Федеральное государственное бюджетное образовательное учреждение высшего образования «Сибирский государственный университет телекоммуникаций и информатики» (СибГУТИ)

Кафедра Вычислительных Систем (ВС)

**ОТЧËТ ПО ЗАЧЁТНОМУ ЗАДАНИЮ**

по дисциплине «Моделирование»

Выполнили:

Студенты гр. ИВ-622

Тимофеев Д.А. Ольман Р.Ю.

Неудахин В.Д.

Проверил:

Доцент кафедры ВС Родионов А.С.

Новосибирск 2020 Содержание

**Оглавление**

[Постановка задачи 3](#_Toc39710744)

[Теоретические сведения 4](#_Toc39710745)

[Средство разработки 5](#_Toc39710746)

[Ход работы 6](#_Toc39710747)

[Случайный порядок 7](#_Toc39710748)

[Короткая очередь 8](#_Toc39710749)

[Вывод 9](#_Toc39710750)

# Постановка задачи

Смоделировать в программе AnyLogic систему массового обслуживания на примере аттракционов. Существует обработчиков-каруселей. Для каждого обработчика определена своя очередь. Каждый посетитель должен пройти каруселей. Есть предельное время нахождения в системе.

Необходимо узнать: среднее время пребывания в системе и процент не успевших использовать билеты:

1. Выбирается самая короткая очередь из оставшихся

2. Выбирается случайная из оставшихся очередей

Моделирование проводить со следующими параметрами:

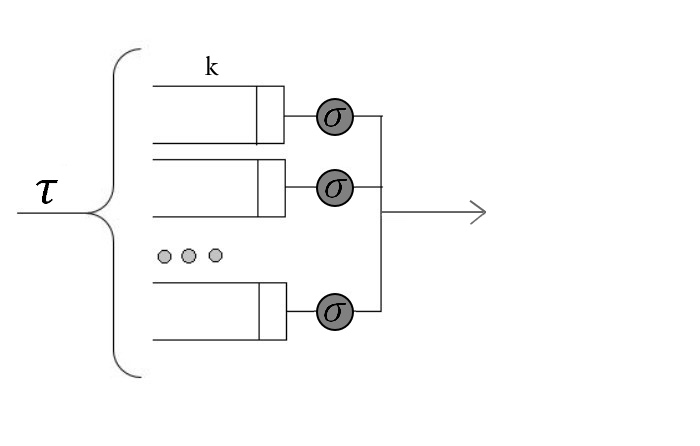
– интенсивность потока агентов

–время работы одного аттракциона

– количество очередей

– количество билетов на одного агента

Т – максимальное время пребывания в системе



*Рисунок 1 – Общая схема модели*

# Теоретические сведения

Системой массового обслуживания(СМО) называется система, в которой: возникают массовые требования на выполнения каких-либо видов услуг, происходит удовлетворение этих требований - обслуживание Главной особенностью процессов массового обслуживания является их случайность. Выделяются две взаимодействующие стороны, одна из которых обслуживает, а вторая выступает в качестве обслуживаемой. Присутствие случайности в поведении одной из сторон приводит к случайному протеканию всего процесса обслуживания. Причины случайности заключаются в массовом характере потребностей, а также в случайности работы обслуживающей системы.

Классическая СМО содержит от одного до бесконечного числа обслуживающих приборов. В зависимости от наличия возможности ожидания поступающими требованиями начала обслуживания СМО подразделяются на:

* Системы с потерями, в которых требования, не нашедшие в момент поступления ни одного свободного прибора, теряются;
* Системы с ожиданием, в которых имеется накопитель бесконечной ёмкости для буферизации поступивших требований, при этом ожидающие требования образуют очередь;
* Системы с накопителем конечной ёмкости (ожиданием и ограничениями), в которых длина очереди не может превышать ёмкости накопителя; при этом требование, поступающее в переполненную СМО (отсутствуют свободные места для ожидания), теряется.
* Системы с потерями, в которых требования находятся в очереди лишь ограниченное время, после чего теряются.

# Средство разработки

AnyLogic — программное обеспечение для имитационного моделирования, разработанное российской компанией The AnyLogic Company. Инструмент обладает современным графическим интерфейсом и позволяет использовать язык Java для разработки моделей. Для моделирования СМО требуется знать следующие компоненты:

Source — создает агентов. Обычно используется в качестве начальной точки потока агентов.

