АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ
ПРОФЕССИОНАЛЬНАЯ ОБРАЗОВАТЕЛЬНАЯ ОРГАНИЗАЦИЯ
«Международный Колледж Бизнеса и Дизайна»
(АНО ПОО «Международный Колледж Бизнеса и Дизайна»)

УТВЕРЖДАЮ
Директор АНО ПОО «МКБиД»
_____ Н.Н.Репин
«22» августа 2023 г.

СБОРНИК МЕТОДИЧЕСКИХ УКАЗАНИЙ

Для студентов по выполнению практических работ
по учебной дисциплине МДК.03.01 Моделирование и анализ программного обеспечения

программы подготовки специалистов среднего звена
по специальности: 09.02.07 «Информационные системы и программирование»

2023 год

Тема: разработка диаграммы последовательности

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

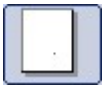
Основные элементы диаграммы - участник (actor) и прецедент (вариант).

Участник - это множество логически связанных ролей, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). Участником может быть человек или другая система, подсистема или класс, которые представляют нечто вне сущности. Графически участник изображается “человечком”.

Прецедент (use case) - описание множества последовательных событий (включая варианты), выполняемых системой, которые приводят к наблюдаемому участником результату. Прецедент представляет поведение сущности, описывая взаимодействие между участниками и системой. Прецедент не показывает, “как” достигается некоторый результат, а только “что” именно выполняется. Прецеденты обозначаются очень простым образом - в виде эллипса, внутри которого указано его название.

Основные элементы диаграммы вариантов использования

На диаграмме вариантов использования можно отобразить следующие элементы нотации UML, доступные в панели элементов:

1.  Участник (Actor)
2.  Вариант (Use case)
3.  Граница
4.  **Ненаправленная ассоциация**
5.  **Направленная ассоциация**
6.  **Обобщение**
7.  **зависимость (включение и расширение)**
8.  комментарий
9.  Коннектор комментария

Между субъектами и прецедентами - основными компонентами диаграммы прецедентов - могут существовать различные отношения, которые описывают взаимодействие экземпляров одних субъектов и прецедентов с экземплярами других субъектов и прецедентов. В языке UML имеется несколько стандартных видов отношений между субъектами и прецедентами:

Отношение ассоциации (association) - **определяет наличие канала связи между экземплярами субъекта и прецедента (или между экземплярами двумх субъектов).** Обозначается сплошной линией, возможно наличие стрелки и указание мощности связи.

Отношение расширения (extend) - **определяет взаимосвязь экземпляров отдельного прецедента с более общим прецедентом, свойства которого определяются на основе**

способа совместного объединения данных экземпляров. Обозначается пунктирной линией со стрелкой, направленной от того прецедента, который является расширением для исходного прецедента, и помечается ключевым словом "extend" ("расширяет").

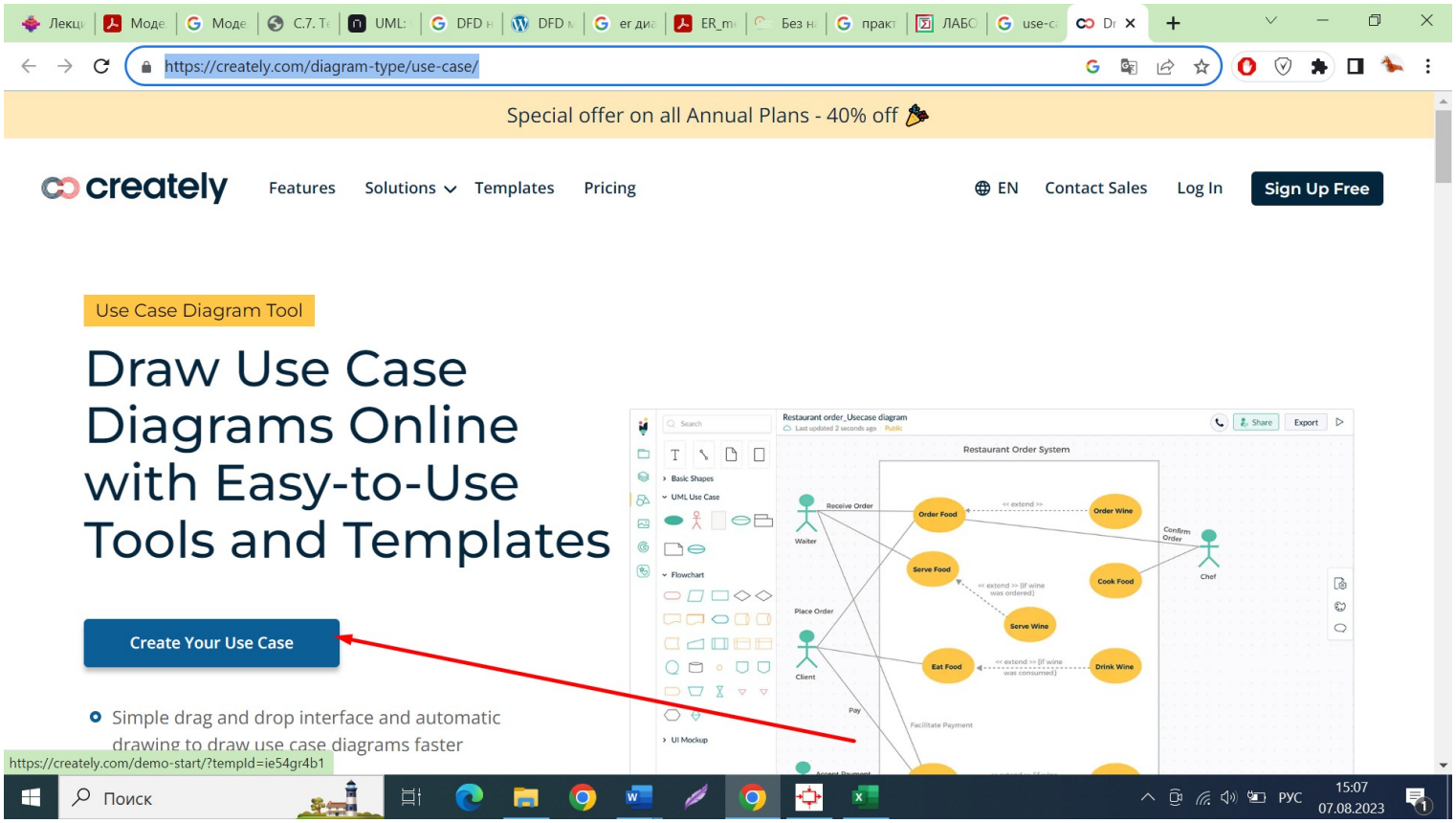
Отношение включения (include) - **указывает, что некоторое заданное поведение для одного прецедента включает в качестве составного компонента поведение другого прецедента.** Данное отношение является направленным бинарным отношением в том смысле, что пара экземпляров прецедентов всегда упорядочена в отношении включения. Обозначается пунктирной линией со стрелкой, направленной от базового прецедента к включаемому, и помечается ключевым словом "include" ("включает").

Отношение обобщения (generalization) - **служит для указания того факта, что некоторый прецедент А может быть обобщен до прецедента В. В этом случае прецедент А будет являться специализацией прецедента В.** При этом В называется предком или родителем по отношению к А, а прецедент А - потомком по отношению к прецеденту В. Следует подчеркнуть, что потомок наследует все свойства и поведение своего родителя, а также может быть дополнен новыми свойствами и особенностями поведения. Графически данное отношение обозначается сплошной линией со стрелкой в форме незакрашенного треугольника, которая указывает на родительский прецедент.

ХОД РАБОТЫ

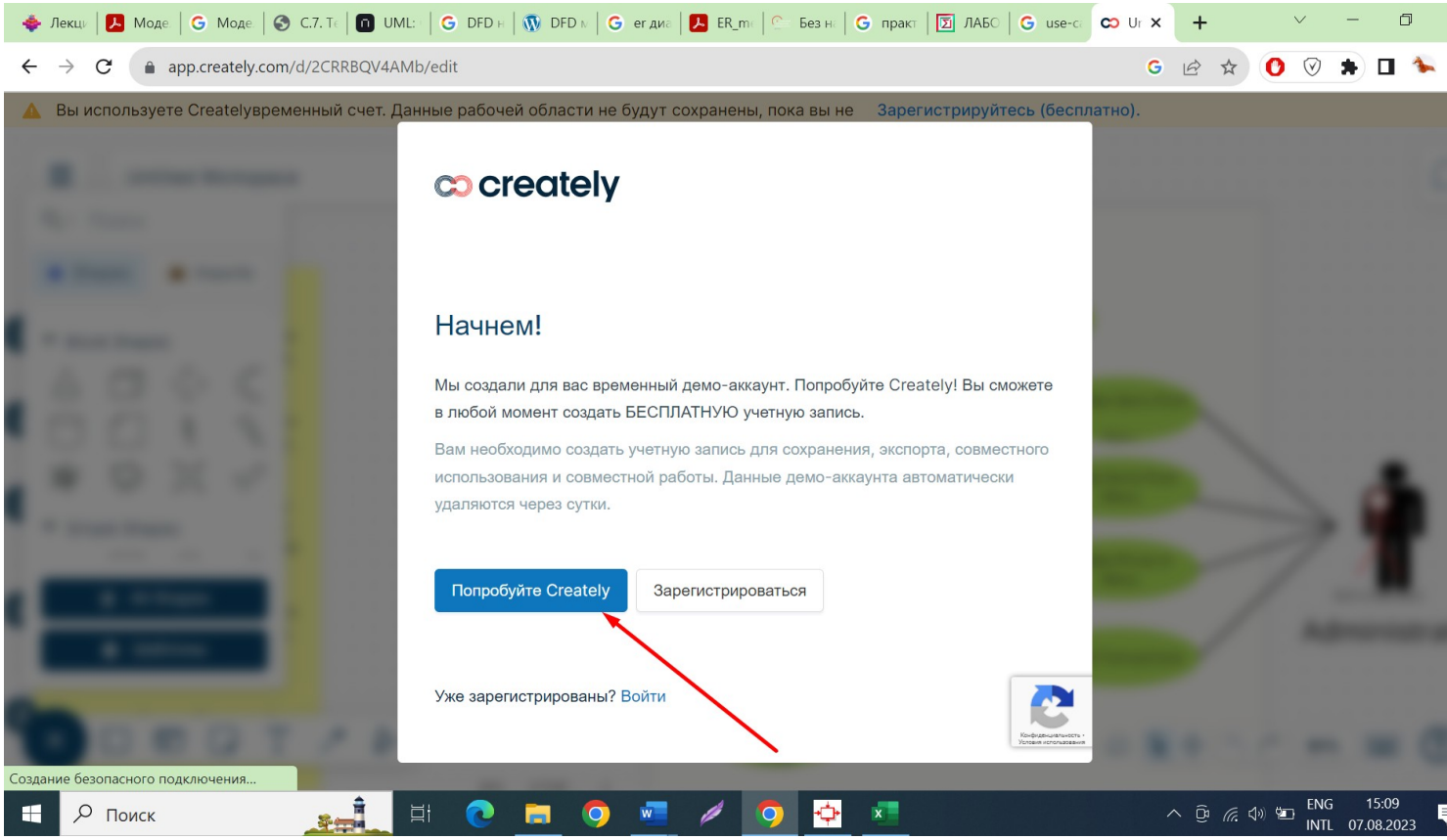
Перейдите по ссылке https://creately.com/diagram-type/use-case/

ЧТО БЫ НЕ РЕГИСТРИРОВАТЬСЯ, и воспользоваться демо версией Create Your Use Case.

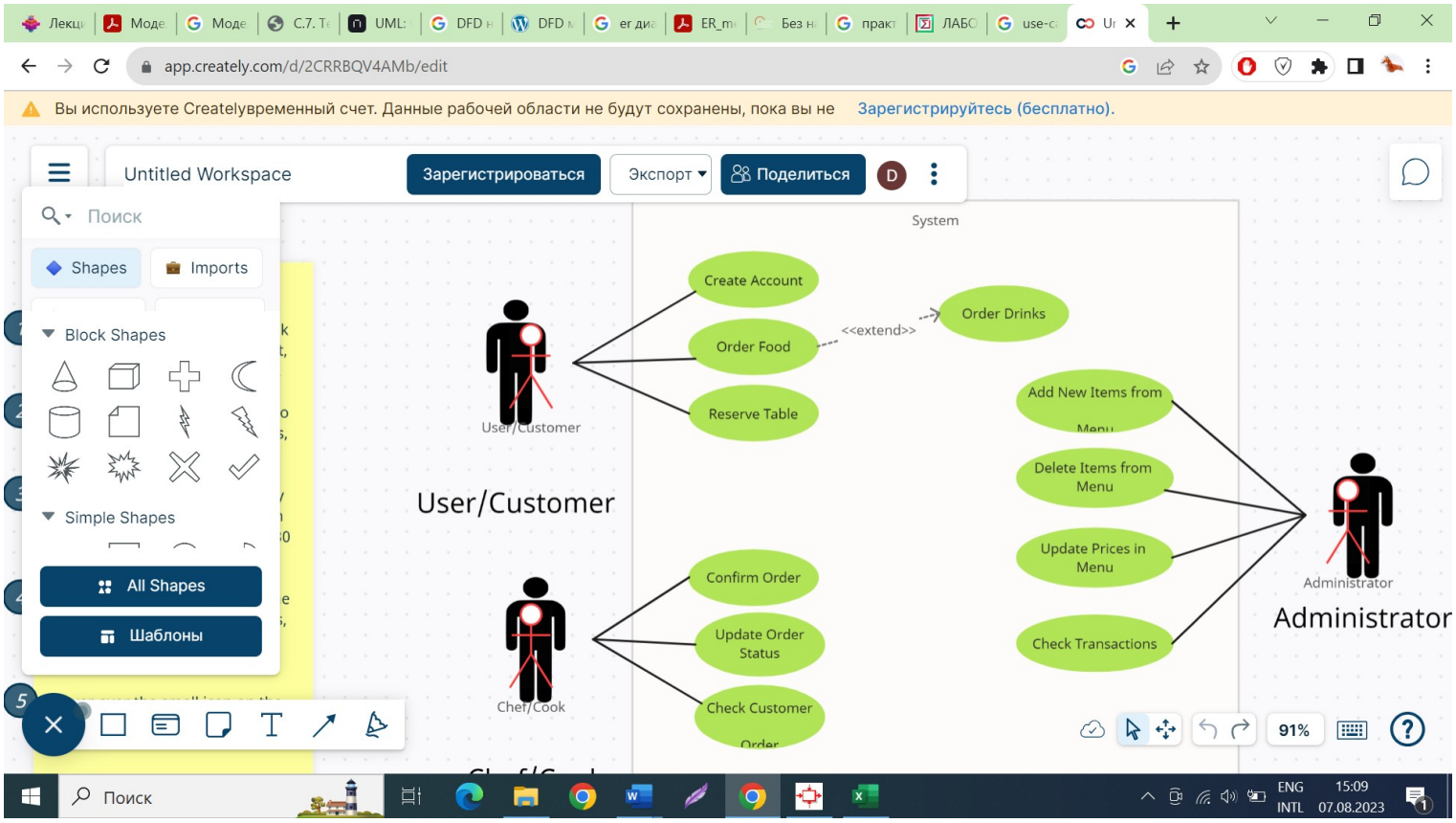


Если же вы решили воспользоваться полной версией, зарегистрируйтесь с помощью своего аккаунта гугл.

После этого перед вами появится окно с таким сообщением, нажмите на попробовать



После этого вам будет доступна рабочая зона и панель инструментов

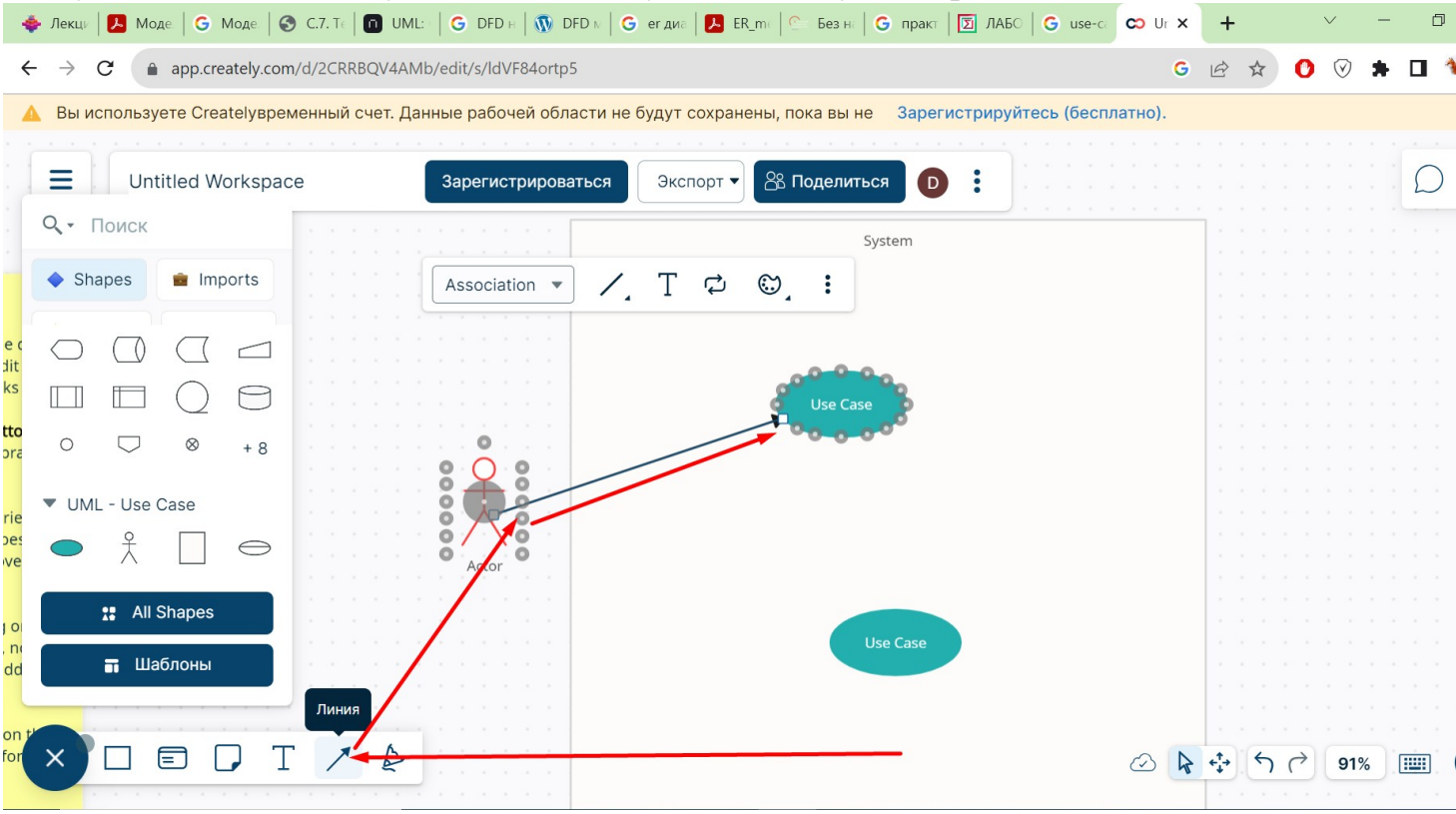


Для того, чтобы начать создавать свою работу , вы можете либо удалить готовые элементы , либо сдвинуть область для создания своей диаграммы.

Основные правила работы в данном ПО:

1. чтобы воспользоваться любым элементов из панели инструментов, зажмите данный элемент и перетяните его на рабочую область.

