Практическая 19-20

Тема: Имитационное моделирование. AnyLogic разработка типового проекта работы одного терминала.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Source

Создает агентов. Обычно используется в качестве начальной точки потока агентов.

Агенты могут быть стандартными или заданными пользователем агентами типа **Agent**. Вы можете сконфигурировать блок так, чтобы он создавал агентов других типов, указав конструктор нужного типа в параметре **Новый агент**, а также задать действие, которое должно выполняться перед тем, как новый агент покинет блок **Source**, в поле действия **При выходе**.

Есть несколько способов задания того, сколько агентов и когда должен создавать этот блок.

Агенты могут создаваться согласно заданной интенсивности (которая может изменяться динамически с помощью функции set_rate()), времени между прибытиями, изменяющейся во времени интенсивности, заданной с помощью расписания, расписанию, задающему точные времена и количество прибывающих агентов, или "вручную" путем вызова функции блока inject(). Например, пуассоновский поток агентов может быть промоделирован путем генерации агентов согласно заданной интенсивности, или согласно времени между прибытиями, подчиняющемуся экспоненциальному закону распределения. Может быть задано как максимально допустимое число генераций, так и число агентов, создаваемых за каждый раз.

Queue

Блок **Queue** моделирует очередь агентов, ожидающих приема блоками, следующими за данным в потоковой диаграмме, или же хранилище агентов общего назначения. При необходимости вы можете задать максимальное время ожидания агента в очереди. Вы также можете программно извлекать агентов из любых позиций в очереди.

Поступающие агенты помещаются в очередь в определенном порядке: либо согласно правилу FIFO (First In, First Out, первый поступивший в очередь агент будет и извлекаться из очереди первым), LIFO (Last In, First Out, первым из очереди будет извлекаться последний поступивший в очередь агент), согласно приоритетам агентов, либо согласно сравнению агентов друг с другом. Приоритет может либо явно храниться в агенте, либо вычисляться согласно свойствам агента и каким-то внешним условиям. Очередь с приоритетами всегда примет нового входящего агента, вычислит его приоритет и поместит его в очередь в позицию, соответствующую его приоритету. Если очередь будет заполнена, то приход нового агента вынудит последнего хранящегося в очереди агента покинуть блок через порт outPreempted (но если приоритет нового агента не будет превышать приоритет последнего агента, то тогда вместо него будет вытеснена именно этот новый агент).

В режиме таймаута агент покинет очередь через порт outTimeout, если проведет в очереди заданное количество времени.

Вы можете извлекать любых агентов из очереди с помощью функции remove(). В некоторых случаях, например, когда блок **Queue** используется для моделирования хранилища с произвольным доступом, имеет смысл оставлять порт out несоединенным и извлекать агентов из очереди с помощью функции remove(). Извлеченные из очереди агенты могут быть вставлены в другие процессы с помощью блоков **Enter**.

Вы можете расширять функциональность блока **Queue**, выполняя необходимые вам действия в момент помещения агента в очередь, при достижении им начала очереди, а также при уходе агента из блока через любой из его портов. Пожалуйста, обратите внимание, что на момент вызова кода параметра onEnter агент уже будет помещен в очередь, а на момент вызова кода параметров onExit или onExitTimeout будет удален из очереди.

Вы можете динамически изменять вместимость очереди.

Агенты в очереди могут отображаться на презентации как стоящими один за другим (тип анимации **Путь**), так и стоящими в как-то по-другому заданных местах.

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 блока capacity) могут быть задержаны одновременно или независимо друг от друга.

Пример задания времени задержки: пусть время обработки пакета данных (агент типа Packet) пропорционально размеру пакета (для задания размера пакета вы можете создать параметр size в типе агента Packet). Тогда вы можете сделать следующее: указать Packet в качестве **Типа агента** блока **Delay** и написать processingTimePerDataUnit * agent.size в поле параметра **Время задержки**.

Если вместимость блока **Delay** меняется динамически, и количество агентов, находящихся в блоке в данный момент времени, превышает значение вместимости блока, то блок **Delay** даст каждому такому агенту завершить его время ожидания, и не будет принимать новых агентов до тех пор, пока их количество в блоке не станет меньше нового значения вместимости блока.

