400-500** заявок( интерпретировав на задачу -  $X$  поставок партий вина).

## Рекомендации по выбору конфигурации системы

Изначальная конфигурация для системы

1. 10 источников
2. 100 мест в буфере
3. 1 прибор
4.  $\lambda = 0.9$
5.  $a = 3, b = 6$

Получим следующие результаты:

Источники	Приборы	Буфер	Р отк	Т преб	Р загруз
10	1	100	0,55	66,2	0,996

Загруженность приборов - хорошая, однако вероятность отказа - 36-45 %, что неприемлемо для заданной системы. Попробуем увеличить, кол-во приборов.

Получаем следующие данные:

Источники	Приборы	Буфер	Р отк	Т преб	Р загруз
10	5	100	0,05	0,94	0,44
10	5	200	0,01	1,05	0,45

Вероятность отказа уменьшиться, но загруженность тоже упала, что не подходит для модели, так как получается простой на приборах.

Попробуем подобрать такую систему, чтобы вероятность отказа падала, а загруженность была высокой. Увеличим кол-во источников до 5 и будем варьировать кол-во приборов от 3-5.

Источники	Приборы	Буфер	Р отк	Т преб	Р загруз
15	3	100	0,18	10,65	0,96
15	3	200	0,1	7,20	0,96
15	4	100	0,09	1,50	0,74
15	4	200	0,01	1,16	0,87
15	5	100	0,000	1,34	0,64
15	5	200	0,000	1,21	0,65

Результаты моделирования стали лучше: вероятность отказа, для 4-5 приборов удовлетворяют требованиям системы. Однако, система все еще не до конца нагружена на 90%. Вероятность отказа в этом эксперимента стремится к нулю, однако загруженность приборов находится от 87% до 64%.

Следовательно, надо еще увеличить кол-во источников, так как у нас есть "запас" в вероятности отказа (должно быть <5%). Возьмем 20 источников в системе.



Источники	Приборы	Буфер	Р отк	Т преб	Р загруз
20	3	100	0,48	32,1	0,998
20	3	200	0,36	25,7	0,998
20	4	100	0,23	9,9	0,991
20	4	200	0,11	9,9	0,991
20	5	100	0,01	2,3	0,988
20	5	200	<b>0,00</b>	1,8	<b>0,988</b>

Можно увидеть, что мы добились оптимальной конфигурации по проценту загруженности: >98,8%. Вероятность отказа для 3 приборов существенна: >36% - не подходит. При 4 приборах: >11% - не подходит. А вот уже для 5 приборов: <5% - получаем систему, подходящую под начальные требования.

Для 25 источников имеем следующие результаты моделирования:

Источники	Приборы	Буфер	Р отк	Т преб	Р загруз
25	5	100	0,18	30,8	0,992
25	5	200	0,07	28,7	0,981

Система уже не удовлетворяет требованиям. Если мы и дальше будем наращивать число источников в пределах 5 приборов, тогда вероятность отказа будет еще расти. Следовательно предыдущую конфигурацию (с 20 источниками) - можно считать оптимальной.

В итоге, изначальная система может быть заменена более оптимальной конфигурацией:

1. 20 источников
2. 200 мест в буфере
3. 5 приборов
4.  $\lambda = 0.9$
5.  $a = 3, b = 6$

## **Вывод**

В ходе курсовой работы было создано приложения для моделирования системы массового обслуживания на языке Kotlin с использованием графической библиотеки JavaFx. Реализованы два вида режима моделирования - пошаговый и автоматический.

С помощью данной программы была проанализирована реальная система и подобрана максимально выгодная комплектация данной системы.

## Приложение

Код реализованной программы находится по ссылке:

**<https://github.com/jekasrs/queuing-system>**