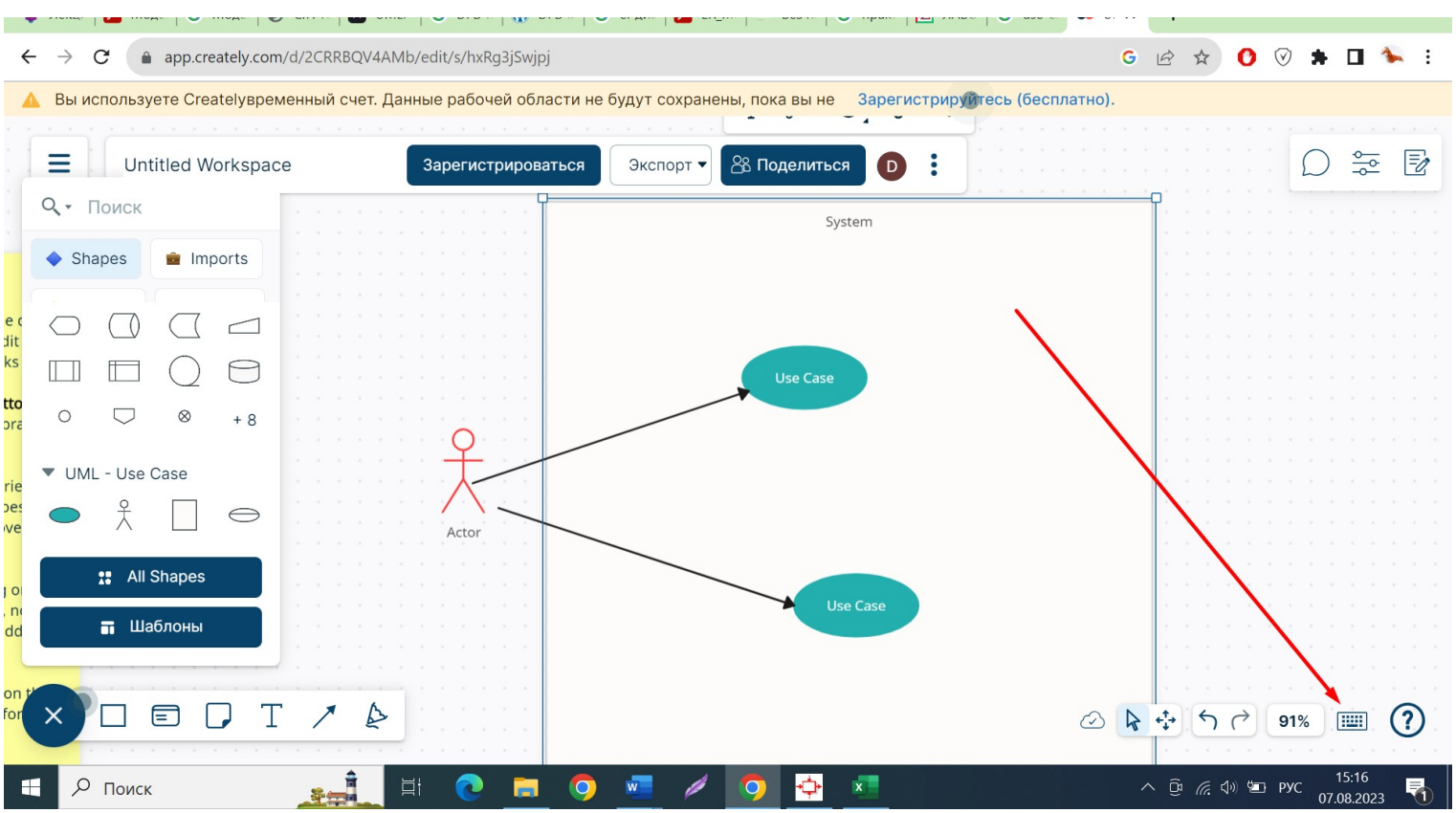
2. Прокрутите меню слева до раздела с элементами use-case
3. Чтобы добавить стрелку, создайте два нужных элемента и на панели инструментов снизу нажмите линия и укажите место куда вам ее нужно провести



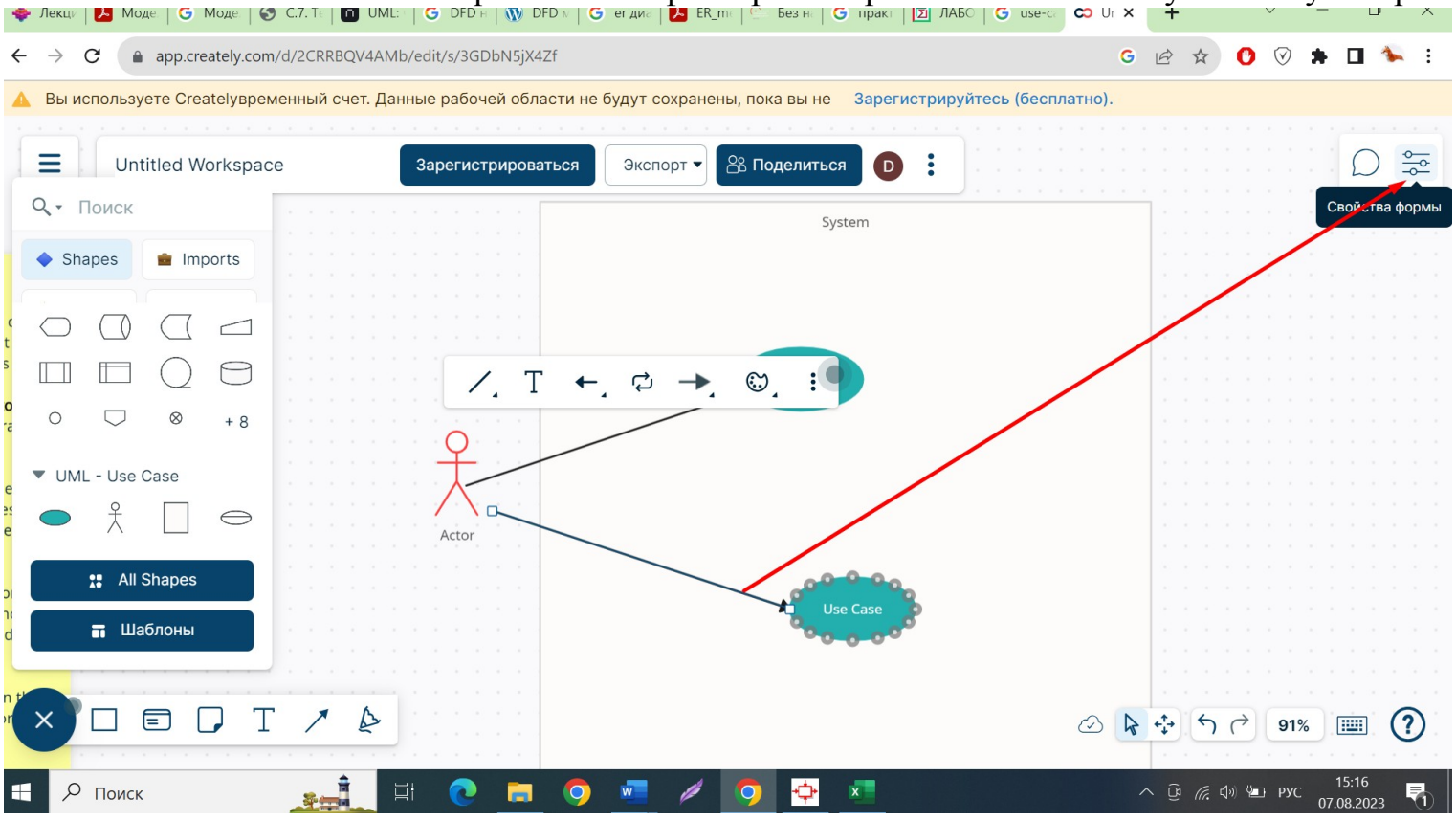
4. Для того ,чтобы изменить тип стрелки ,нажмите на саму стрелку и у вас появится меню редактирования стрелки

- Первое свойство задает гибкость линии
- Втрое свойство - это текст, который вы указываете над стрелкой
- Третье свойство типы стрелок
- Четвёртое свойство - это возможность изменять направление стрелки
- Пятое свойство дает возможность изменить одну из частей наконечника стрелки
- Шестое свойство изменение цвета стрелки

5. Справка по всем горячим клавишам и свойствам

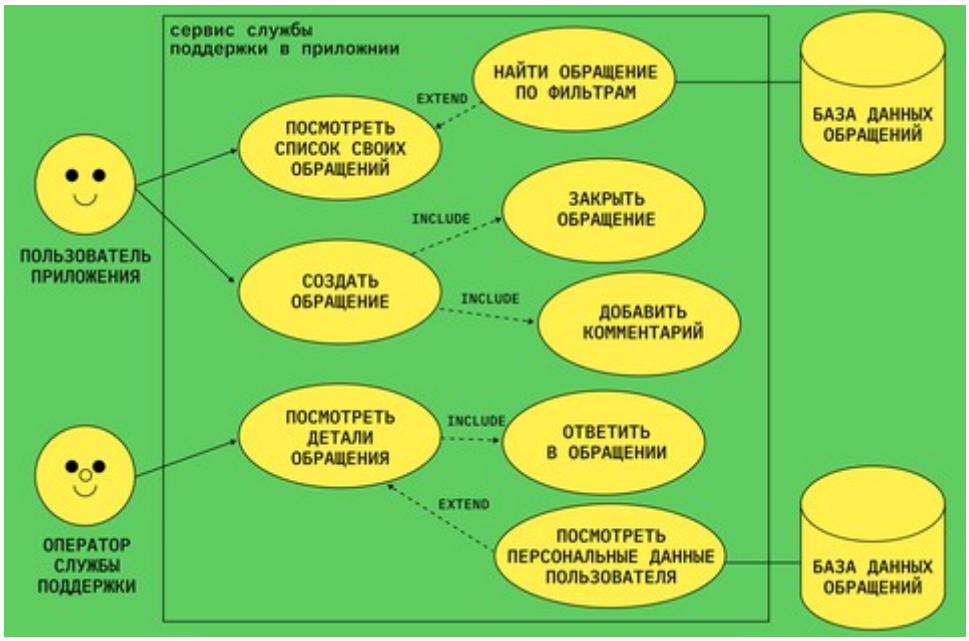


6. Для того, чтобы создать стрелку пунктирную, для этого вам нужно : нажать на объект и в появившемся окне выбрать значок фильтр и выбрать стиль линии и указать пунктир.



Задание 1:

Постройте схему, как показано на картинке ниже с помощью онлайн инструмента, не забудьте изменить значок ПОЛЬЗОВАТЕЛЬ ПРИЛОЖЕНИЯ и ОПЕРАТОР СЛУЖБЫ ПОДДЕРЖКИ на значок актера.



Задание 2:

Добавьте до созданной вами схемы все варианты видов связей, изученных с вами, генерируя действия и устанавливая между ними определенный тип связи.

Контрольные вопросы:

1. Какие типы связей вы знаете?
2. Чем отличается включение от расширения?
3. Как изменить тип связи на диаграмме?
4. Каким типом связи определяется зависимый актер?
5. Дайте определение use-case диаграмме, для чего она нужна?

Содержание отчета:

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

Выводы

6.

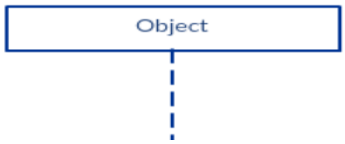
Практическая 3-4

Тема: разработка диаграммы последовательности

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

Принцип построения: Схема последовательности построена таким образом, что она представляет собой временную шкалу, которая начинается сверху и постепенно опускается, чтобы отметить последовательность взаимодействий. Каждый объект имеет колонку, а сообщения, которыми обмениваются между собой, представлены стрелками.

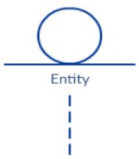
Части диаграммы последовательности



1.Нотация линии жизни обьект последовательность состоит из нескольких таких обозначений линии жизнеобеспечения, которые должны быть расположены горизонтально в верхней части диаграммы.

Никакие две нотации страховочной линии не должны перекрывать друг друга. Они представляют собой различные объекты или части, которые взаимодействуют друг с другом в системе во время последовательности.

2. линия жизни Actor – экземпляр участника процесса (роль на диаграмме прецедентов)



2.линия жизни Entity – Класс-сущность - обычно применяется для обозначения классов, которые хранят некую информацию о бизнес-объектах (соответствует таблице или элементу БД)

3. линия жизни Boundary – Класс-Разграничитель - используется для классов, отделяющих внутреннюю структуру системы от внешней среды (экранная форма, пользовательский интерфейс, устройство ввода-вывода). Объект большей предназначен для вызова методов класса, с которым он связан. Объект boundary показывает именно экранную форму, которая принимает и передает данные обработчику

4. линия жизни Control – Класс-контроллер - активный элемент, который используются для выполнения некоторых операций над объектами (программный компонент, модуль, обработчик)

В UML различают следующие виды сообщений:

- **синхронное сообщение** (**synchCall**) - соответствует синхронному вызову операции и подразумевает ожидание ответа от объекта получателя. Пока ответ не поступит, никаких действий в Системе не производится.
- **асинхронное сообщение** (**asynchCall**) - которое соответствует асинхронному вызову операции и подразумевает, что объект может продолжать работу, не ожидая ответа.
- **ответное сообщение** (**reply**) – ответное сообщение от вызванного метода. Данный вид сообщения показывается на диаграмме по мере необходимости или, когда возвращаемые им данные несут смысловую нагрузку.
- **потерянное сообщение** (**lost**) – сообщение, не имеющее адресата сообщения, т.е. для него существует событие передачи и отсутствует событие приема

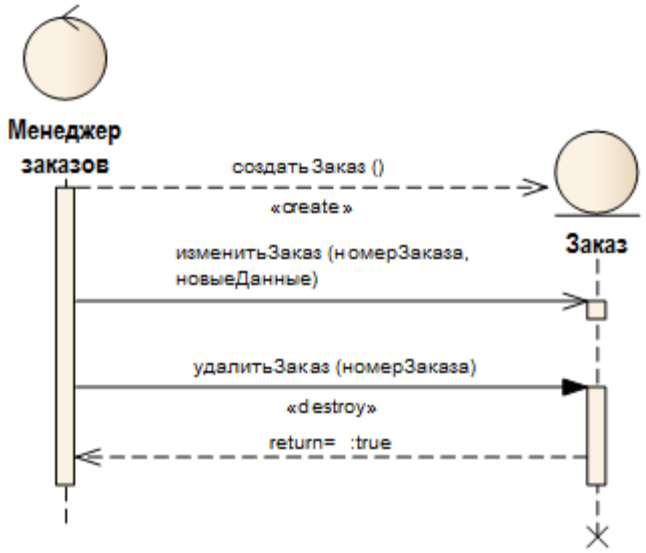
- **найденное сообщение** (**found**) – сообщение, не имеющее инициатора сообщения, т.е. для него существует событие приема и отсутствует событие передачи

Для сообщений также доступен ряд предопределенных стереотипов. Наиболее часто используемые стереотипы это **create** и **destroy.**

Сообщение со стереотипом **create**, вызывает в классе метод, которые создает экземпляр класса. На диаграмме последовательности не обязательно показывать с самого начала все объекты, участвующие во взаимодействии. При использовании сообщения со стереотипом **create**, создаваемый объект отображается на уровне конца сообщения.

Для уничтожения экземпляра класса используется сообщение со стереотипом **destroy**, при этом в конце линии жизни объекта отображаются две перекрещенные линии.

При отображении работы с сообщениями иногда возникает необходимость указать некоторые временные ограничения. Например, длительность передачи сообщения или ожидание ответа от объекта не должно превышать определенный временной интервал.



Если нам нужно создать нового участника(в данном случае новый заказ), мы ведем стрелку к этому элементу, если он не был создан.
После этого всегда должен быть блок активации на этой линии жизни, в данном случае это активатор для изменения заказа, так как он был создан пустым.

Отдельные фрагменты диаграммы взаимодействия можно выделить с помощью фрейма. Фрейм должен содержать метку оператора взаимодействия. UML содержит следующие операнды:

- **Alt -** Несколько альтернативных фрагментов (alternative); выполняется только тот фрагмент, условие которого истинно

НАШ ОПЕРАТОР if-then-else; case; switch

- **Opt -** Необязательный (optional) фрагмент; выполняется, только если условие истинно. Эквивалентно **alt** с одной веткой

if-then, без ветки иначе

- **Par -** Параллельный (parallel); все фрагменты выполняются параллельно

Это наши разграничители на схеме отделает наш if от else

- **loop -** Цикл (loop); фрагмент может выполняться несколько раз, а защита обозначает тело итерации
- **Neg -** Отрицательный (negative) фрагмент; обозначает неверное взаимодействие

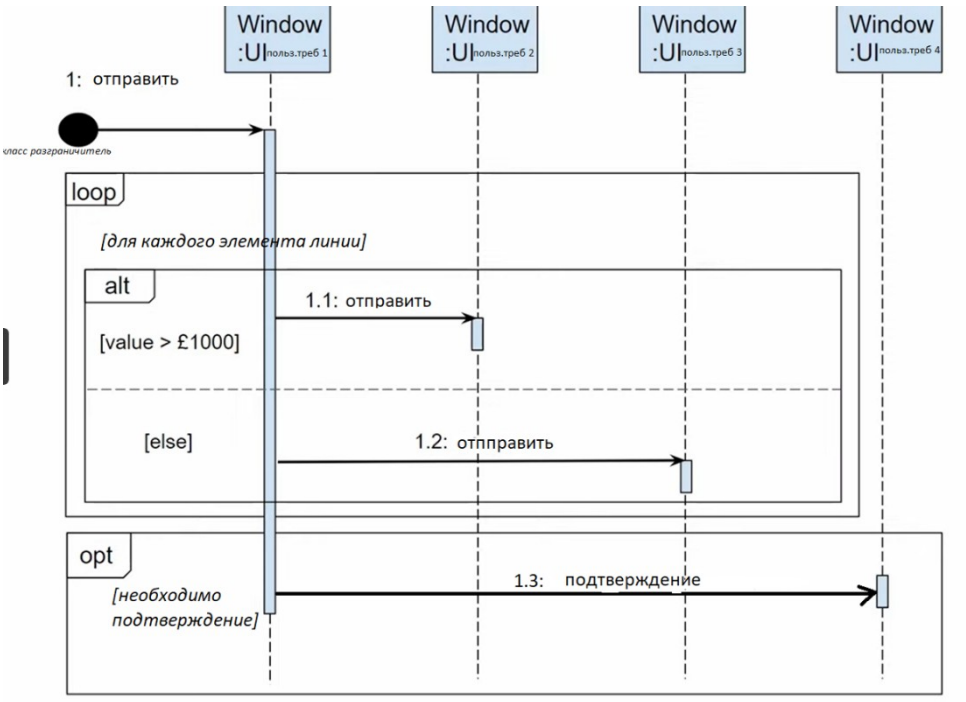
К примеру, блок ждет пароля от пользователя, время ожидания вышло и ему выведется сообщение Время вышло

- **ref -** Ссылка (reference); ссылается на взаимодействие, определенное на другой диаграмме. Фрейм рисуется, чтобы охватить линии жизни, вовлеченные во взаимодействие. Можно определять параметры и возвращать значение

ссылка на другой блок или диаграмму

• **Sd** - (sequence diagram); используется для очерчивания всей диаграммы последовательности, если это необходимо.