Вы можете выполнять любые необходимые вам действия над содержащимися в блоке агентами, например, узнавать, сколько времени еще агентам осталось находиться в блоке, и даже извлекать агентов, не дожидаясь того, когда их времена задержек истекут.

Блок **Delay** может отображать анимации находящихся в нем агентов как движущимися вдоль заданного пути, так и ожидающими в заданных точках. В том случае, если агенты отображаются движущимися, за время их задержки они должны будут пройти весь путь выбранной фигуры анимации блока **Delay**.

Вы можете динамически увеличить длительность задержки для тех агентов, которые уже находятся в состоянии задержки, используя функцию extendDelay().

Параметры

Тип задержки

Определяет способ задания времени задержки: задержка может быть задана либо как **Определенное время**, либо **До вызова функции stopDelay()**.

Время задержки

[Параметр виден, если **Тип: Определенное время**] Выражение, вычисляющее время задержки для агента.

Вместимость

[Параметр виден, если не выбрана опция **Максимальная вместимость**] Вместимость блока **Delay**. Задает максимальное количество агентов, которые могут одновременно находиться в блоке.

Максимальная вместимость

Если опция выбрана (true), то вместимость блока **Delay** будет максимально возможной (ограничена константой Integer.MAX_VALUE).

Место агентов

Фигура разметки (узел или путь), где располагаются агенты, пока они обрабатываются блоком **Delay**.

Sink

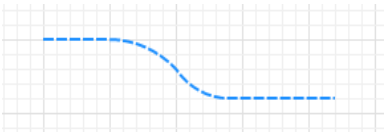
Уничтожает поступивших агентов. Обычно используется в качестве конечной точки потока агентов.

Для того, чтобы агенты удалялись из модели и уничтожались, нужно соединить выходной порт последнего блока процессной диаграммы с портом блока **Sink** или **Exit**.

Для успешного уничтожения агента необходимо выполнение трех условий:

1. Если агент находится в сети, то он должен быть удален из этой сети.
2. Агент не должен обладать ни одним ресурсом или сетевым ресурсом.
3. Если агент содержит других агентов, то они тоже должны удовлетворять вышеуказанным условиям.

Если какое-то из этих условий не выполняется, блок **Sink** выдает ошибку.



Путь

Пути и Узлы являются элементами разметки пространства, которые задают местоположение агентов в пространстве.

- **Путь** графически задает траекторию движения агентов из одного места в другой.
- **Узел** задает место, в котором агенты могут находиться.

Узлы могут соединяться путями. Вместе они образуют сеть. Также AnyLogic автоматически создает отдельную сеть для каждого пути, не соединенного с узлами. В сети узел задает место, где агенты могут останавливаться, тогда как пути, соединяющие узлы, задают маршрут, по которому агенты следуют при движении из одного узла в другой. Перемещение всегда происходит по самому короткому пути между

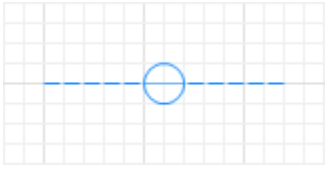
узлами пунктов отправления и назначения. Агенты и ресурсы могут иметь индивидуальную скорость, кроме того, их скорость может изменяться динамически. Например, вы можете задать разную скорость для загруженного и свободного автопогрузчиков. Предполагается, что сегменты пути имеют неограниченную вместимость, и таким образом агенты, движущиеся вдоль сегмента, не мешают друг другу.

Вы можете выбрать путь в качестве **Места агентов** в параметрах следующих блоков Библиотеки Моделирования Процессов:

- Delay
- Queue
- Match
- Combine
- Seize
- Service
- Conveyor
- Batch
- RackStore

Рисование пути

Путь может состоять из нескольких сегментов. Каждый сегмент может быть либо линейным, либо дуговым.



Точечный узел

Узлы и пути являются элементами разметки пространства, которые задают местоположение агентов в моделируемом пространстве. Узлы могут соединяться путями. Вместе они образуют сеть. В сети, узел задает место, где агенты могут останавливаться, тогда как пути, соединяющие узлы, задают маршрут, по которому агенты следуют при движении из одного узла в другой.