SelectOutput — объект направляет входящих агентов в один из двух выходных портов в зависимости от выполнения заданного условия. Условие может зависеть как от агента, так и от каких-то внешних факторов. Поступивший агент покидает объект SelectOutput в тот же момент времени. Может использоваться для сортировки агентов согласно заданному критерию, для случайного разделения потока агентов на части и т.д.

Queue — моделирует очередь агентов, ожидающих приема объектами, следующими за данным в потоковой диаграмме, или же хранилище агентов общего назначения

Delay — задерживает агентов на заданный период времени. Время задержки вычисляется динамически, может быть случайным, зависеть от текущего агента или от каких-то других условий. Сразу несколько агентов (не более заданной вместимости объекта capacity) могут быть задержаны одновременно или независимо друг от друга.

Sink — уничтожает поступивших агентов. Обычно используется в качестве конечной точки потока агентов.

# Ход работы

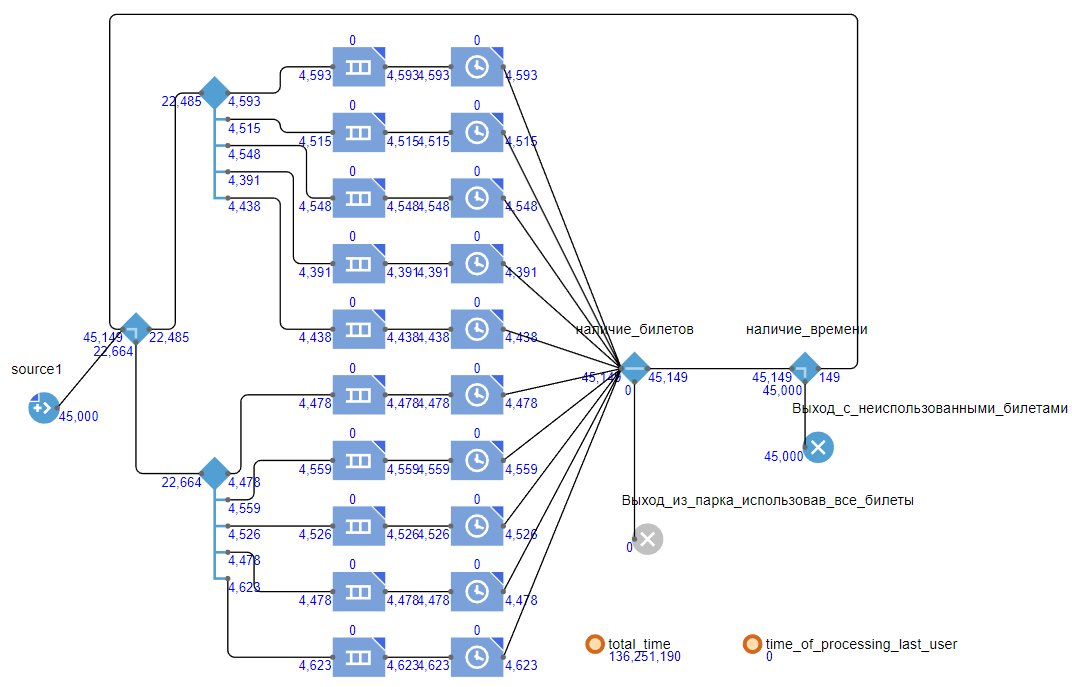
Посетители приходят на аттракционы, у каждого из них уже имеется запас билетов и время, которое они могут потратить на аттракционы. Создается 10 очередей, каждая ведет к своему аттракциону, выход из системы будет возможен по следующему алгоритму:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | если время нету && билеты есть  выход из парка с неиспользованными билетами,  если время есть && билеты есть  продолжаем кататься,  если билеты закончились  выход из парка, использовав все билеты, |

Источником наших посетителей является элемент Source. Интервал прибытия посетителей составляет exponential *(1)* в минуту, максимальное количество посетителей составляет 1000. Для выбора посетителем одного из нескольких аттракционов используются условные операторы, такие как SelectOutput и SelectOutput5. Их разница в количестве выходов у первого их два, а у второго 5. Около каждого аттракциона есть очередь, она создается при помощи компонента Queue. Сам аттракцион имитируется компонентом Delay, он имеет время на задержку посетителя. Компонент Sink служит выходом из системы для посетителя.