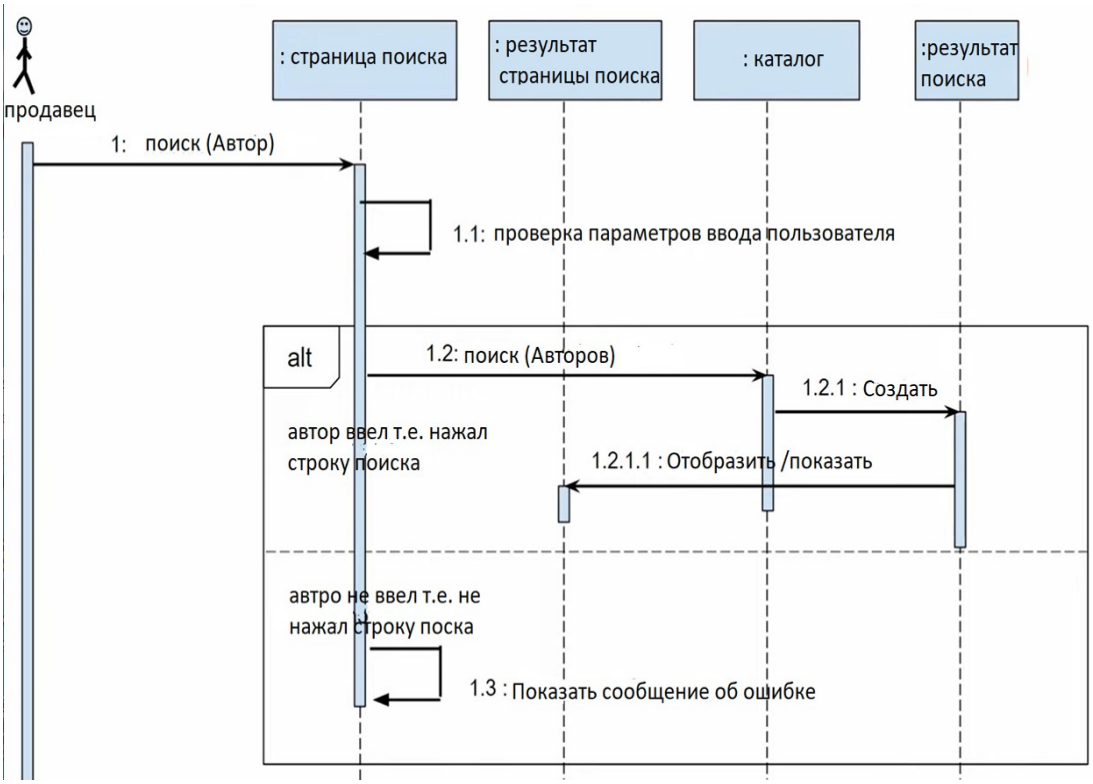
•



- **Как показать цикличность**
- В данном случае используется класс разграничитель(черный кружек)и 4 линии жизни (пользовательские инерфейсы 4 шт). Прямоукольник loop выделил область которая указывет на цикл, а прямоуголник с alt указывает на условие(пока пользователь не достиг просмотра 1000минут -он видит сообщение 1.1, если время становится больше 1000-он увидит сообщение 1.2. Далее Блок блок ALT,он показывет истенное выражение- в данном случае пользователю нравится интерфейс и он нажимает на лайк-это и есть подтверждение.

Цикличность потока взаимодействия может быть представлена на диаграмме последовательности с помощью операнда loop. При использовании оператора цикла можно указать минимальное и максимальное число итераций. Также фрейм должен содержать условие, при наступлении которого взаимодействие повторяется.

есть два сценария:



1. Основной
2. Альтернативный

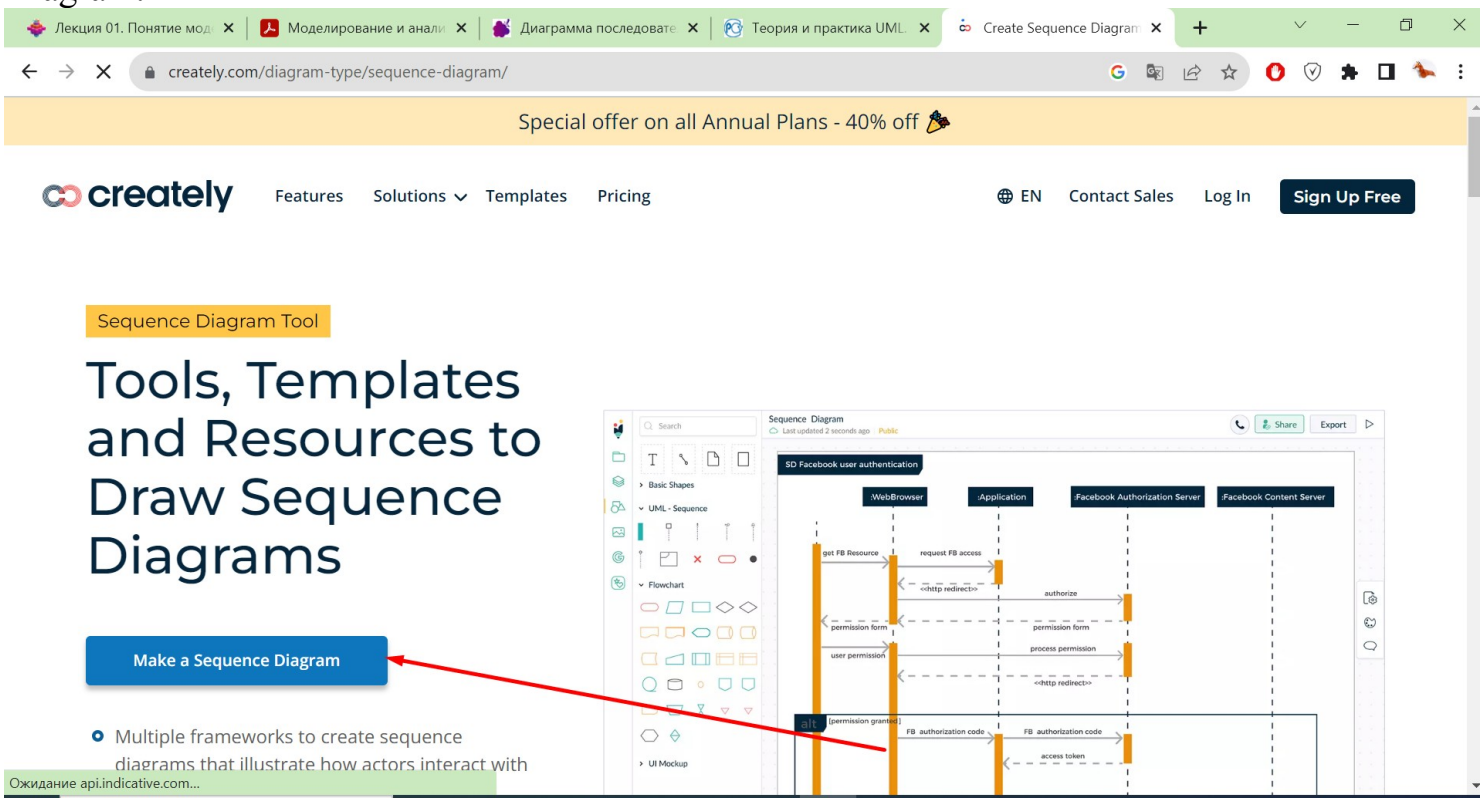
1.Основной: ПРОДАВЕЦ вводит ФИО автора и нажимает кнопку поиска. После чего система проверяет параметры ввода пользователя, если валидация пройдена- система ищет Автора по каталогам, когда поиск окончен - система выдает результат

2. Если покупатель не ввел имя автора, но нажал на кнопку поиска, то система выводит сообщение об ошибке.

ХОД РАБОТЫ:

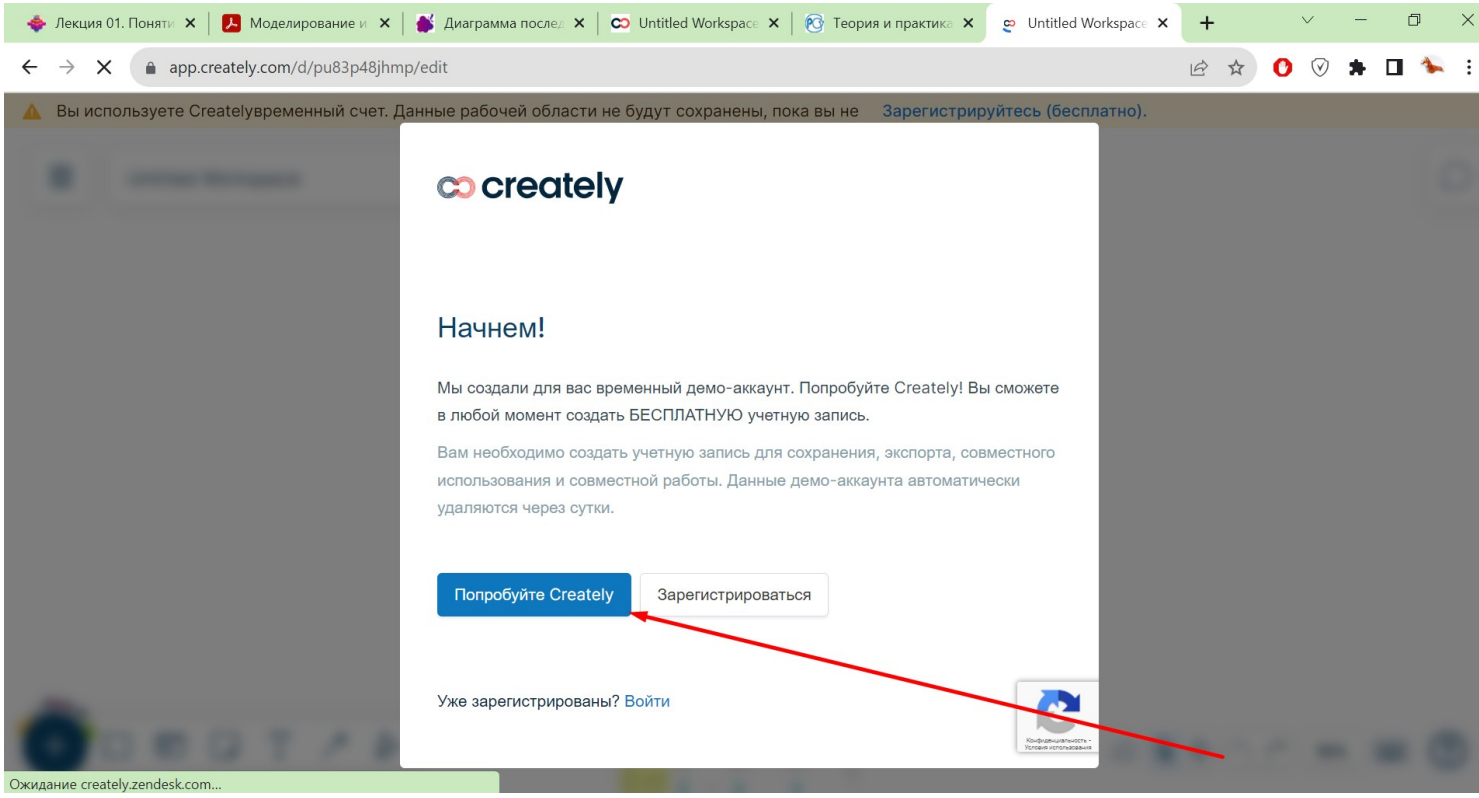
Перейдите по ссылке https://creately.com/diagram-type/sequence-diagram/

ЧТО БЫ НЕ РЕГИСТРИРОВАТЬСЯ, и воспользоваться демо версией Make a Sequence Diagram.

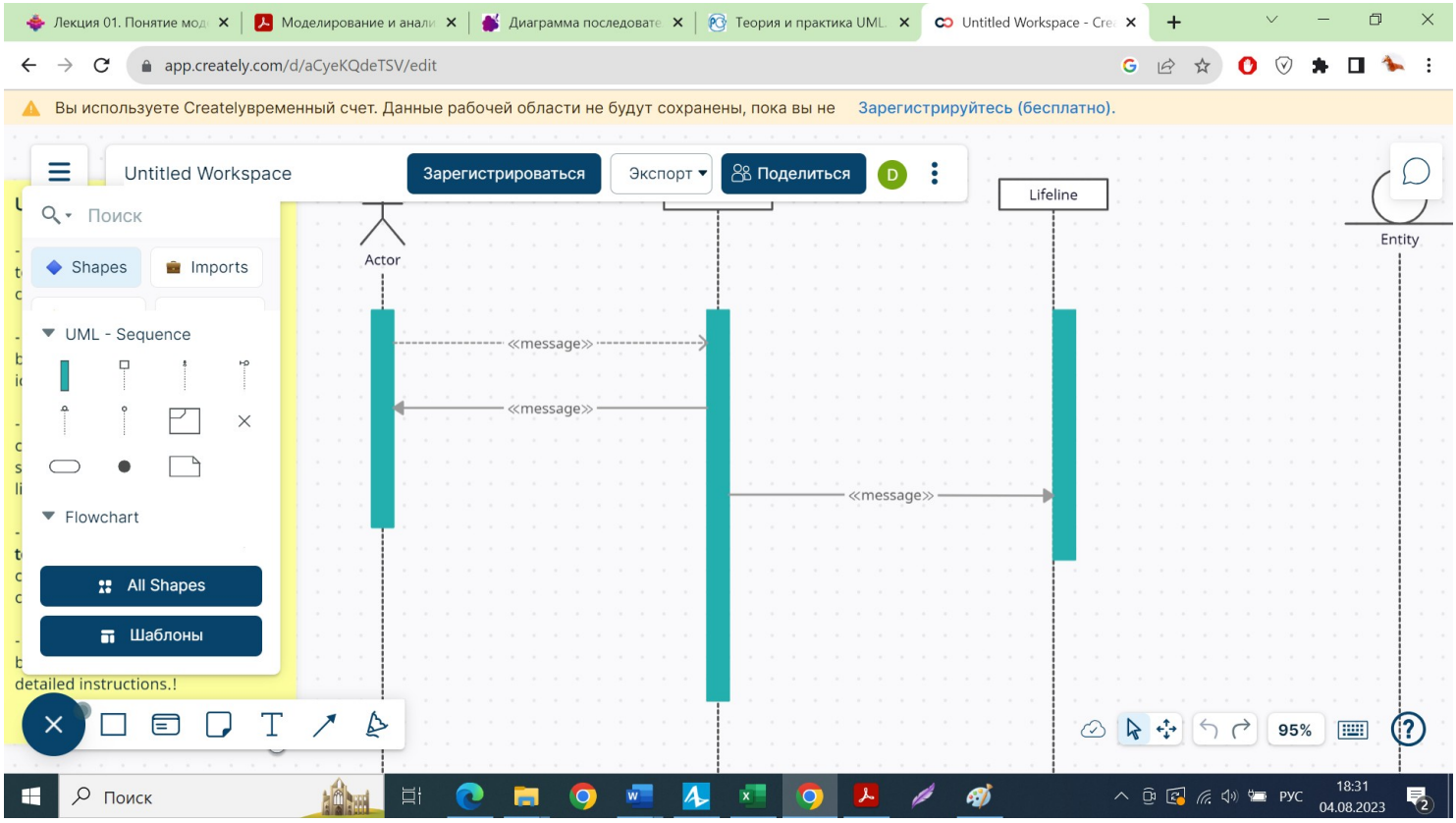


Если же вы решили воспользоваться полной версией, зарегистрируйтесь с помощью своего аккаунта гугл.

После этого перед вами появится окно с таким сообщением, нажмите на попробовать



После этого вам будет доступна рабочая зона и панель инструментов

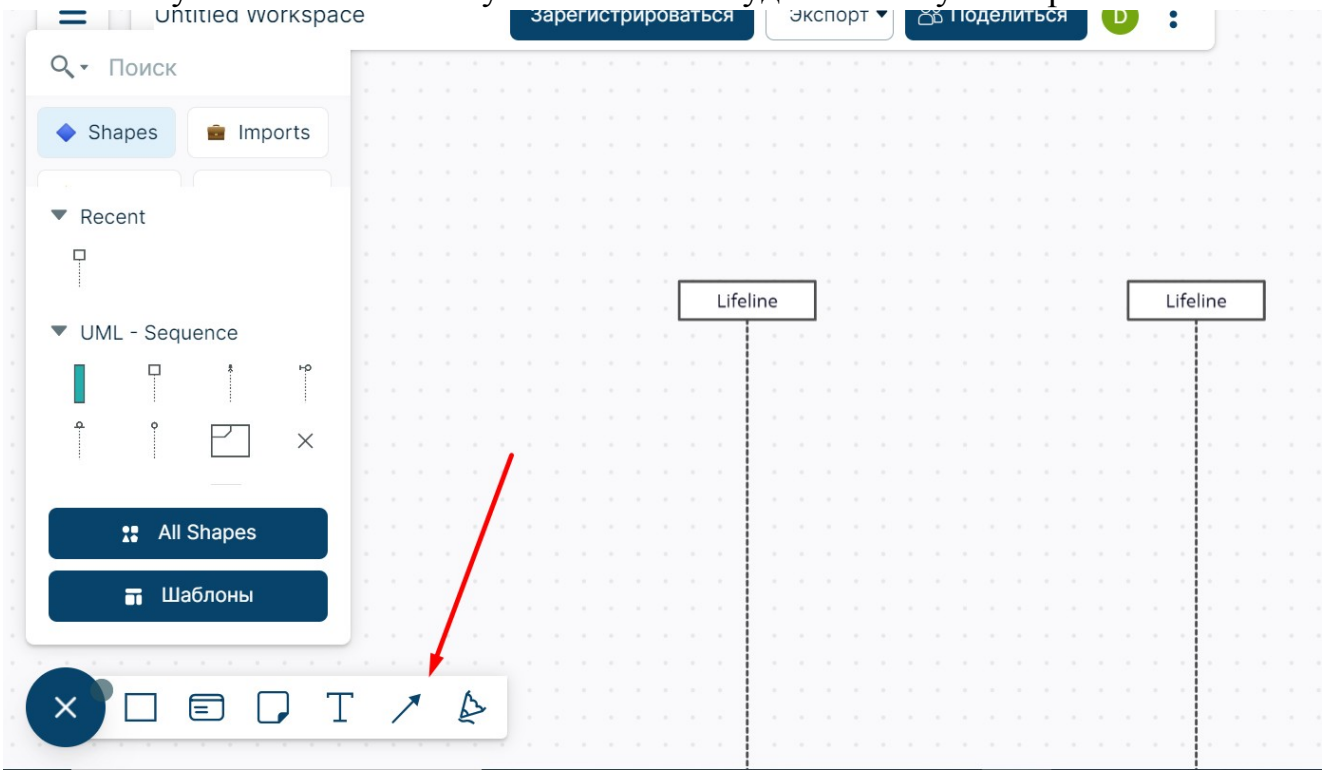


Для того, чтобы начать создавать свою работу , вы можете либо удалить готовые элементы ,либо сдвинуть область для создания своей диаграммы.

Основные правила работы в данном ПО:

7. чтобы воспользоваться любым элементов из панели инструментов, зажмите данный элемент и перетяните его на рабочую область.