Чтобы добавить точечный узел

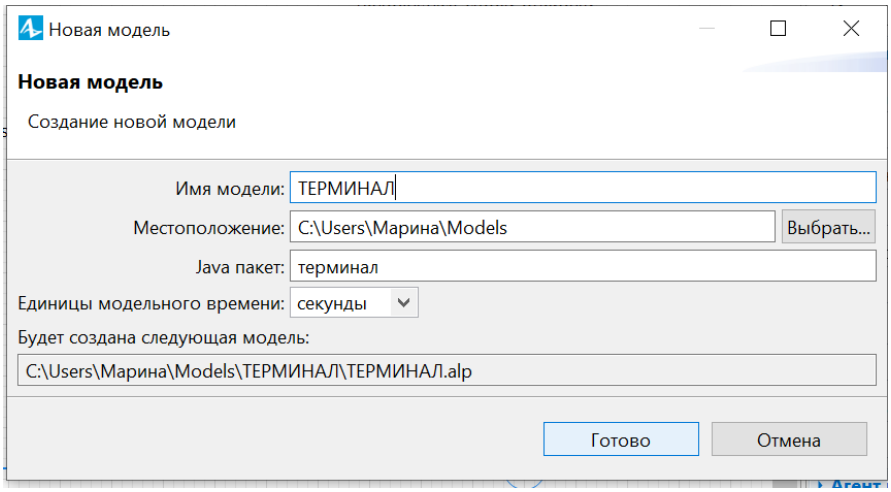
1. Перетащите элемент **Точечный узел** из палитры **Разметка Пространства** в графический редактор.

По умолчанию внутри точечного узла может одновременно находиться не больше одного агента. Если вам нужно нарисовать область, в которой одновременно могут находиться несколько агентов, мы предлагаем использовать **Прямоугольный узел** или **Многоугольный узел**.

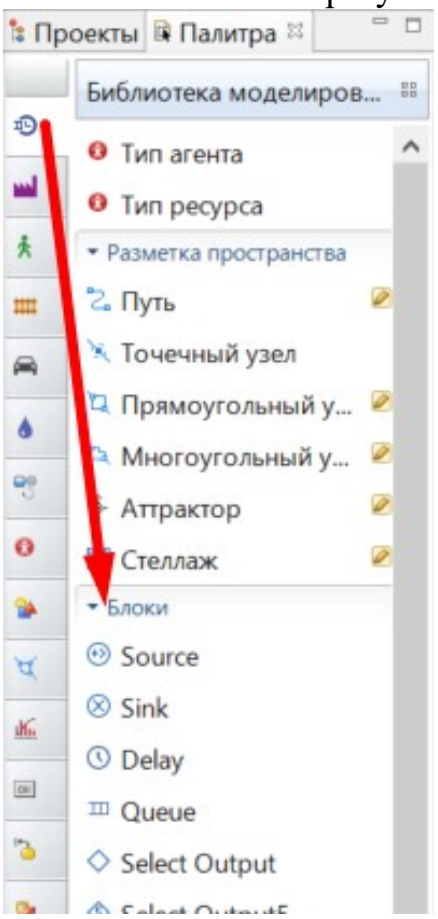
ХОД РАБРТЫ

Задание 1

1. Создайте новую модель под название терминал



2. После чего сразу в свойствах переименуйте main на терминал.



Ваша задача будет создать ресурс, очередь ресурсов к терминалу и завершающий процесс.

Для этого добавьте на рабочую область **терминал** из библиотеки моделирования процессов, элемент **source, queue, delay и sink** и соедините все элементы между собой.