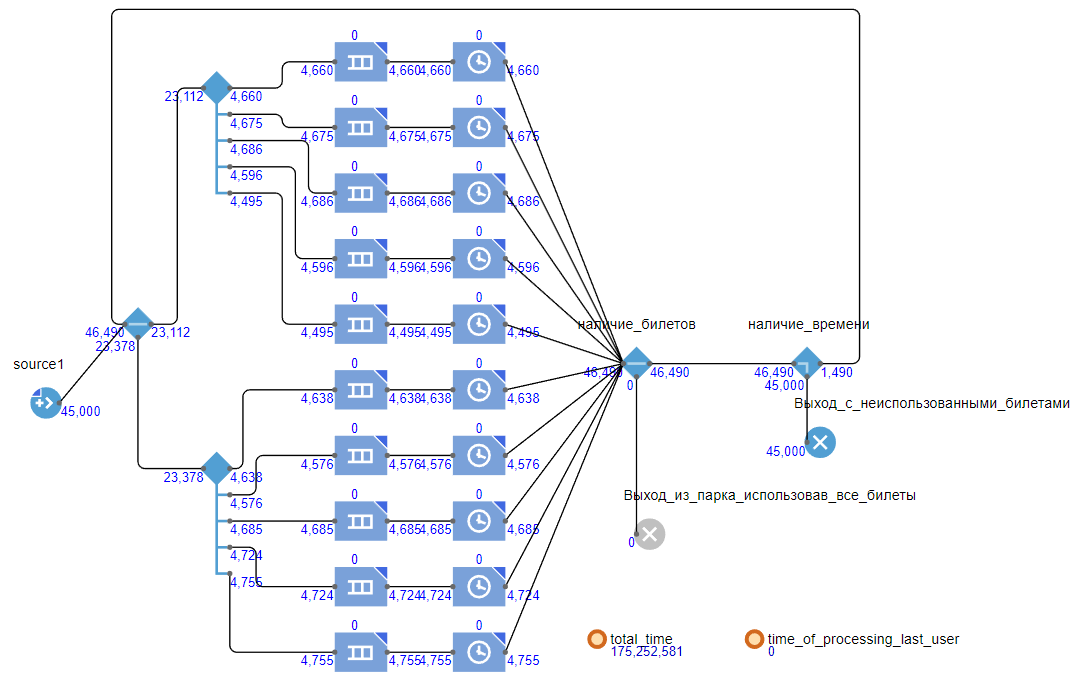
## Случайный порядок

* Реализация случайного порядка выбора очереди с изначальными параметрами , , = 10, = 6, T = 10. То есть посетитель будет выбирать с вероятностью в 0.2 аттракцион и вставать к нему в очередь. Результат работы можно увидеть на рисунке 2.



*Рисунок 2 - Результат модели со случайным порядком очереди эксперимент 1*

* Замена общего времени нахождения в системе пользователя результатов не дала. Все посетители по прежнему уходили из системы с неиспользованными билетами. Рисунок – 3.



*Рисунок 3 – Результат модели со случайным порядком очереди эксперимент 2*

* А вот уменьшение количества билетов для каждого пользователя дали небольшие результаты, посетители начали выходить из системы по исчерпанию билетов, а не по общему времени. Рисунок – 4.