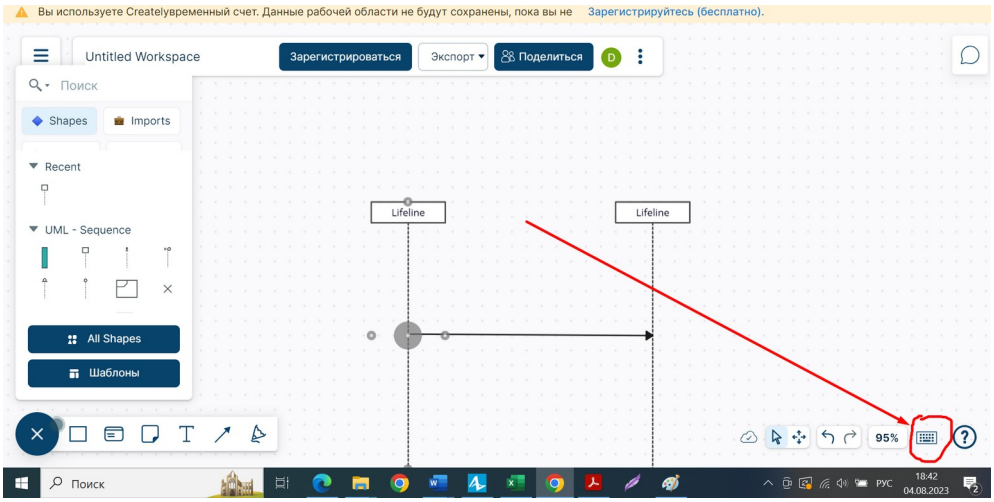
8. Чтобы добавить стрелку , создайте два нужных элемента и на панели инструментов снизу нажмите линия и укажите место куда вам ее нужно провести



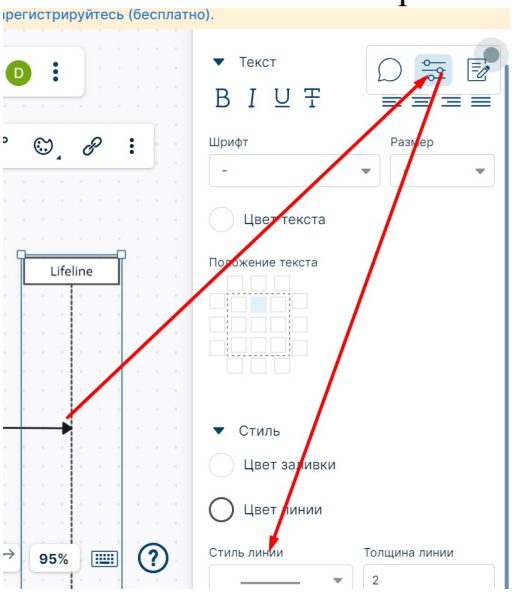
9. Для того ,чтобы изменить тип стрелки ,нажмите на саму стрелку и у вас появится меню редактирования стрелки

- Первое свойство задает гибкость линии
- Втрое свойство - это текст, который вы указываете над стрелкой
- Третье свойство типы стрелок
- Четвёртое свойство - это возможность изменять направление стрелки
- Пятое свойство дает возможность изменить одну из частей наконечника стрелки
- Шестое свойство изменение цвета стрелки

10. Справка по всем горячим клавишам и свойствам

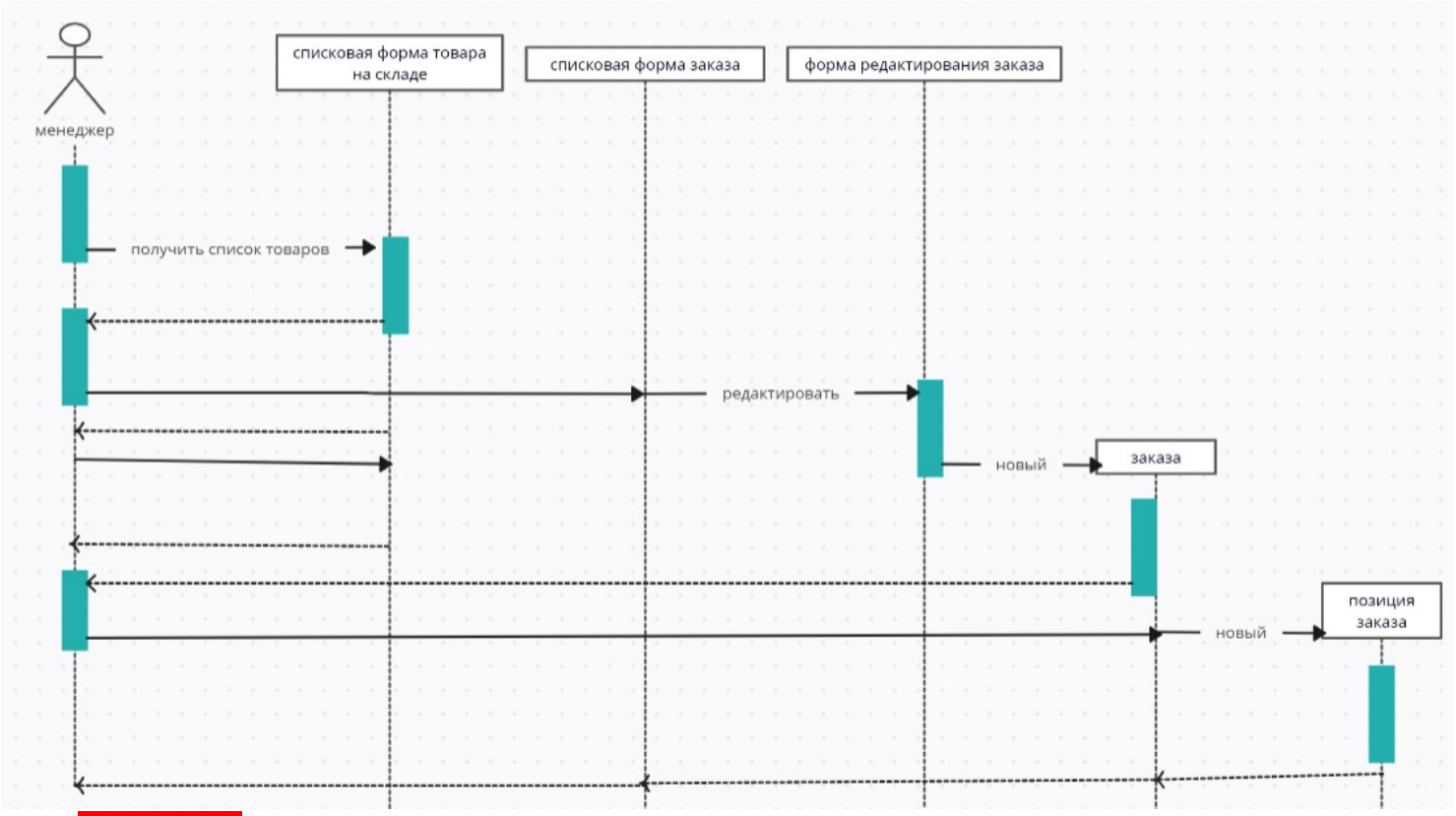


11. Для того, чтобы создать стрелку пунктирную, для этого вам нужно : нажать на объект и в появившемся окне выбрать значок фильтр и выбрать стиль линии и указать пунктир.



Задание1:

1. Создайте идентичную схему ,как показано на рисунке ниже



Задание2:

Выберите предметную область, в которой вы лучше всего разбираетесь (тема не должна совпадать с вашими одногруппниками) далее создайте:

1. Две лини жизни, а третью линию жизни создайте в процессе.

2. Используйте Бар /активация минимум 3 шт
3. Использовать два и более различных вида сообщения;
4. Использовать два любые вида фрейма.

Постройте схему.

Контрольные вопросы:

1. Сколько линий жизни вы знаете, назовите их
2. Как изменить тип стрелки на диаграмме?
3. Как показать цикличность?
4. Назовите 3 вида сообщений?
5. За что отвечает параллельный фрейм?
6. Что показывает стрелка сама в себя на блоке активации?
7. что показывает диаграмма последовательности?

Содержание отчета:

5. Тема, цель практической работы
6. Поэтапное описание выполнения практической работы
7. Скриншоты или результат практической
8. Краткие ответы на контрольные вопросы

Выводы

8.

Практическая 5-7

Тема: разработка DFD модели

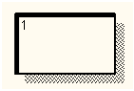
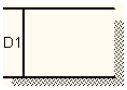
ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

Диаграммы потоков данных показывают, как каждый процесс преобразует свои входные данные в выходные, и выявляют отношения между этими процессами. DFD представляет моделируемую систему как сеть связанных работ.

При построении DFD-схемы бизнес-процесса *нужно помнить, что данная схема показывает потоки материальных и информационных потоков и ни в коем случае не говорит о временной последовательности работ***,** хотя в большинстве случаев временная последовательность работ и совпадает с направлением движения потоков в бизнес-процессе.

Все процессы должны быть связаны либо с другими процессами, либо с другими хранилищами данных. Процессы не существуют сами по себе, а потому результат должен куда-то передаваться;

Основные компоненты нотации:

Описание	**Графическое представление**	
Работа (Activity)	Объект обозначает функции или процессы, которые обрабатывают и изменяют информацию.	
Информационный поток (Precedence)	Объект обозначает информационный поток от объекта-источника к объекту-приемнику.	
Внешняя ссылка (External reference)	Указывают на место, организацию или человека, которые участвуют в процессе обмена информацией с системой, но располагаются за рамками этой диаграммы.	
Хранилище данных (Data store)	Хранилища данных представляют собой собственно данные, к которым осуществляется доступ, эти данные также могут быть созданы или изменены работами. На одной диаграмме может присутствовать несколько копий одного и того же хранилища данных.	

Требования к оформлению функций:

1. Каждая функция должна иметь идентификатор;

2. Названия работы нужно формулировать согласно следующее формуле:

Название работы = Действие + Объект, над которым действие осуществляется

Например, если эта работа связана с действием по продаже продукции, то ее нужно назвать Продажа продукции

3. Название работы должно быть по возможности кратким (не более 50 символов) и состоять из 2-3 слов. В сложных случаях также рекомендуется для каждого краткого названия работы сделать ее подробное описание, которое поместить в глоссарий.

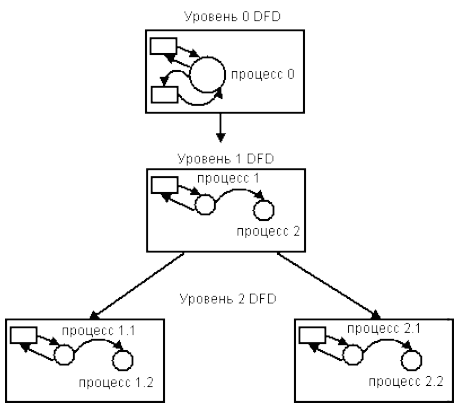
Требования к оформлению потока данных:

1. Название потока нужно формулировать согласно следующей формуле:

Название потока = Объект, представляющий поток + Статус объекта

Если речь идет о продукции, которую отгрузили клиенту, то поток можно назвать <Продукция, отгруженная> или <Продукция, отгруженная клиенту>. В данном случае <Продукция> это объект, представляющий поток, а <отгруженная клиенту> — статус объекта.

Название должно быть по возможности кратким и состоять из 2-3 слов.



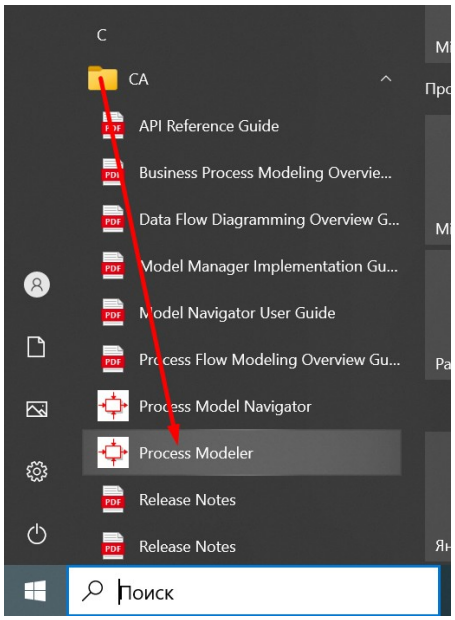
При детализации должны выполняться следующие правила:

- **правило балансировки** — *при детализации процесса дочерняя диаграмма в качестве внешних источников/приемников данных может* иметь только те компоненты данные с которыми имеет информационную связь соответствующий процесс на родительской диаграмме;

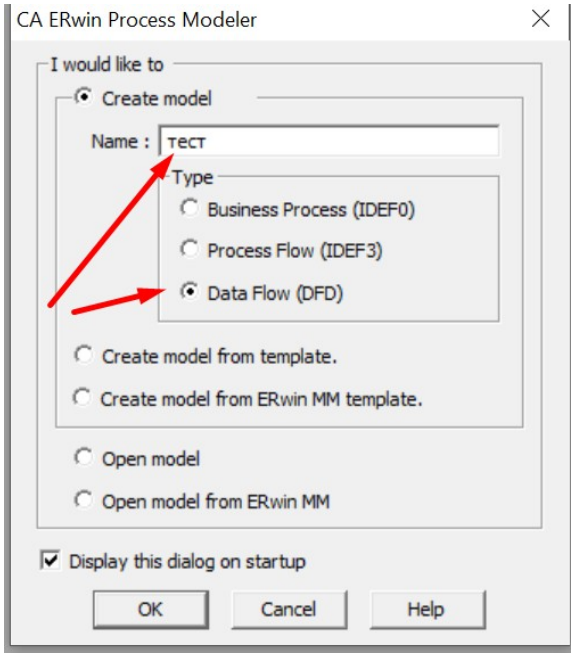
- **правило нумерации** — при детализации процессов должна поддерживаться их иерархическая нумерация.
- **правило семи** — для того, чтобы диаграмма легко читалась, количество функций на диаграмме не должно быть больше семи.

ХОД РАБОТЫ:

1.Запустите Erwin

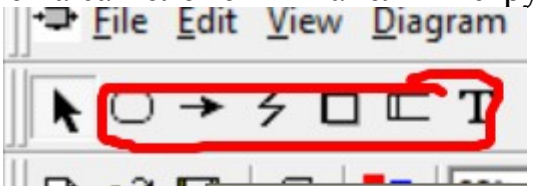


2. создайте проект DFD с темой поликлиника

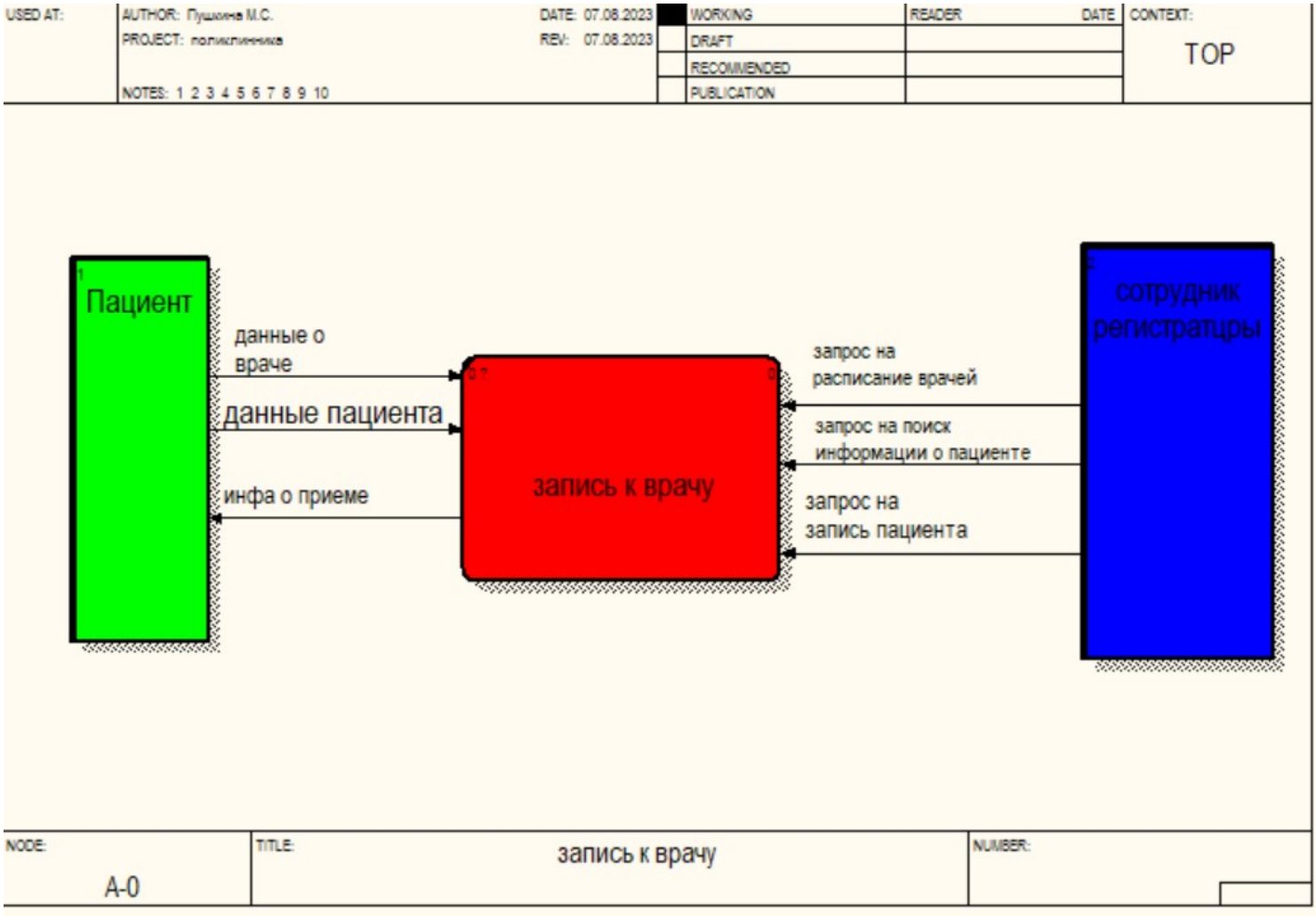


3.укажите свое ФИО

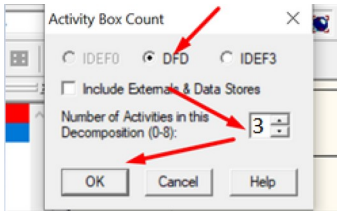
4. в запущенном проекте укажите кириллицу, самой диаграмме и его структуре. Во вкладке Model => Default Font и кликнув два раза по основному процессу в меню Font.

5.для выбора любого из элементов нажмите на сам элемент в панели инструментов и нажмите на поле куда вы его хотите поставить 

6. задание 1. после чего постройте схему как показано на рисунке.



7. перейдите к детализации процессов, нажав на основной процесс и после чего выбрать стрелку в низ > выбрать DFD , 3 блока, и нажмите ок.



8. задание 2. далее постройте схему по таким параметрам:

8.1. создайте 3 процесса:

«Поиск врача и возможного времени», «Запись в журнал приёма», «Поиск карточки пациента»

8.2. создайте 3 хранилища данных:

«Расписание приёма врачей», «Карточки регистраторы», «Журнал приема»

8.3. создайте 2 внешние ссылки:

«Сотрудник регистратуры», «пациент»

8.4. соедините все потоками данных, присоединив уже от наследованные данные от первой схемы и добавив новые потоки. Результат сохраните и вставьте в отчет.

9. задание 3. Выберите предметную область, в которой вы лучше всего разбираетесь. Главное проверти, чтобы темы не совпадали с вашими одногруппниками. **Постройте модель из двух уровней детализации:**

1й уровень: должен состоять из работы и внешних ссылок и потоков, количество потоков генерируйте по надобности.

2й уровень: должен содержать 3-4 работы, 2 и более внешних источников, 2 и более хранилищ и от 6 до 10 потоков.

Контрольные вопросы:

9. Что такое поток?
10. что такое внешняя ссылка, что она показывает?
11. Могут ли процессы существовать сами по себе?
12. Для чего используют DFD?
13. Сколько функций рекомендовано использовать на одном уровне? почему?

Содержание отчета:

9. Тема, цель практической работы
10. Поэтапное описание выполнения практической работы
11. Скриншоты или результат практической
12. Краткие ответы на контрольные вопросы

Выводы

14.

Практическая 8

Тема: разработка контекстной диаграммы верхнего уровня в нотации IDEF0 в среде разработки ER win

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Нотация IDEF0

IDEF0 (Integration Definition for Function Modeling) – нотация описания бизнес-процессов. Основана на методологии SADT.Нотация может быть использована для моделирования широкого круга автоматизированных и неавтоматизированных систем.

Идея IDEF0 лежит в том, что бизнес-процесс отображается в виде прямоугольника, в которой входят и выходят стрелки.

Для IDEF0 имеет значение сторона процесса и связанная с ней стрелка:

1. слева входящая стрелка – **вход бизнес-процесса** – информация (документ), который будет преобразован в ходе выполнения процесса;
2. справа исходящая стрелка – **выход бизнес-процесса** – преобразованная информация (документ);
3. сверху входящая стрелка – **управление бизнес-процесса** – информация или документ, который определяет как должен выполняться бизнес-процесс, как должно происходить преобразование входа в выход;
4. снизу входящая стрелка – **механизм бизнес-процесса** – то, что преобразовывает вход в выход: сотрудники или техника. Считается, что за один цикл процесса не происходит изменения механизма.

Выход одного бизнес-процесса является входом/управлением/механизмом другого бизнес-процесса. На диаграмме процессы принято располагать по диагонали с верхнего левого угла в нижний правый. Количество процессов не более 6-8.