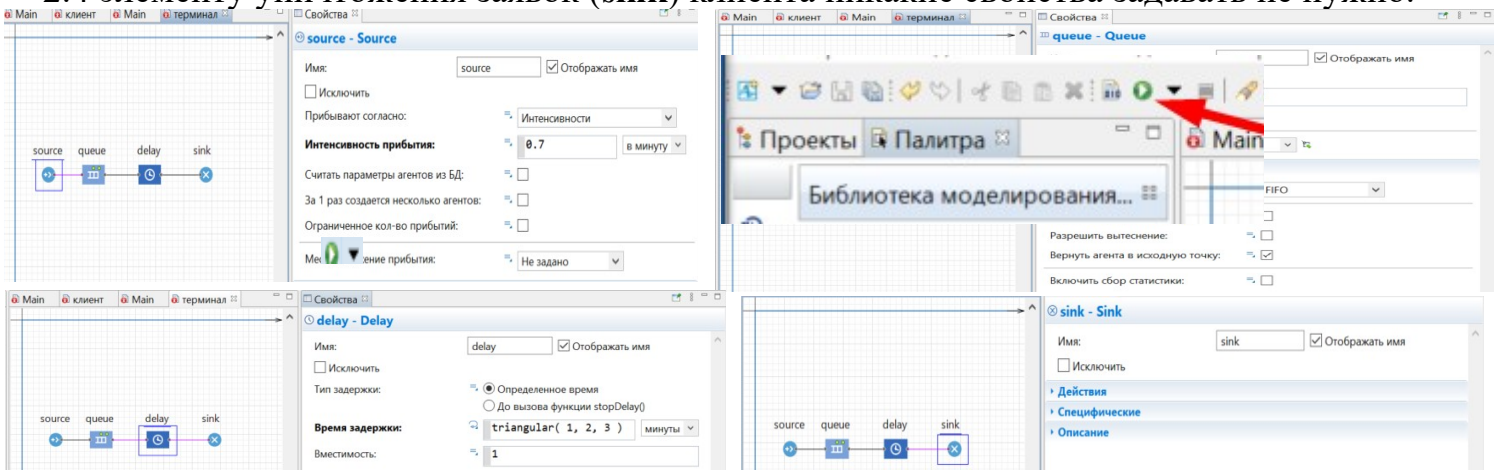
2.1 в свойствах ресурса(**source)** укажите **интенсивность прибывания** 0.7 (это указывает какое количество людей будет приходить в ваше заведение)

2.2 элементу очередь(**queue**) в свойствах объекта, укажите вместимость количества людей 500. В свойства *специфические* => у вас очередь должна быть указана FIFO(первый вошел-первый вышел)

2.3 элементу задержке заявок (**delay**) в свойствах укажите приблизительное *время задержки* одного клиента у терминала. `Triangular(1, 1.5, 3)`, где 1-это минимальное время нахождение у банкомата, 1.5- это наиболее вероятное время нахождения у

банкомата, 3-это максимальное время нахождения у банкомата. *Вместимость* укажите 1-это сколько человек за раз может находится за одним терминалом.

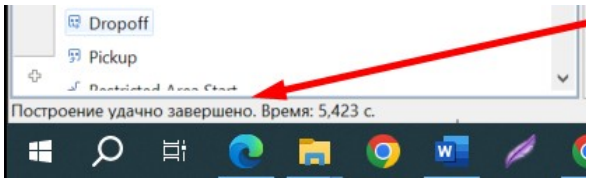
2.4 элементу уничтожения заявок (**sink**) клиента никакие свойства задавать не нужно.



П

е

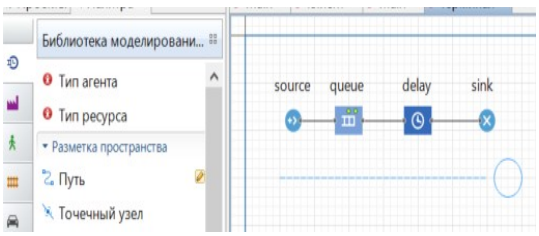
чего проверьте модель на ошибки. Для этого нажмите клавишу **F7,** для запуска модели нажмите клавишу **F5.** *Рекомендация перед запуском всегда проверять модель на ошибки! Либо можете нажимать в меню навигации*



2.6 если снизу рабочей модели написано: Построение удачно завершено, то в этом случае можете запускать модель на просмотр, если там написана ошибка, сначала тогда исправьте ошибку, прежде чем запускать модель.

2.7 Теперь изменить интенсивность прибывания людей с минут на секунды и время задержки у терминала тоже укажите секунды! Для того, чтобы более быстро показать процесс загрузки терминала и обработки клиентов. результат, при нажатии на любой элемент вы увидите статистику каждого элемента

2.8 Далее создадим анимацию для нашей модели, как люди подходят к банкомату и как быстро ублажаться:

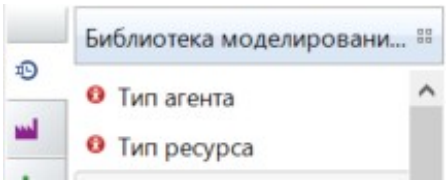


2.8.1 добавьте элемент **путь** и элемент **точечный узел (**банкомат наш**)**, перетащим их на рабочую область.