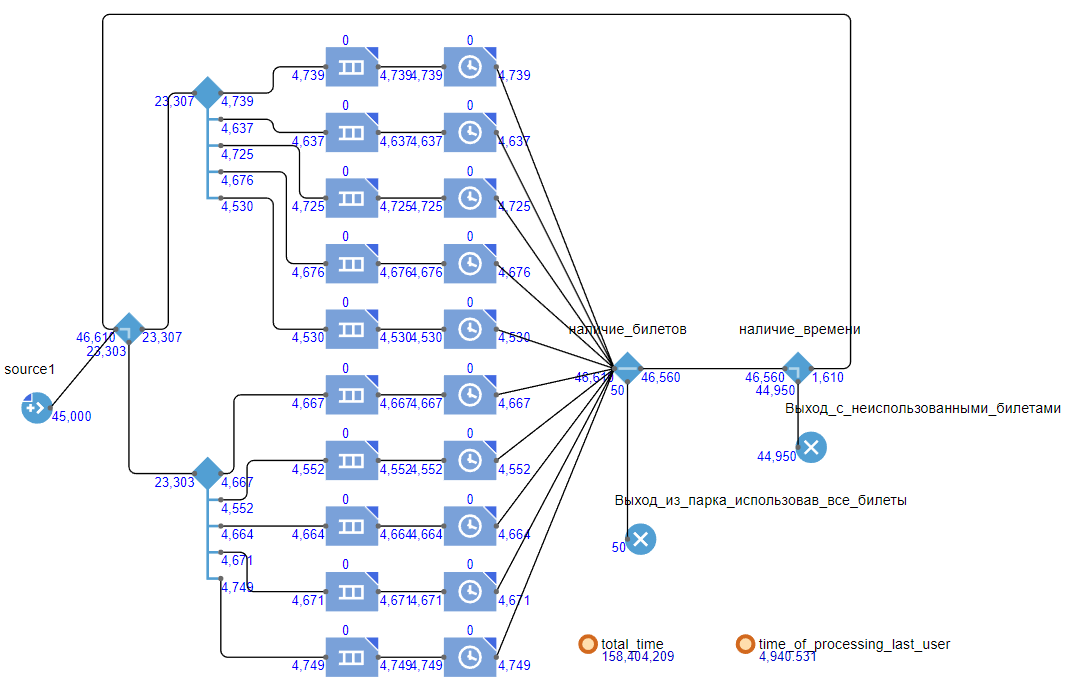


Рисунок 4 - *Результат модели со случайным порядком очереди эксперимент 3*

* Уменьшение exp примерно в 6 раз, дает прирост вышедших посетителей по билетам в примерно в 120 раз. Рисунок - 5.

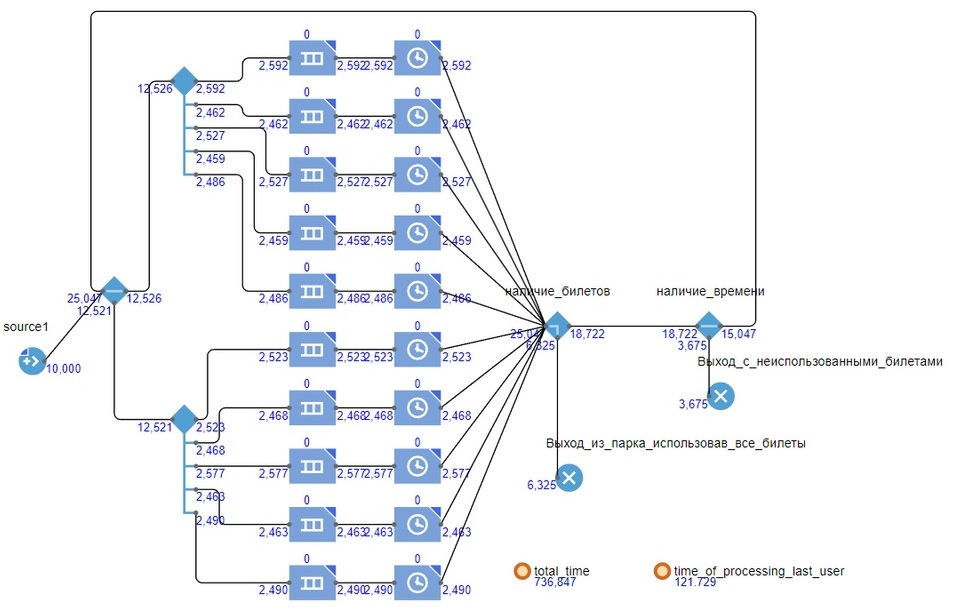
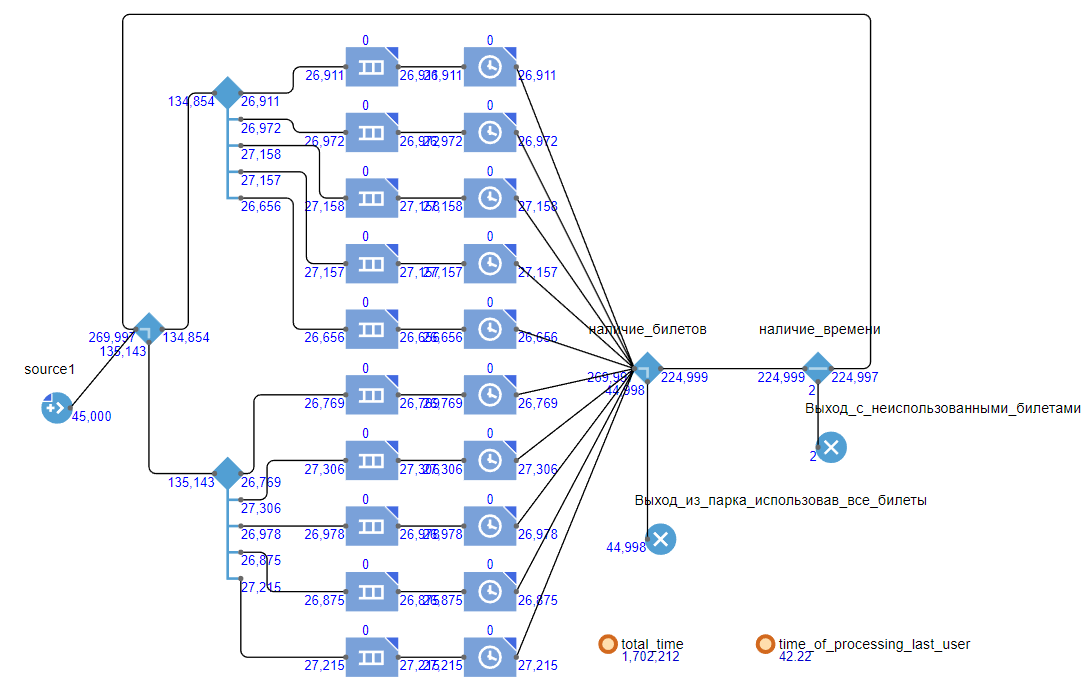