1. *В левом верхнем углу всегда – главный элемент.*
2. *Все элементы должны иметь входящие и исходящие стрелки, так как для выполнения необходимо что-то получить на входе (заказ, поставленную задачу), а после обработки на выходе необходимо передать готовый продукт. Входящие стрелки всегда слева, исходящие – справа.*
3. *Сверху – управляющие элементы, снизу – механизмы, необходимые для выполнения процесса.*
4. *Если на одном листе (экране) располагается несколько блоков, каждый последующий располагается справа и ниже предыдущего.*
5. *Необходимо стремиться создавать схемы таким образом, чтобы пересечение стрелок было сведено к необходимому минимуму.*

ХОД РАБОТЫ

1. зададим имя заготовке контекстной диаграммы, выбрав свойства модели (меню Model>Model Properties…), свойства диаграммы – двойной клик мыши на свободном поле диаграммы, или пункт меню Diagram Properties…, или контекстное меню на свободном поле диаграммы.
2. зададим свойства модели. На вкладке General зададим информацию о модели. Временные рамки Time Frame примем AS-IS. Это означает, что рассматриваются существующие процессы.
3. на вкладке Purpose (Цель) внесем цель моделирования Purpose: "Моделировать текущие бизнес–процессы библиотеки" и точку зрения, с которой строится модель Viewpoint: "Директор",не обязательно.
4. на вкладке Definition (Определение) задаем определение модели Definition: "Учебная модель, описывающая деятельность компании" и границы (рамки) модели Scope: "Общее управление бизнесом компании".
5. выделим функциональный блок на контекстной странице и начинаем его форматирование, задаем кириллицу

6. чтобы указать входной параметр: Выбираем стрелку направляем к объекту
7. чтобы задать имя 2 раза кликаем на стрелку и вводим имя
8. закольцеванность на начало стрелки и выбираем External reference

более понятный способ

2.1 **Если вы выбрали ERwin Process Modeler**

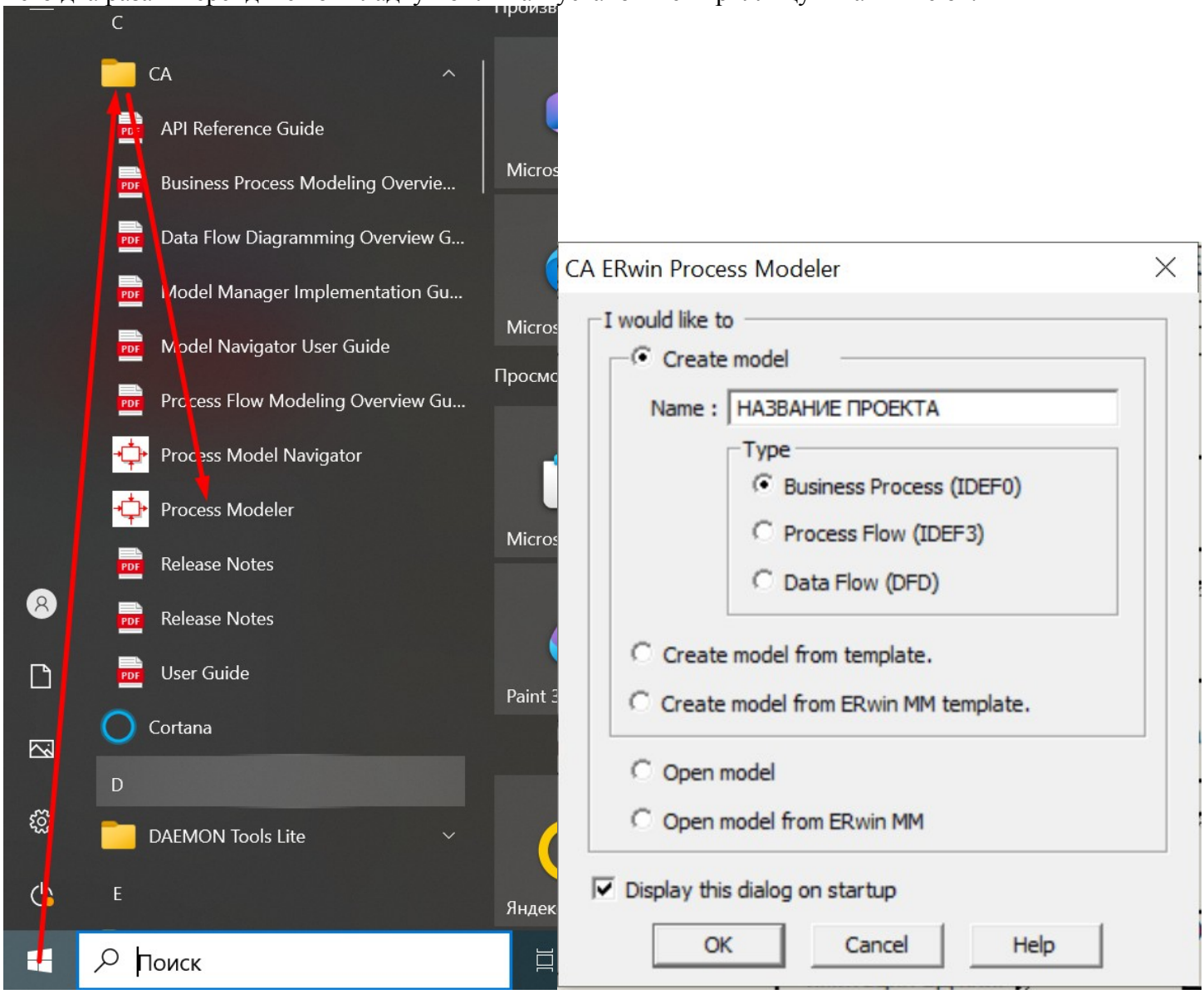
2.1.1 после установки ERwin Process Modeler, активируйте его с помощью генерации ключей

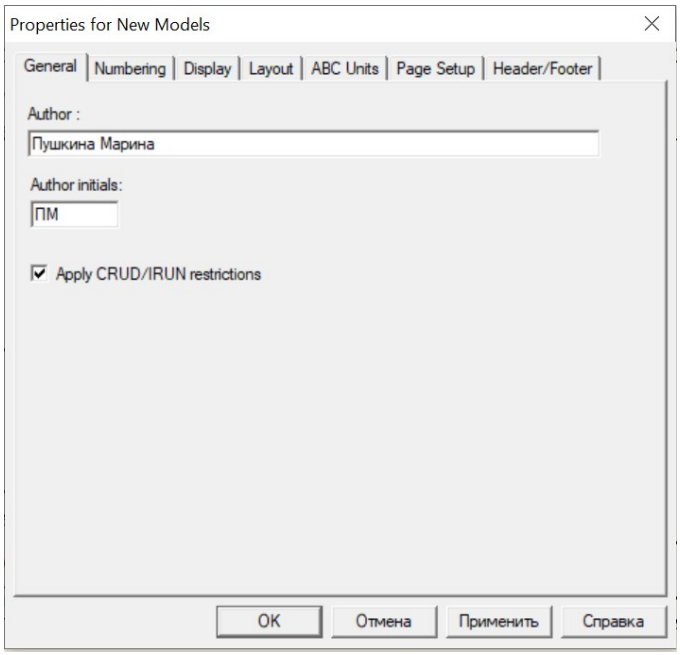
2.1.2 Внимание, Erwin имеет только английскую версию. Для того чтобы IDEF0 представлялась на русском языке выполните следующие действия:

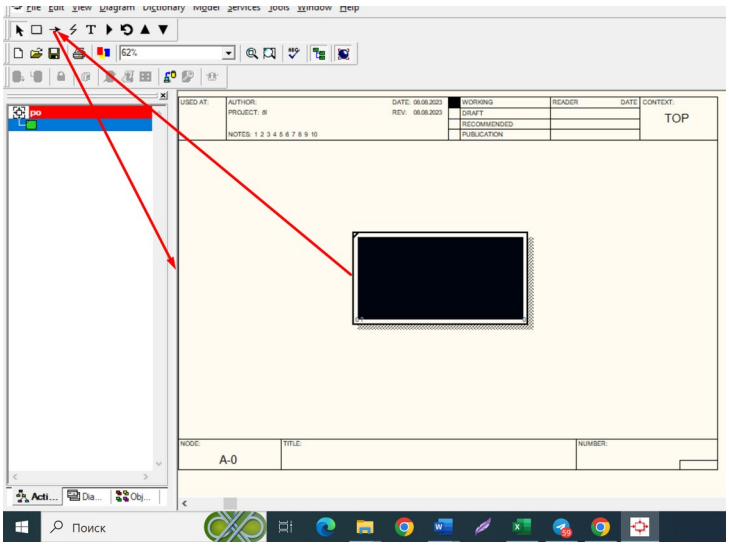
2.1.2.1 выберите тип модели IDEF0 и укажите свое название проекта и нажмите ОК➔укажите фамилию и инициалы

2.1.2.2 ДЛЯ того, чтобы включить кириллицу выберите: -model==default fronts== включаем для каждого элемента в списке кириллицу!! Или в левом углу галочку и выберите кириллицу!

2.1.2.3 Чтобы установить кириллицу на функциональный блок ,нажмите на него два раза и перейдите во вкладку Font и там установите кириллицу и нажмите ок.

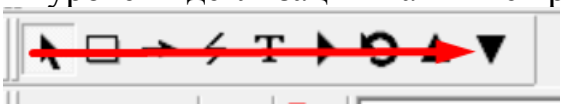


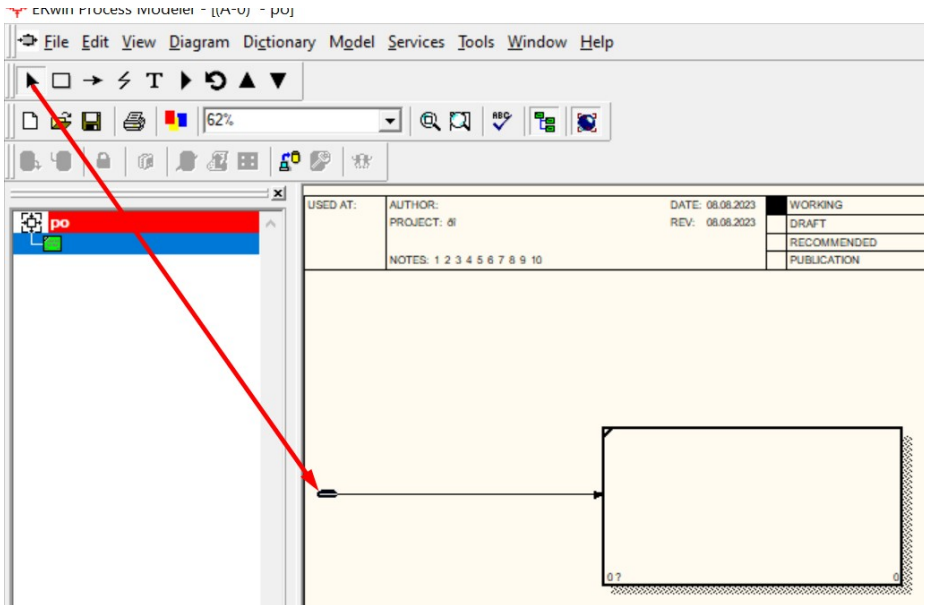




Для того., чтобы добавить запрос нажмите на функциональный блок,как показано выше на рисунке и выберите стрелку. После чего подведите к краю модели после фиксации подведите на вход функционального блока. Анологично добавлять запросы на другие стороны функционального блока.

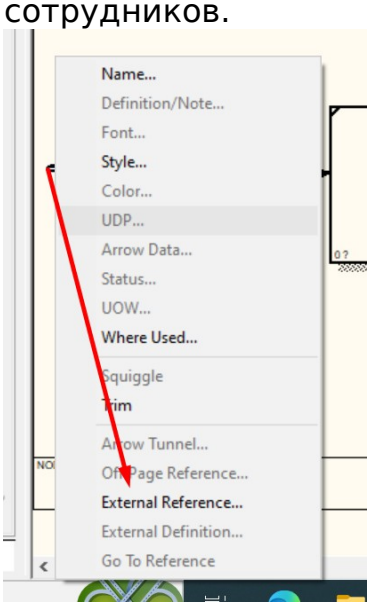
Для переходна на слудующий уровень детлизации нажмите треугольник вниз на панеле инструментов и выберите IDEF3



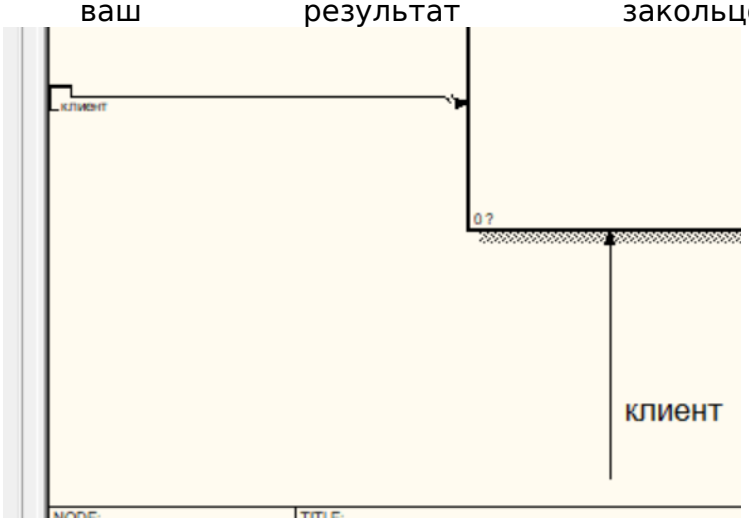


Для установки закольцованности, нажмит активную стрелку, подведите на начало входящего/авыходящего запроса .

После чего нажмите правой кнопкой мыши и выберите External reference, после чего в появившемся окне задайте того, от кого будет исходить данный запрос из ваших созданных сотрудников.



ваш результат закольцованносте должен выглядеть



После чего начинайте создавать ваш проект!
задание 1 выберите предметную область, чтобы не повторялась с вашими одногруппниками, постройте диаграмму 1-го уровня , на диаграмме должно быть:

1. **5 входных запросов;**
2. **7 и более выходных запросов**
3. **3 документа**
4. **Для входных и выходных запросов установите закольцованность**
5. **1 сторонний сервис и 4 сотрудника**

№	Варианты возможных условий
1.	Паспортный стол
2.	Изучение языков по карточкам
3.	Цветочный магазин
4.	Доставка продукции с интернет магазина
5.	Пекарня хлебобулочных изделий
6.	Завод мягких игрушек
7.	Создание web-сайта
8.	Изготовление продукции
9.	Call-центр
10.	Рекламное агентство
11.	Проведение акций провайдером
12.	Оформление кредита
13.	Изготовление программного продукта
14.	Печатное агентство
15.	Процесс создания статьи
16.	Процесс создания персонажа
17.	Разработка интернет-магазина
18.	Агентство недвижимости
19.	Кинотеатр
20.	Зоомагазин
21.	Отдел кадров
22.	Оформление загранпаспорта
23.	Склад предприятия
24.	Информационная система тестирования
25.	Квест комнаты

Контрольные вопросы:

1. Что такое входной бизнес процесс?
2. Что такое выходной бизнес процесс?
3. Что такое управляющий элемент?
4. Что такое механизм, кто им выступает?
5. Как перейти на декомпозицию первого уровня?

Содержание отчета:

13. Тема, цель практической работы
14. Поэтапное описание выполнения практической работы
15. Скриншоты или результат практической
16. Краткие ответы на контрольные вопросы

Выводы

Практическая 9

Тема: Разработка декомпозиции в нотации IDEF3 на основе организационной структуры, в среде разработке ER win. Шаблон построения, основные элементы построения.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Нотация IDEF3

IDEF3 является технологией, хорошо приспособленной для сбора данных, требующихся для проведения структурного анализа системы.

В отличие от большинства технологий моделирования бизнес-процессов, **IDEF3** не имеет жестких синтаксических или семантических ограничений, делающих неудобным описание неполных или нецелостных систем. Кроме того, автор модели (системный аналитик) избавлен от необходимости смешивать свои собственные предположения о функционировании системы с экспертными утверждениями в целях заполнения пробелов в описании предметной области. На рис. 3.1 изображен пример описания процесса с использованием **методологии IDEF3**.

IDEF3 также может быть использован как метод проектирования бизнес-процессов. **IDEF3-моделирование** органично дополняет традиционное моделирование с использованием стандартаметодологии IDEF0. В настоящее время оно получает все большее распространение как вполне жизнеспособный путь построения моделей проектируемых систем для дальнейшего анализа имитационными методами. Имитационное тестирование часто используют для оценки эксплуатационных качеств разрабатываемой системы. Более подробно методы имитационного анализа будут рассмотрены ниже.

Методология IDEF3

Стандарт IDEF0, который был рассмотрен ранее является развитием классического DFD – подхода и предназначен для описания бизнес-процессов верхнего уровня. Для описания временной последовательности и алгоритмов выполнения работ стандарт IDEF0 не подходит. Для решения этой задачи стандарт IDEF0 получил дальнейшее развитие в результате чего был разработан стандарт IDEF3, который входит в семейство стандартов IDEF.
Стандарт IDEF3 предназначен для описания бизнес-процессов нижнего уровня и содержит объекты – логические операторы, с помощью которых показывают альтернативы и места принятия решений и в бизнес-процессе, а также объекты – стрелки с помощью которых показывают временную последовательность работ в бизнес-процессе (рис. 5).

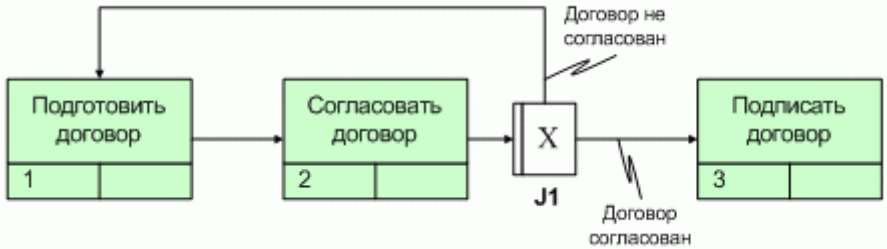
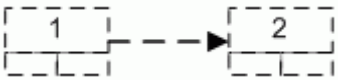


Рис. 5. Схема бизнес-процесса в стандарте IDEF3.

В отличие от классической методологии WFD в стандарте IDEF3 связи между работами делятся на три типа, обозначения, названия и смыл которых, приведены в таблице 3.

Таблица 3. Типы связей между работами в стандарте IDEF3.

Название связи	Вид связи	Смысл связи
Связь предшествования		Обозначает, что вторая работа начинает выполняться после завершения первой работы.

Связь отношения		Обозначает, что вторая работа может начаться и даже закончиться до того момента, когда закончится выполнение первой работы.
Связь потоков объектов		Одновременно обозначает временную последовательность работ и материальный либо информационный поток. В данном случае вторая работа начинает выполняться после завершения первой работы. При этом выходом первой работы объект название которого надписано над стрелкой (в данном случае документ). Эта связь также обозначает, что объект порождаемый первой работой, используется в последующих работах.

Помимо наличия нескольких типов связей между работами в стандарте IDEF3 логические операторы, которые в данном случае называются перекрестками также делятся на несколько типов: "Исключающий ИЛИ", "И" и "ИЛИ".

Перекресток "Исключающий ИЛИ" обозначает, что после завершения работы "А", начинает выполняться только одна из трех расположенных параллельно работ B, C или D в зависимости от условий 1, 2 и 3. Перекресток "И" обозначает, что после завершения работы "А", начинают выполняться одновременно три параллельно расположенные работы B, C и D. Перекресток "ИЛИ" обозначает, что после завершения работы "А", может запуститься любая комбинация трех параллельно расположенных работ B, C и D. Например может запуститься только одна из них, могут запуститься три работы, а также могут запуститься двойные комбинации B и C, либо C и D, либо B и D. Перекресток "Исключающий ИЛИ" является самым неопределенным, так как предполагает несколько возможных сценариев реализации бизнес-процесса и применяется для описания слабо формализованных ситуаций.

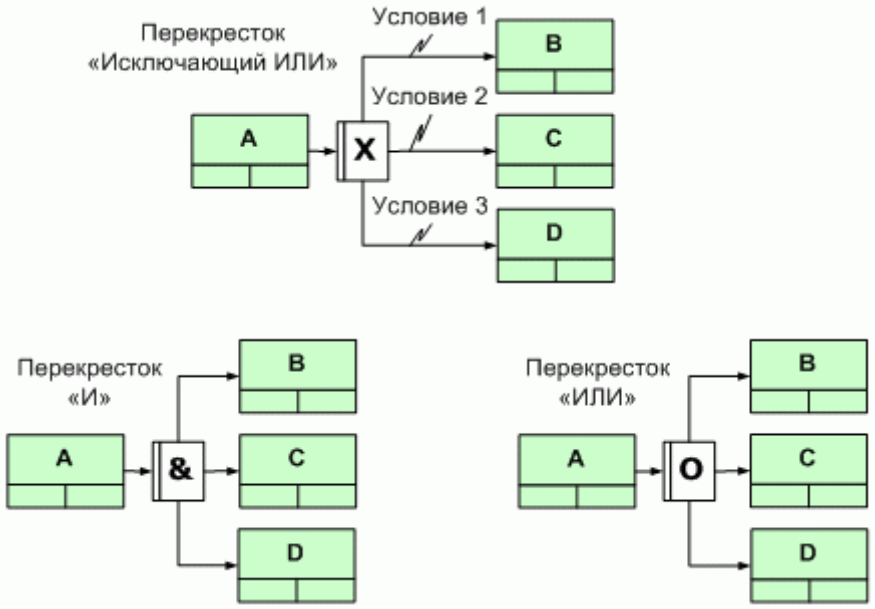


Рис. 6. Применение перекрестков "Исключающий ИЛИ", "И" и "ИЛИ" - схемы расхождения.