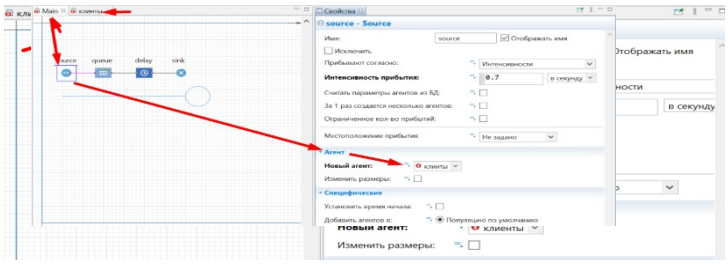
2.8.2 Нажмите на узел и в свойствах установите ему динамическое значение, это будет значить, что во время анимации цвет будет меняться с красного на зеленый. Напишите в поле такое значение:

```
delay.size()>0? red: green
```

2.8.3 Далее для анимации нам нужны люди, те кто будут подходить к терминалам и обслуживаться. Для этого из библиотеки моделирования процессов перетяните объект **Тип агента** на рабочую область. В диалоговом окне укажите имя нового агента: **клиенты** нажмите **Далее**,выберите **человек,** нажмите, **Далее** и после чего нажмите **ГОТОВО.**

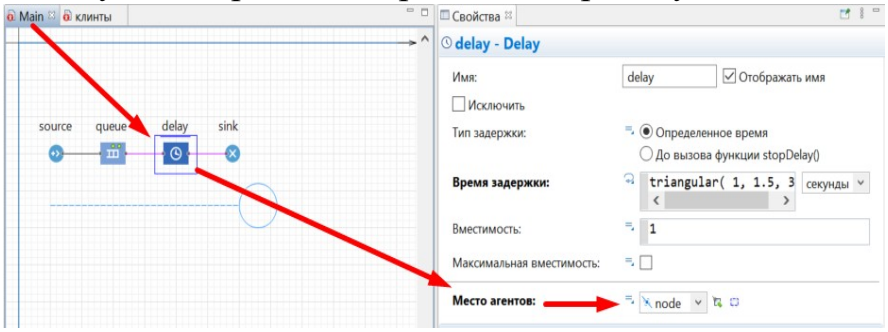


У ВАС ПОЯВИЛОСЬ ПУСТОЕ РАБОЧЕЕ ПОЛЕ, ДЛЯ ПРОДОЛЖЕНИЯ РАБОТЫ ПЕРЕЙДИТЕ НА ВКЛАДКУ main.



2.8.4 перейдите на вкладку **Main** нажмите на элесмент **sourse** , в свойствах выберите **Новый агент** и установите ему созданный тип агента, в нашем случае-это **клиенты** .После чего всей нашей схеме присвоится тип агента клинты.