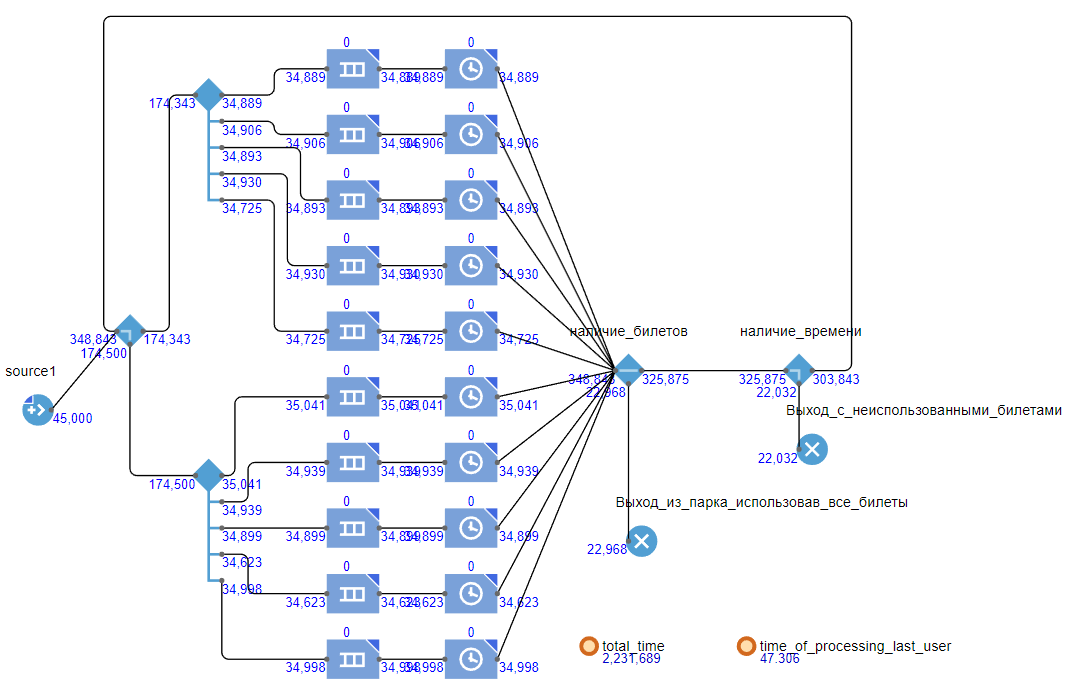
Рисунок 5 - *Результат модели со случайным порядком очереди эксперимент 4*