Перекрестки "И" и "ИЛИ" подразделяются еще на два подтипа – синхронные и асинхронные. Перекрестки синхронного типа обозначают, что работы B, C и D запускаются одновременно после завершения работы A. Перекрестки асинхронного типа требований к одновременности не предъявляют.

взаимосвязи работ и перекрестков называются схемами расхождения, так как от перекрестков расходятся несколько работ. Существует и другие схемы взаимосвязи перекрестков и работ – это так называемые схемы схождения, когда к перекрестку подходит несколько работ (рис. 7).

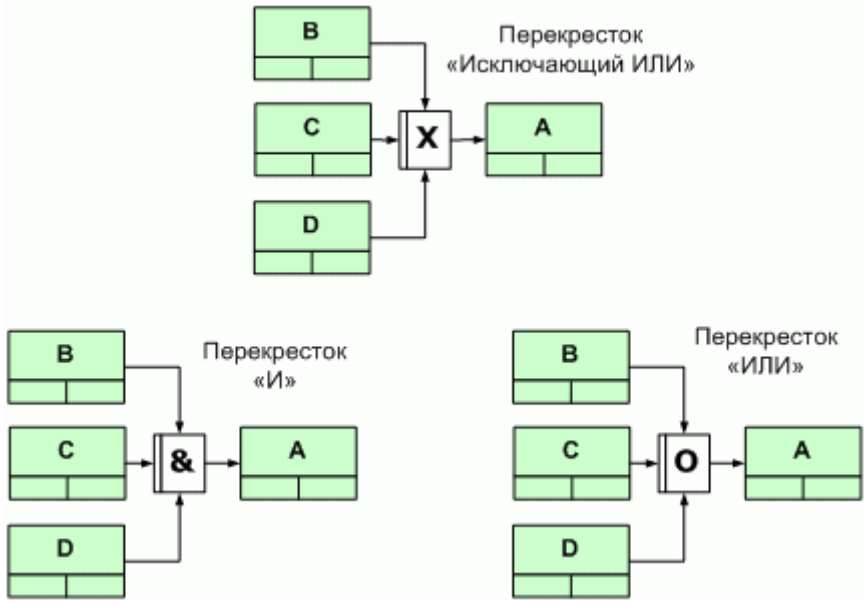


Рис. 7. Применение перекрестков "Исключающий ИЛИ", "И" и "ИЛИ" - схемы схождения.

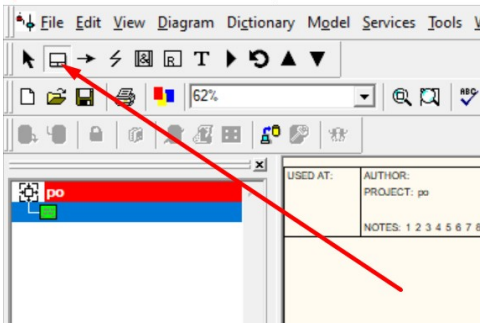
В таблице 4 приведены обозначения, названия и смысл всех типов перекрестков как в схемах схождения, так и в схемах расхождения.

Таблица 4. Обозначения, названия и смысл типов перекрестков в схемах схождения и расхождения.

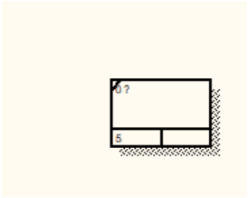
Название перекрестков		Обозначение перекрестков	Смысл перекрестков	
			Схема расхождения	Схема схождения
"Исключающий ИЛИ"		X	Только одна последующая работа запускается	Только одна предшествующая работа должна быть завершена
"И"	**Асинхронный**	&	Все последующие работы запускаются	Все предшествующие работы должны быть завершены
	Синхронный	&	Все последующие работы запускаются одновременно	Все предшествующие работы должны быть завершены одновременно
"ИЛИ"	**Асинхронный**	O	Одна или несколько последующих работ запускаются	Одна или несколько предшествующих работ должны быть завершены
	Синхронный	O	Одна или несколько последующих работ запускаются одновременно	Одна или несколько предшествующих работ должны быть завершены одновременно

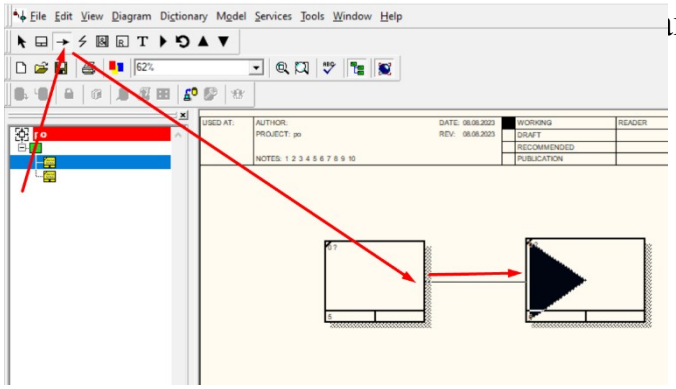
Последним отличием стандарта IDEF3 в отличие от классической методологии WFD является использование на схеме бизнес-процесса такого элемента как "объект ссылки", который связывается с работами и перекрестками. С помощью объектов ссылки показывается прочая важная информация, которую целесообразно зафиксировать при описании бизнес-процесса.

ХОД РАБОТЫ



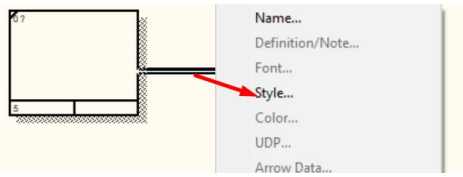
Добавить новый процесс на диаграмму. Выбрать в меню блок процесс и кликнуть по рабочему полю.

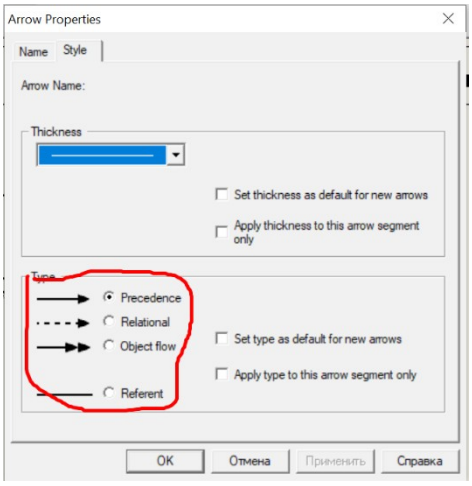




ми, для этого на панели инструментов выберите стрелку связей, подведите к процессу откуда будет исходить поток, после чего подведите к процессу или к оператору куда будет вливаться запрос.

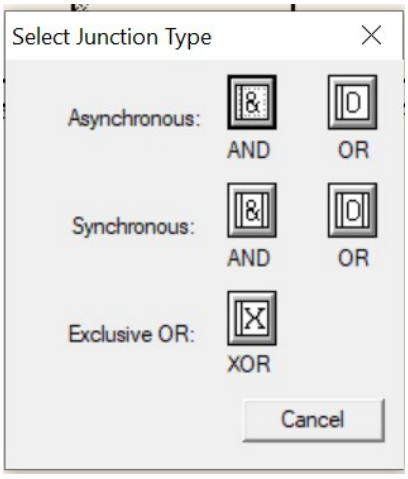
После чего нажать на поток и выбрать стиль нужного потока из меню.





После чего нужно, в диалоговом окне выбрать любой тип который нам нужен

Чтобы Добавить оператор связи, нужно на панели нажать значок оператора и на основном поле клацнуть правой кнопкой мыши по полю для установки вида оператора.



После чего из диалогового окна выбрать нужный оператор связи.

Задание 1

1.По схеме показанной ниже, создайте модель в нотации idef3
2. установите типы связей как на риске и выберите нужные операторы связи процессов.
3. в отчете опишите все процессы, виды и типы потоков, которые здесь использовались

Контрольные вопросы:

6. Что такое входной бизнес процесс?
7. Какие операторы показаны на схеме в практической работе?

8. Что такое синхронность и асинхронность?
9. Какие типы связей знаете?
10. Какие операторы знаете?
11. Дайте определение типу исключающий «ИЛИ»

Содержание отчета:

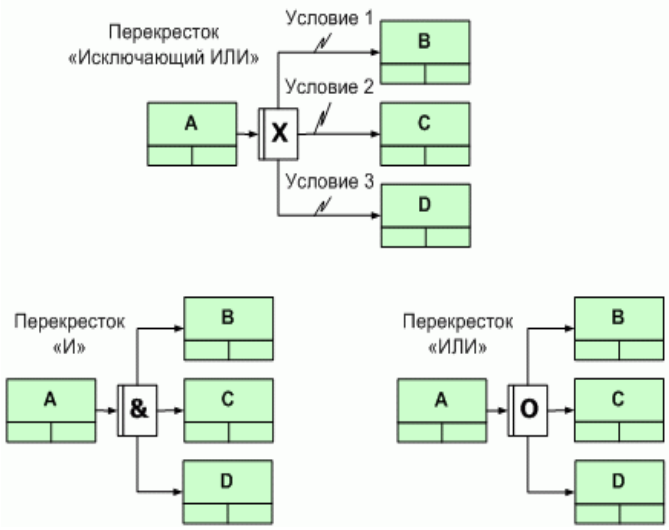
17. Тема, цель практической работы
18. Поэтапное описание выполнения практической работы
19. Скриншоты или результат практической
20. Краткие ответы на контрольные вопросы

Выводы

Практическая 10

Тема: Разработка декомпозиции 2 уровня нотации IDEF3 на основе организационной структуры в среде разработке ER win по индивидуальному заданию

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ



Перекресток "Исключающий ИЛИ" обозначает, что после завершения работы "А" (рис. 6), начинает выполняться только одна из трех расположенных параллельно работ B, C или D в зависимости от условий 1, 2 и 3.

Перекресток "И" обозначает, что после завершения работы "А", начинают выполняться одновременно три параллельно расположенные работы B, C и D.

Перекресток "ИЛИ" обозначает, что после завершения работы "А", может запуститься любая комбинация трех параллельно расположенных работ B, C и D.

Соединение типа **"исключающее “или”**:

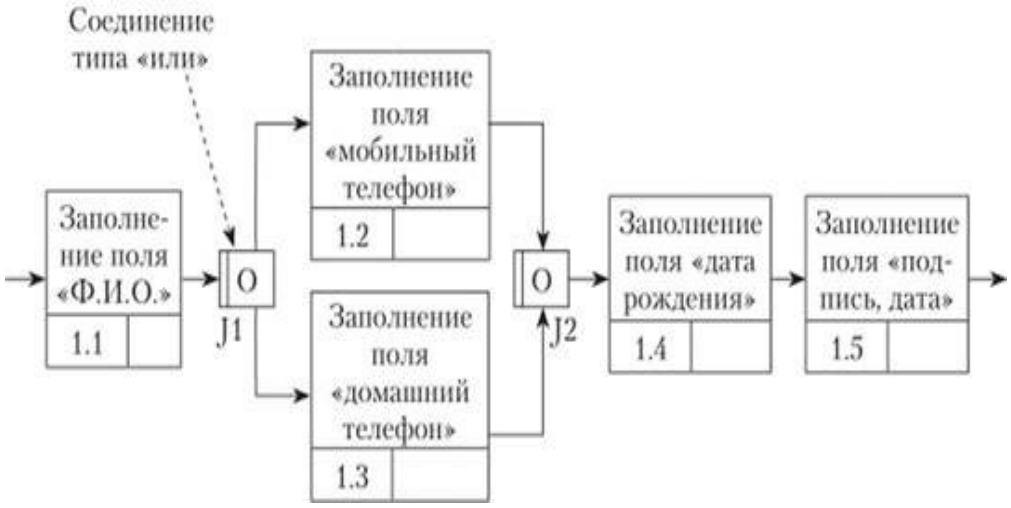
"исключающее “или”" используется для того, чтобы показать, что результатом согласования проекта договора может быть:

а) проект договора согласован;

б) по проекту договора есть замечания и он отправлен на доработку

. В первом случае, если он согласован, то осуществляется следующее действие — подписание договора. Во втором случае, когда по нему есть замечания, осуществляется его доработка.

Соединение типа **"или"**

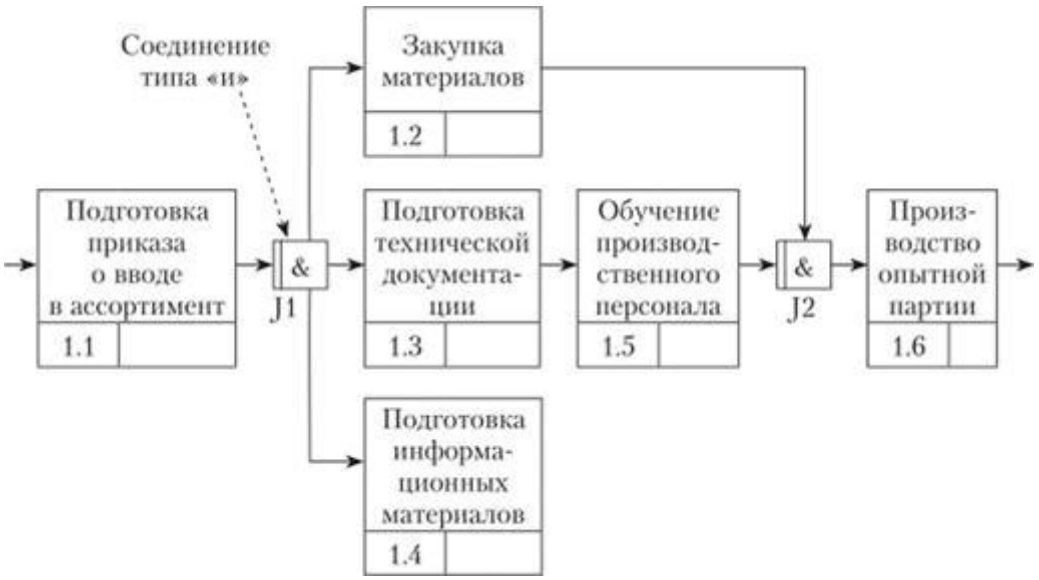


процесса "Заполнение анкеты"

пример использования соединения типа "или", где после действия "Заполнение поля “Ф.И.О.”" может быть выполнено действие "Заполнение поля “мобильный телефон”" или действие "Заполнение поля “домашний телефон”" либо оба эти действия. Одно из них точно должно быть выполнено.

Перекресток "Исключающий ИЛИ" является самым неопределенным, так как предполагает несколько возможных сценариев реализации бизнес-процесса и применяется для описания слабо формализованных ситуаций.

Соединение **типа «И»**

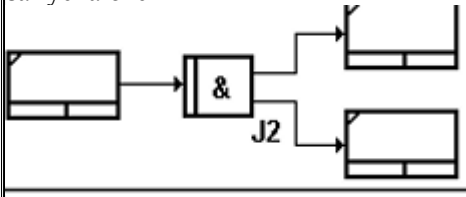
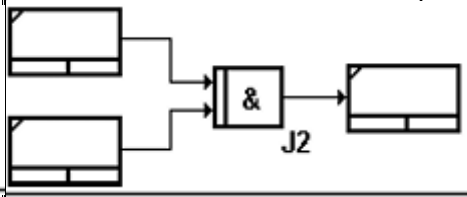


Следует учитывать, что если соединение "и" инициирует выполнение последнего действия, то все действия, которые присоединяются к сворачиваемому соединению типа "и" должны быть выполнены полностью.

"Подготовка к продаже нового изделия" состоит из следующих подпроцессов:

- 1.1. Подготовка приказа о вводе в ассортимент нового продукта.
- 1.2. Закупка материалов для производства изделия.
- 1.3. Подготовка технической документации по изготовлению нового изделия.
- 1.4. Подготовка информационных материалов для продвижения и продажи.
- 1.5. Обучение производственного персонала изготовлению нового изделия.
- 1.6. Производство опытной партии нового изделия.

Процессы "Закупка материалов для производства изделия", "Подготовка информационных материалов для продвижения и продажи" и "Подготовка технической документации по изготовлению изделия" начинаются сразу после того, как выпущен приказ о вводе в ассортимент нового продукта. Процесс "Производство опытной партии нового изделия" может начаться только после того, как обучен производственный персонал и закуплен материал для производства.

Название перекрестков		**Обозначение перекрестков**	**Смысл перекрестков**	
			Схема расхождения	**Схема схождения**
"Исключающий ИЛИ"		X	Только одна последующая работа запускается	Только одна предшествующая работа должна быть завершена
"И"	**Асинхронный**	&	Все последующие работы запускаются 	Все предшествующие работы должны быть завершены 
	Синхронный	&	Все последующие работы запускаются одновременно	Все предшествующие работы должны быть завершены одновременно
"ИЛИ"	**Асинхронный**	O	Одна или несколько последующих работ запускаются	Одна или несколько предшествующих работ должны быть завершены
	Синхронный	O	Одна или несколько последующих работ запускаются одновременно	Одна или несколько предшествующих работ должны быть завершены

				одновременно

ХОД РАБОТЫ

Задание 1

По выбранному ранее вами варианту проектирования 1-го уровня детализации бизнес-модели составьте 2-й уровень детализации основных процессов вашей предметной области используя:

1. 3шт основных функциональных блоков и один конечных процесс
2. 3шт различных типов связей между процессами и операторами
3. один оператор связи на выбор
4. в отчет вставить скншот модели с описание использованных операторов потоков и процессов, дать логическое описание последовательности процессов

Контрольные вопросы:

1. Что такое схема схождения процессов? Какие схемы схождения знаете?
2. Дайте определение синхронному и асинхронному оператору ИЛИ
3. Как слить два процесса в один процесс используя оператор исключающий или?
4. Что такое поток?

Содержание отчета:

21. Тема, цель практической работы
22. Поэтапное описание выполнения практической работы
23. Скриншоты или результат практической
24. Краткие ответы на контрольные вопросы

Выводы

Практическая 11-13

Тема: Разработка декомпозиции 3 уровня нотации IDEF3 на основе организационной структуры 2го уровня детализации среде разработке ER win. Установка основных потоков и операторов связи между процессами.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

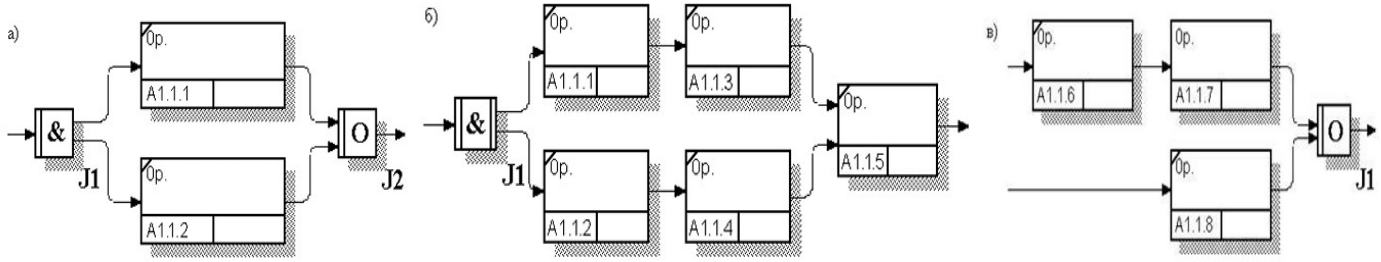
DEF3 – нотация описания бизнес-процессов. IDEF3 обычно применяется для построения процессов нижнего уровня и может использоваться при декомпозиции блоков процесса IDEF0.

В отличие от IDEF0 данная нотация не поддерживает отображение «механизмов» и «управления», зато отображает очередность выполнения работ персоналом. Несмотря на схожесть с нотацией блок-схем (FlowChart), имеет некоторые существенные отличия:

1. весь процесс строится не сверху вниз, а слева направо и при этом, как правило, ограничен количеством используемых блоков на одну диаграмму.
2. нотация изначально предназначалась для технических специалистов, поэтому содержит специальные перекрёстки, такие как, «XOR», «Synchronous OR», «Asynchronous OR», «Synchronous AND» и «Asynchronous AND», знакомые программистам, но требующие дополнительное пояснения менеджерам предприятия.
3. связи между работами делятся на три типа: *Связь предшествования*, *Связь отношения*, *Связь потоков объектов*.