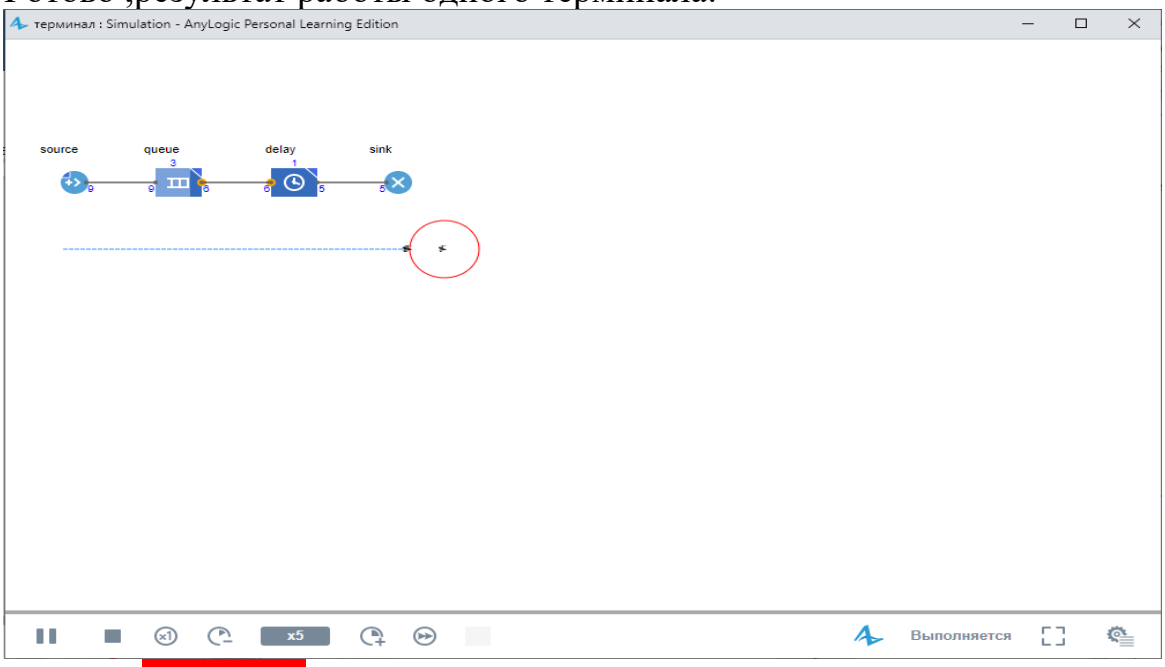
2.8.5Далее нам нужно указать путь нашим клиентам. Для этого нажимаем на элемент **очередь (**queue) и указываем ему в свойствах **место агентов** из выпадающего списка **path** наш путь,который мы перенесли на рабочую область в виде прямой.



2.8.6 Далее нажмите элемент задержка заявок (**delay**) и укажите ему в свойствах **Место агентов** из выпадающего списка **узел node.** Это укажет очереди, что это будет финальный этап обработки, куда должны попасть клиенты.

2.8.7 Нажимаем F7 после чего F5.

Готово ,результат работы одного терминала:



Задание 2

1. Измените интенсивность прибывания людей с 0.7 в сек на 0.2 в сек.
2. Время обработки заявок измените с 3 сек максимального времени обработки заявок на 10 сек, среднее измените на 5 и минимальное на 2.
3. Переименуйте элементы на:

-клиенты

-очередьКлиентов

-Выход

4. измените толщину линии пути на 3 pt, сделайте ее красной, сделайте так, чтобы отображалось имя пути на схеме. Для этого откройте свойства пути > Специфические, после чего измените имя пути на ОЧЕРЕДЬ1
5. Переименуйте node на узел1.
6. укажите очереди людей уход по таймеру, если клиент ждет больше 10 сек, человек покидает очередь, должно быть два выхода в вашей имитационной системе.

7.Установите динамическое изменения цвета пути, как на узле, но при этом по умолчанию. Чтобы линия была красная.

8. Ускорьте процесс работы терминала в 50 раз и проанализируйте его работу. Результат анализа и скрин работы терминала сделайте по прохождению 10 секунд реального времени.

10. Добавьте очереди возможность вытеснять людей автоматически, и назовите это вытиснениеизОчереди и выведите имя на экран.

11. Ускорьте процесс работы терминала в 50 раз и проанализируйте его работу. Результат анализа и скрин работы терминала сделайте по прохождению 10 секунд реального времени.

12. Сократите длину пути (очереди 1) на 60%

13. Запустите имитацию ускорьте работу и проанализируйте работу терминала.

!!! для переимебнования объектов лучше использовать сочетание клавиш Ctrl+Enter

Контрольные вопросы:

1. Когда срабатывает вытеснение? На какое свойство очереди оно опирается?
2. Что такое очередь? какие основные свойства знаете?
3. На что влияет элемент delay?
4. Назовите основные свойства элемента source.

для чего добавляют тип Агента на схему? Как сделать его присоединённым к схеме определенной?

5. Точечный узел присоединяется к элементу очередь или к обработке заявки?
6. Можно ли вытеснение и уход из очереди присоединить к одному элементу выхода?

Содержание отчета:

1. Тема, цель практической работы
2. Поэтапное описание выполнения практической работы
3. Скриншоты или результат практической
4. Краткие ответы на контрольные вопросы