## Хотелось найти максимальное количество пользователей, которые могут покинуть аттракционы, использовав все билеты. Получились такие параметры: , , = 10, = 5, T = 120.



*Рисунок 6 - Результат модели со случайным порядком очереди эксперимент 5*

* И так же было бы неплохо найти параметры, которые давали бы 50% результат в обоих случаях. **, , =** 10, = 8, T = 50.

****

*Рисунок 7 - Результат модели со случайным порядком очереди эксперимент 6*

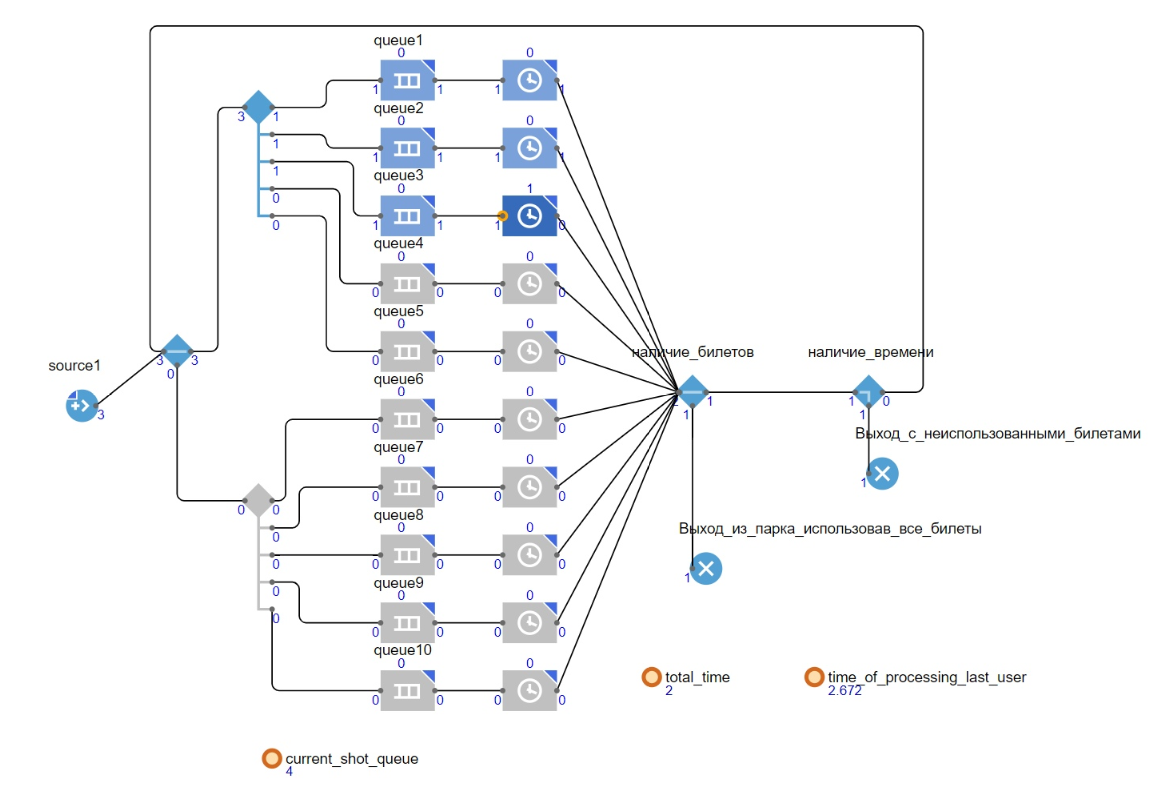
*Таблица 1 – Результаты экспериментов*

|  |  |
| --- | --- |
| **% потерь** | **Среднее время пребывания посетителя** |
| 100% | 3027.804 минуты |
| 100% | 3894.502 минуты |
| 99% | 3520.094 минуты |
| 36% | 16.374 минуты |
| 0.004% | 37.938 минуты |
| 48% | 49.593 минуты |