Соединения могут комбинироваться для создания более сложных ветвлений

Комбинации соединений следует использовать с осторожностью, так как перегруженные ветвлением диаграммы сложны для восприятия.



Логически выяснить взаимосвязь блоков и разделить их на под блоки модели.

Декомпозиция диаграмм

ХОД РАБОТЫ

Задание 1

По выбранной ранее теме проектирования 2-го уровня детализации бизнес-модели составьте 1ю схему детализации 3-го уровня мелких процессов по вашей предметной области используя:

1. минимум 3 и более процесса;
2. минимум 3 и более различных оператора связи;
3. минимум 3 и более различных потока данных.
4. Выполните одно схождение или расхождение процессов.

Результатом должна быть одна схема 3го уровня с описание использованных операторов, последовательности взаимодействия всех процессов в модели.

Задание 2

По выбранной ранее теме проектирования 2-го уровня детализации бизнес-модели составьте 2ю схему детализации 3-го уровня мелких процессов по вашей предметной области используя:

5. минимум 3 и более процесса;
6. минимум 3 и более различных оператора связи;
7. минимум 3 и более различных потока данных.
8. Выполните одно схождение или расхождение процессов.

Результатом должна быть одна схема 3го уровня с описание использованных операторов, последовательности взаимодействия всех процессов в модели.

Задание 3

По выбранной ранее теме проектирования 2-го уровня детализации бизнес-модели составьте 3ю схему детализации 3-го уровня мелких процессов по вашей предметной области используя:

9. минимум 3 и более процесса;
10. минимум 3 и более различных оператора связи;
11. минимум 3 и более различных потока данных.
12. Выполните одно схождение или расхождение процессов.

Результатом должна быть одна схема 3го уровня с описание использованных операторов, последовательности взаимодействия всех процессов в модели.

Контрольные вопросы:

1. Как изменить номер процесса на схеме в левом углу?
2. Как установить новый статус диаграмме процессов? Т.е. изменить время публикации и изменения схемы?
3. Как изменить автора диаграммы?

Содержание отчета:

25. Тема, цель практической работы
26. Поэтапное описание выполнения практической работы
27. Скриншоты или результат практической
28. Краткие ответы на контрольные вопросы

Практическая 14

Тема: имитационное моделирование. AnyLogic обзор инструмента. Установка и создание проекта.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Рабочее пространство AnyLogic

• **Графический редактор** позволяет редактировать диаграмму агента. Вы можете добавлять элементы на диаграмму, перетаскивая их из Палитры на холст редактора. Синяя прямоугольная рамка ограничивает ту область холста, которая будет отображаться в окне модели при ее запуске.

• **Панель Проекты** отображает содержимое моделей AnyLogic, открытых в рабочем пространстве в текущий момент. Элементы каждой модели отображаются в виде иерархического дерева, для облегчения навигации.

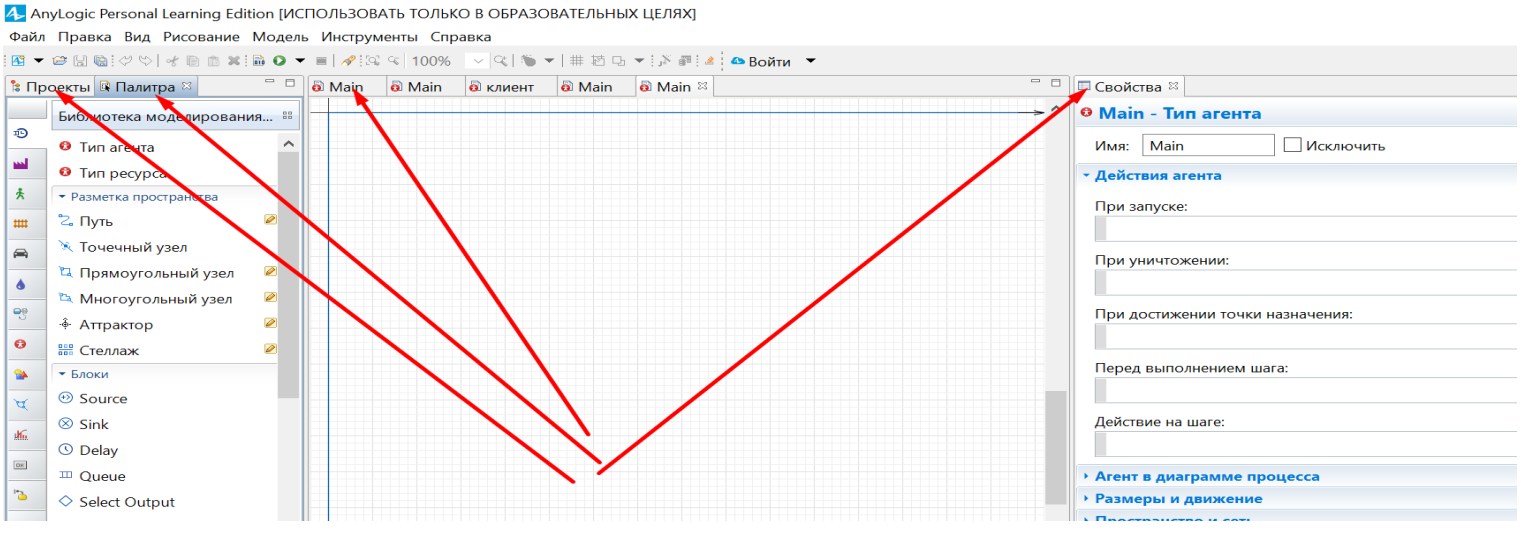
• **Панель Палитра** содержит все графические элементы AnyLogic, сгруппированные в отдельные палитры. Чтобы добавить тот или иной элемент в модель, перетащите соответствующий элемент из палитры в графический редактор.

• **Панель Свойства** позволяет вам просматривать и изменять свойства выделенных в текущий момент элементов модели. Графический редактор Панели Проекты и Палитра. Для переключения щелкните по заголовку панели.

• Чтобы открыть или закрыть панель, выберите в меню Вид соответствующий пункт с именем панели.

• Чтобы изменить размер панели, перетащите мышью ее границу.

• Вы всегда можете воспользоваться опцией Восстановить расположение панелей в меню Инструменты, чтобы вернуть расположение панелей по умолчанию.



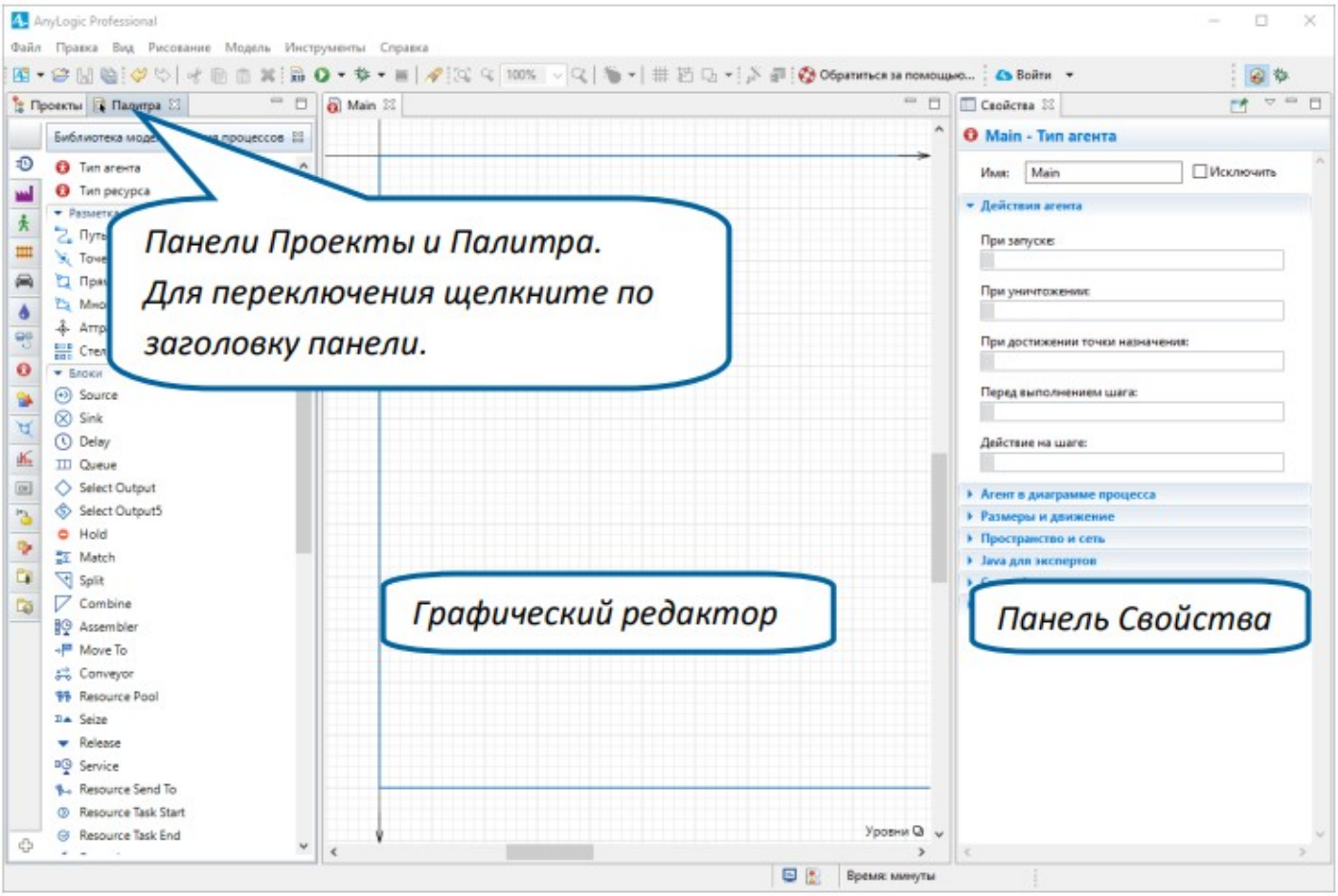
Рабочие области:

1.**Графический редактор** позволяет редактировать диаграмму агента. Вы можете добавлять элементы на диаграмму, перетаскивая их из Палитры на холст редактора. Синяя прямоугольная рамка ограничивает ту область холста, которая будет отображаться в окне модели при ее запуске.

2.**Панель Проекты** отображает содержимое моделей AnyLogic, открытых в рабочем пространстве в текущий момент. Элементы каждой модели отображаются в виде иерархического дерева, для облегчения навигации.

3.**Панель Палитра** содержит все графические элементы AnyLogic, сгруппированные в отдельные палитры. Чтобы добавить тот или иной элемент в модель, перетащите соответствующий элемент из палитры в графический редактор.

4.**Панель Свойства** позволяет вам просматривать и изменять свойства выделенных в текущий момент элементов модели.



ХОД РАБОТЫ

Для установки перейдите посылке https://www.anylogic.ru/resources/anylogic-for-academia/ **1.**нажмите кнопку скачать, после чего **2.** выберите учебную версию , **3.**далее заполните форму диалогового окна на свой ПК.

1.

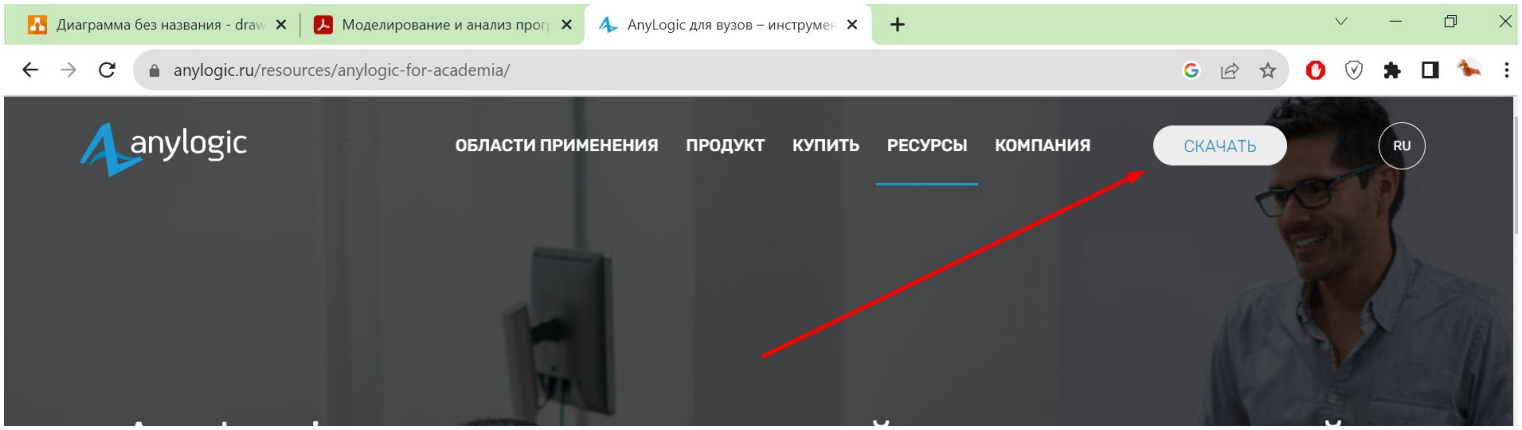


Диаграмма без названия - draw
Моделирование и анализ прогр
AnyLogic для вузов – инструмен
anylogic.ru/resources/anylogic-for-academia/
anylogic
ОБЛАСТИ ПРИМЕНЕНИЯ
ПРОДУКТ
КУПИТЬ
РЕСУРСЫ
КОМПАНИЯ
СКАЧАТЬ
RU

2.

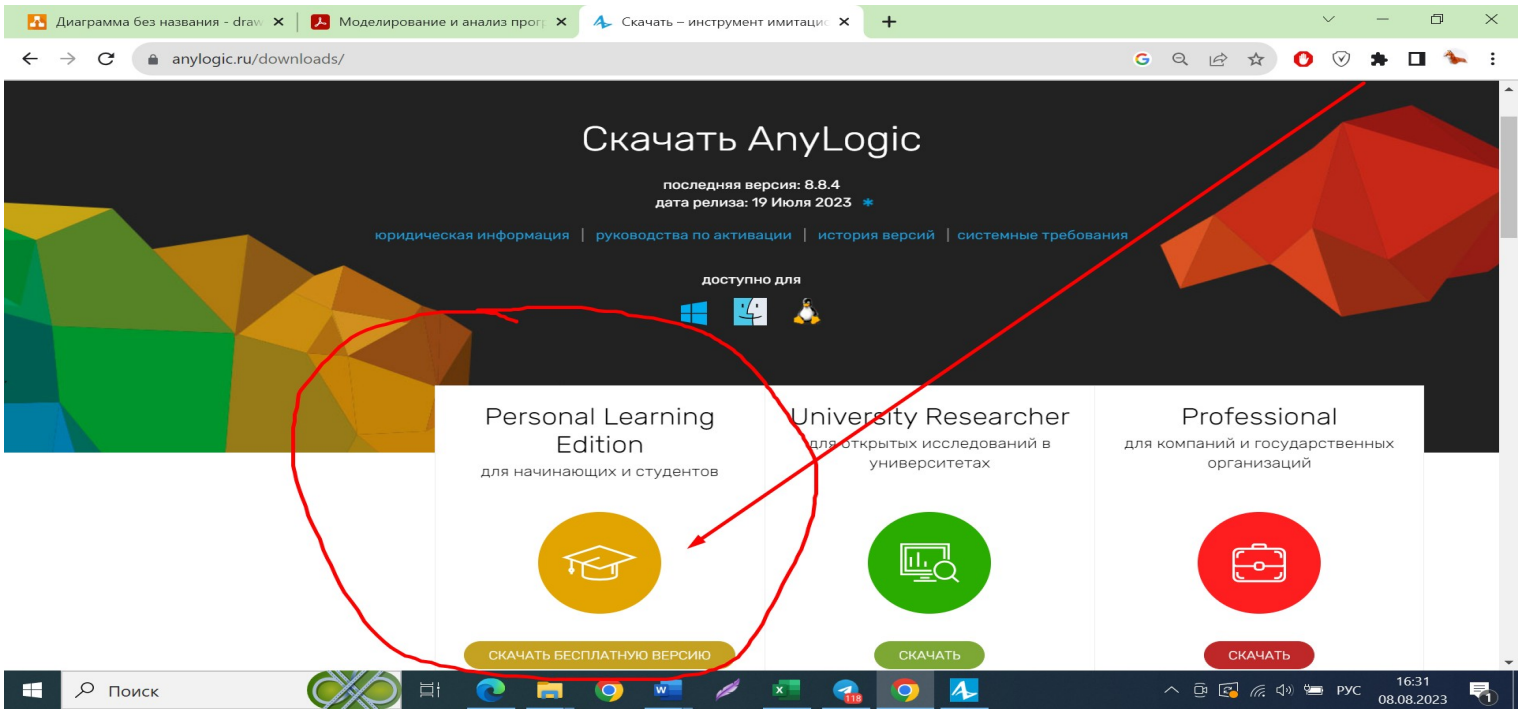
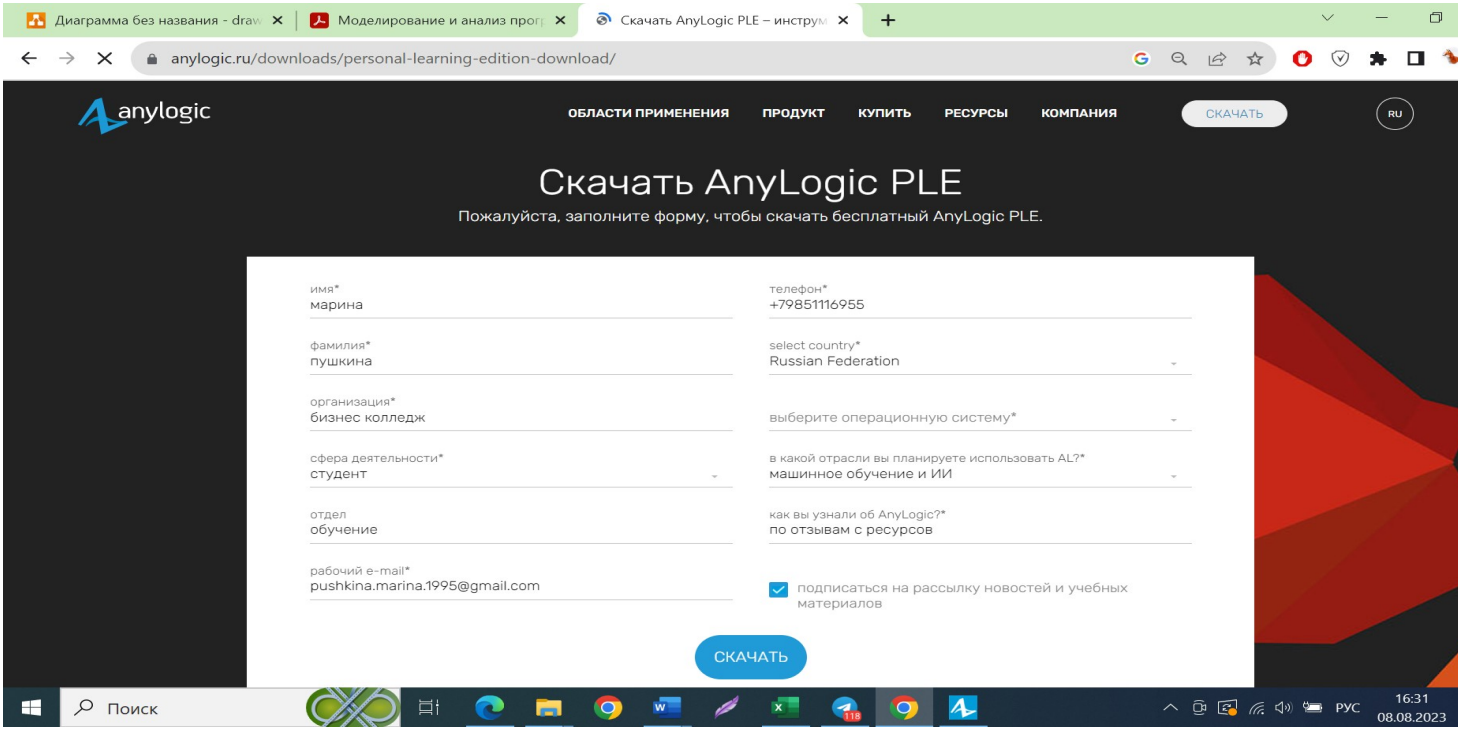
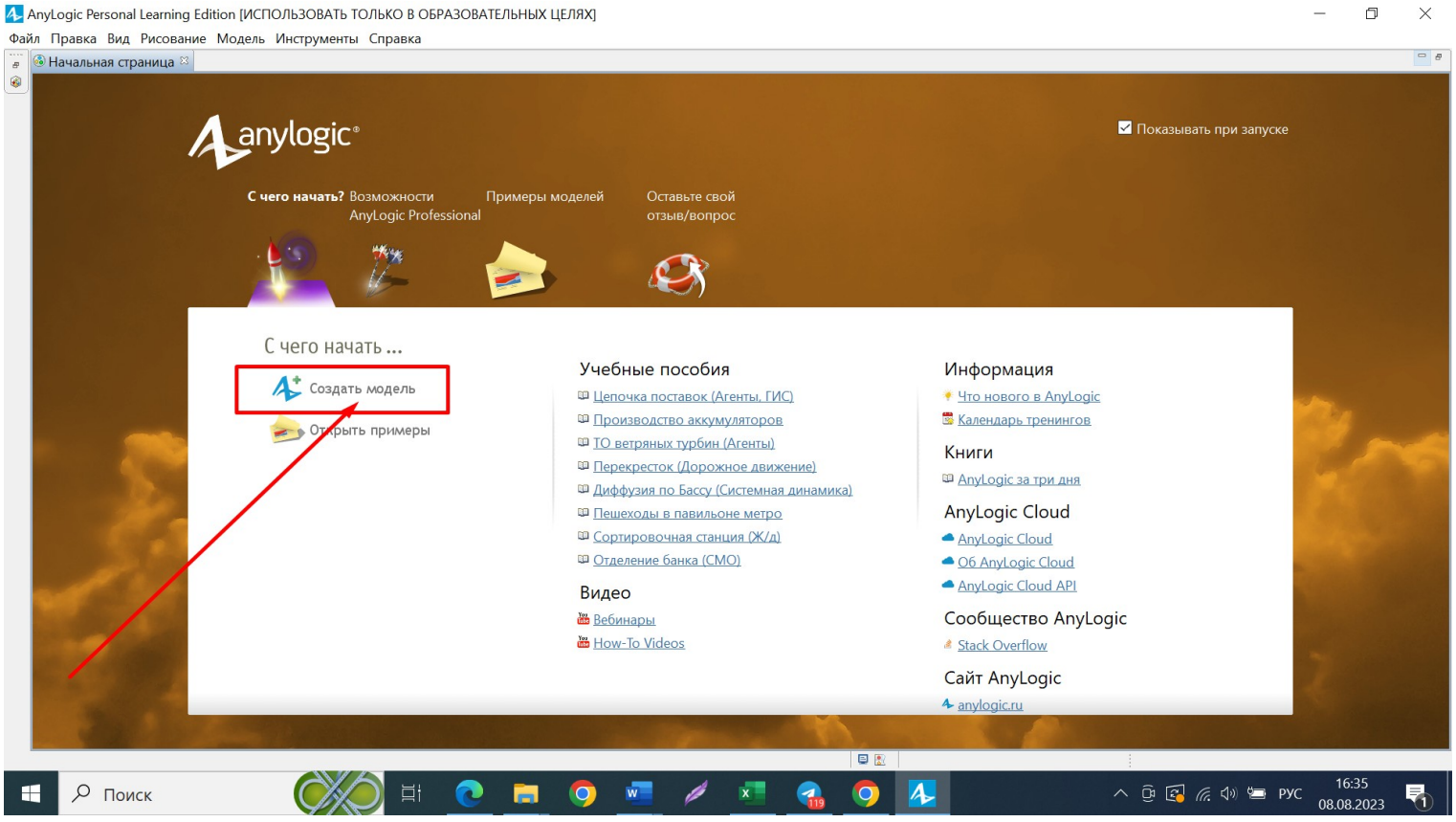