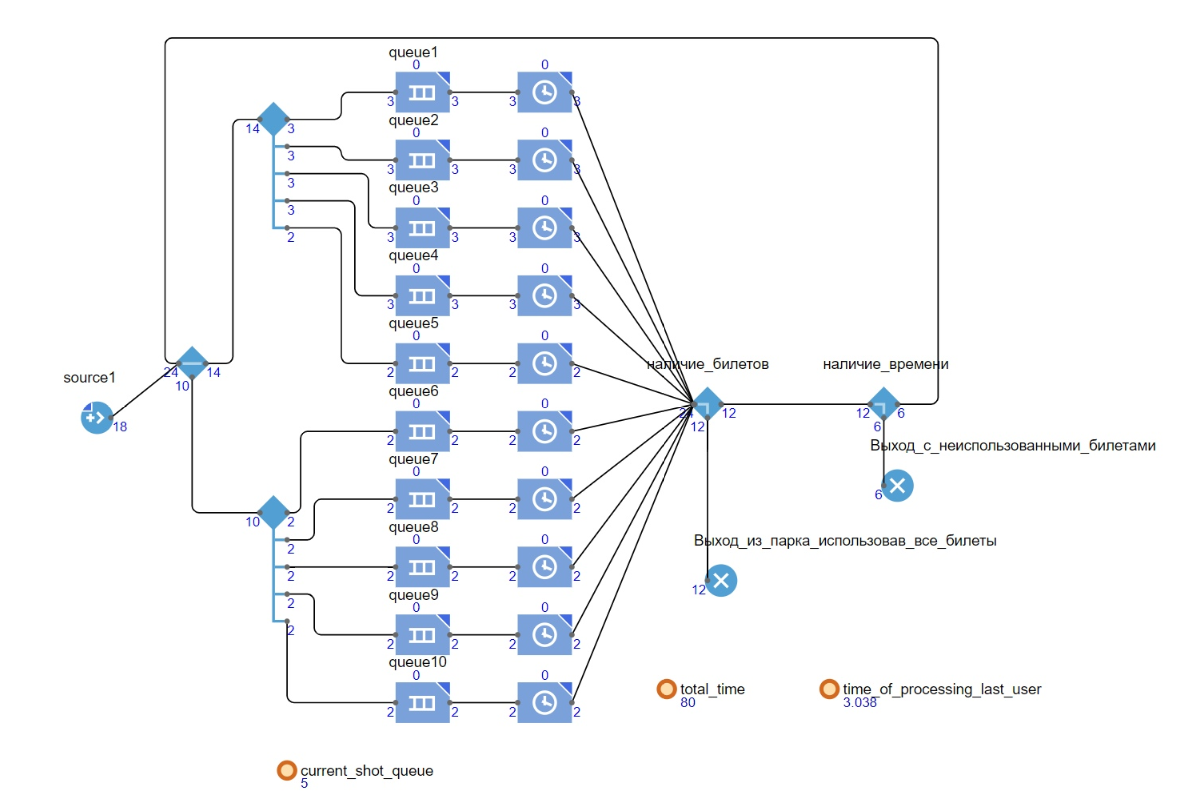
## Короткая очередь

Реализация короткой очереди с параметрами ,

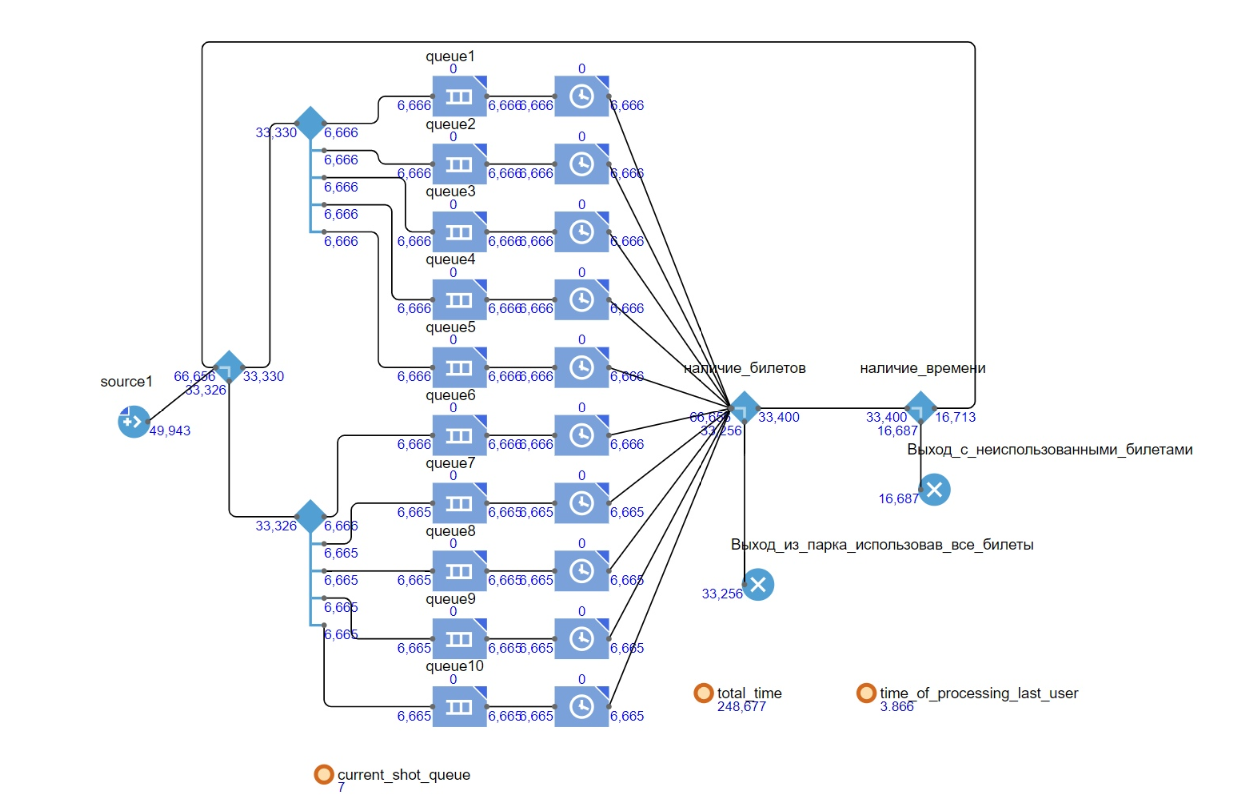
, = 10, = 6, T = 10. Т.е. посетитель будет смотреть, где в очереди стоит меньше людей, туда он и встанет. Результат работы можно увидеть на рисунке 3.



*Рисунок 8 - Результат модели с короткими очередями*



*Рисунок 9 - Результат модели с короткими очередями*



*Рисунок 10 - Результат модели с короткими очередями*

На маленьких числах процент потерь и среднее время пребывания варьируются в пределах 0.05. Работу системы до 1042 людей считаем за "разогрев системы" (данные, которые не учитываются, так как система не стабилизировалась).

# Вывод

Были проведены эксперименты, в ходе которых было выяснено, что при исходных параметрах все посетители уходят, не потратив все билеты,

Вне зависимости от максимального времени пребывания в системе, большая часть задач уходила с потерями.

Но мы видим, что при уменьшении количества билетов появляется небольшое количество людей, которые потратили все билеты. То есть, за исключением нескольких везунчиков, все люди ушли из парка, отстояв очередь и потеряв билеты.

Также можем заметить, что при одновременном увеличении максимального времени пребывания в системе и уменьшении количества билетов - почти все пользователи выходят, потратив все билеты (почти нету потерь.)

Доля потерь (выхода из парка с неиспользованными билетами) при исходных данных была очень большой. Это было потому что у посетителей было много билетов и мало времени покататься на аттракционах.

Была найдено оптимальное соотношение параметров, при которых большая часть задач обрабатывалась почти без потерь.

Исходя из результатов можно сделать вывод, что в обоих случаях получились похожие значения. Это результат того, что в обоих случаях идет равномерная нагрузка на аттракционы (в одном случае случайность, в другом преднамеренно запрограммированная равномерная нагрузка).