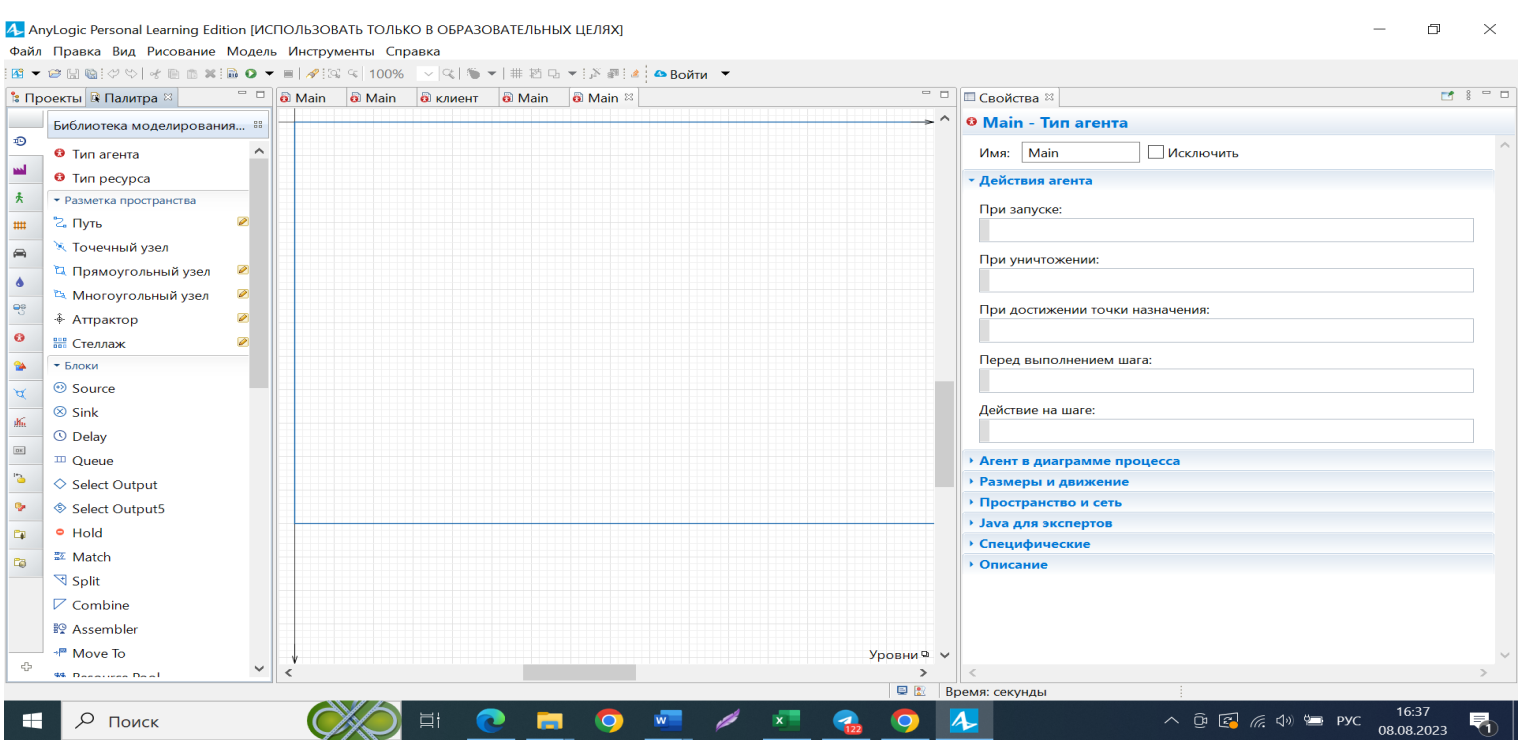
Диаграмма без названия - draw
Моделирование и анализ прогр
Скачать – инструмент имитаци
anylogic.ru/downloads/
Скачать AnyLogic
последняя версия: 8.8.4
дата релиза: 19 Июля 2023
юридическая информация | руководства по активации | история версий | системные требования
доступно для
Personal Learning Edition
для начинающих и студентов
СКАЧАТЬ БЕСПЛАТНУЮ ВЕРСИЮ
University Researcher
для открытых исследований в университетах
СКАЧАТЬ
Professional
для компаний и государственных организаций
СКАЧАТЬ
Поиск
РУС
16:31
08.08.2023



4. После чего запустите exe файл
5. Запустите ПО.
6. Перед вами откроется данное окно, нажмите создать модель



7. После чего дайте имя модели ТЕРМИНАЛ
   У ВВАС ЗАПУСТИТСЯ ПО И СТАНУТ ДОСТУПНЫ ИНСТРУМЕНТЫ МОДЕЛИРОВАНИЯ!



Если вам нужно создать модели в уже открытом ПО , то выполните следующие действия Начальная страница

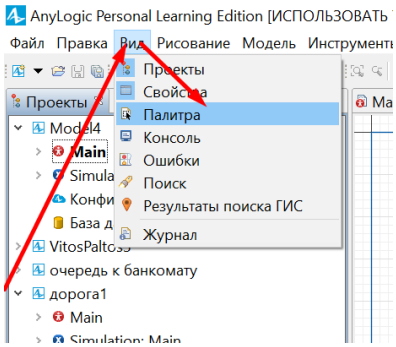
1. Закройте Начальную страницу и создайте новую модель. Для этого выберите Файл > Создать > Модель из главного меню AnyLogic. Откроется диалоговое окно Новая модель.

2. В поле Имя модели введите имя новой модели: тест.

3. В поле Местоположение выберите каталог, в котором вы хотите сохранить файлы модели.

4. Щелкните по кнопке Готово. Будет создана новая модель.

8. Для того чтобы использовать любой элемент, выберите его из меню слева и перетяните



Чтобы открыть или закрыть панель, выберите в меню Вид соответствующий пункт с именем панели.

• Чтобы изменить размер панели, перетащите мышью ее границу.

• Вы всегда можете воспользоваться опцией Восстановить расположение панелей в меню Инструменты, чтобы вернуть расположение панелей по умолчанию. на рабочую область.

Задание 1

1.выберите в палитре любые две библиотеке и перетяните любые два объекта на рабочую область

2. в свойствах справа измените их имя

3.и измените любые свойства из стандартных на другие, какие вам понравится.

Сделайте скриншот и вставьте в отчет.

Контрольные вопросы:

1. **Н**азовите 4 рабочих пространства ПО
2. Дайте определение двум рабочим пространствам
3. Как изменить рабочую область(ее размер)?
4. Как восстановить палитру?
5. Как добавить элемент из библиотеке на рабочую область?

Содержание отчета:

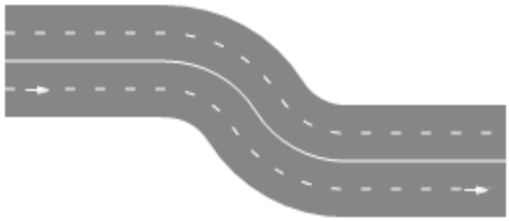
1. Тема, цель практической работы
2. Поэтапное описание выполнения практической работы
3. Скриншоты или результат практической
4. Краткие ответы на контрольные вопросы

Практическая 15-16

Тема: Имитационное моделирование. AnyLogic разработка типового проекта дорожного движения с перекрестками.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

дорога

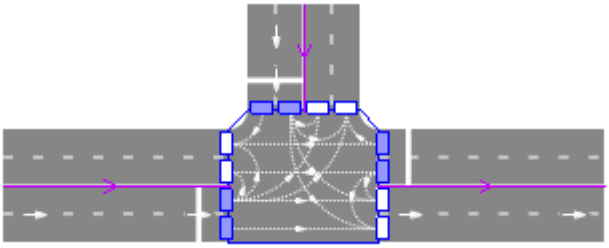


Элемент **Дорога** является графическим элементом разметки пространства, это непрерывная дорога(т.е. такая дорога, которая не содержит перекрестков). Используя дороги, перекрестки и другие элементы разметки пространства, вы создаете дорожные сети для моделей Библиотеки дорожного движения. Дорожная сеть может состоять даже из одной дороги: AnyLogic автоматически создает ее, когда вы рисуете дорогу в графическом редакторе агента.

Дорога может содержать множество прямых и дуговых сегментов (дорога на рисунке выше состоит из двух прямых сегментов и одного дугового сегмента в центре).

Следующие свойства дорог задаются не в свойствах дороги, а в свойствах дорожной сети, которой принадлежит эта дорога: направление движения, ширина полосы, цвет дорожной разметки и т.д.

Дорога может содержать произвольное количество полос. Дорога может быть как односторонняя, в этом случае все автомобили движутся в одном направлении, так и двусторонняя, в этом случае на дороге будет несколько полос основного движения и несколько полос встречного движения. Дорога на рисунке выше содержит две полосы основного движения (стрелка на дороге находится на полосе основного движения) и две полосы встречного движения (они находятся сверху, движение транспорта на них происходит справа налево). Библиотека дорожного движения не поддерживает полосы, по которым транспорт может совершать движение в обе стороны.



Перекресток

С помощью элемента разметки пространства **Перекресток** вы можете соединять две или больше дорог. Используя дороги, перекрестки и другие элементы разметки пространства, вы создаете дорожные сети для моделей Библиотеки дорожного движения.

Отображаемые на перекрестке белые точечные линии — соединители полос — задают разрешенные направления движения транспорта на перекрестке.

CarSource

Создает автомобили и пытается поместить их в указанное место дорожной сети. Автомобиль можно поместить на указанную дорогу или парковку (это задается параметром **Появляется**). Аналогично блоку **Source**, автомобили могут создаваться согласно:

- заданной интенсивности
- времени между прибытиями
- изменяющейся во времени интенсивности, заданной с помощью расписания
- расписанию, задающему точные времена и количество прибывающих автомобилей
- вызову функции блока inject()

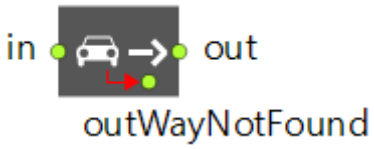
Количество создаваемых автомобилей можно ограничить (выбрав опцию **Ограниченное кол-во прибытий** и задав **Максимальное кол-во прибытий**).

CarDisposeУдаляет машины из модели.

Есть два способа удаления автомобилей:

- Автомобиль может выехать за пределы дорожной сети по незамкнутому пути или после достижения указанной стоп-линии, в этом случае блок **CarDispose** нужно будет поместить после последнего блока **CarMoveTo**.
- Автомобиль можно удалить из модели, когда он находится на парковке или автобусной остановке. В этом случае нет необходимости удалять автомобиль сразу после того, как он заедет на парковку или остановку.

Удалять автомобили нужно именно с помощью блока **CarDispose**, а не блоков **Sink** или **Exit**.



CarMoveTo
Блок, который управляет движением автомобиля. Автомобиль может ехать, только когда он находится в блоке **CarMoveTo**. Автомобиль пытается рассчитать путь от своего текущего места до указанного места назначения, когда поступает в блок **CarMoveTo**.

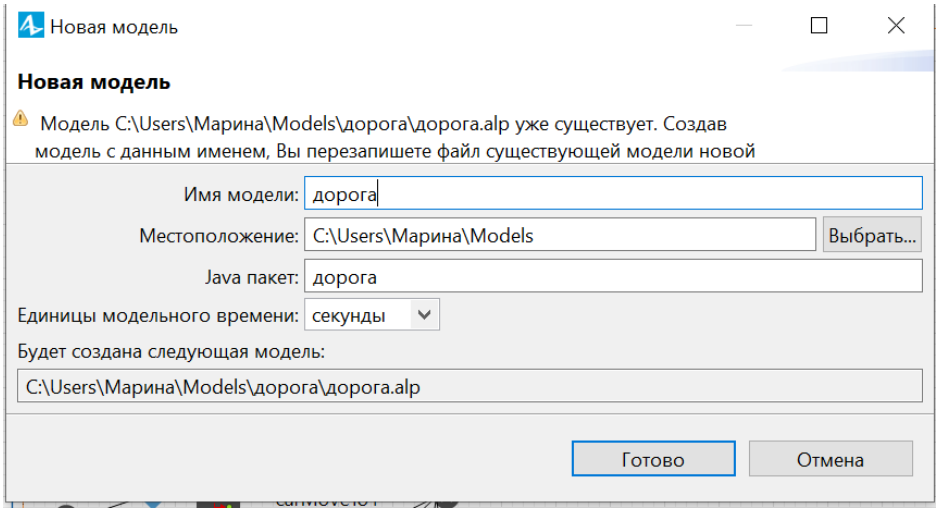
В качестве цели движения могут выступать: **дорога**, **парковка**, **автобусная остановка** или **стоп-линия**. Указанное место назначения должно находиться в той же дорожной сети, что и автомобиль. Если от текущего местоположения автомобиля к указанному месту нет пути, автомобиль покидает блок через порт **outWayNotFound**.

Если в качестве цели движения задана дорога, необходимо дополнительно указать направление движения.

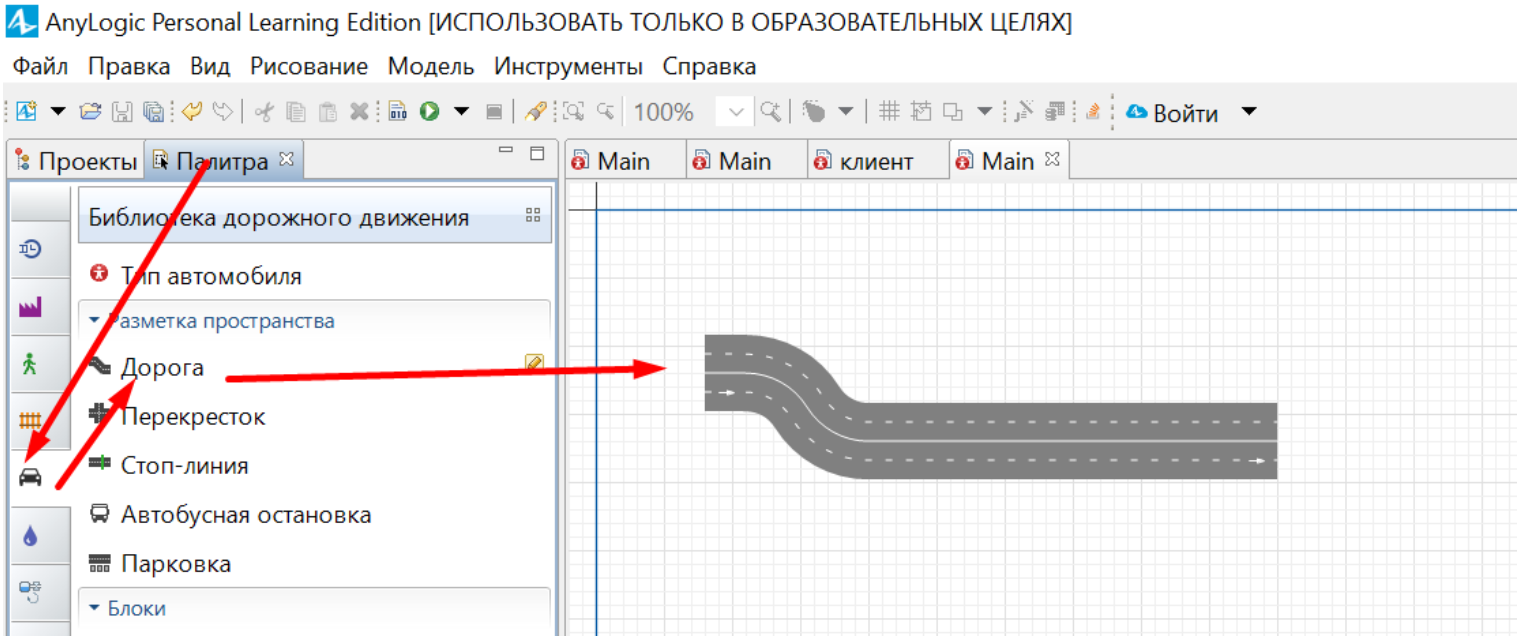
ХОД РАБОТЫ

Задание1

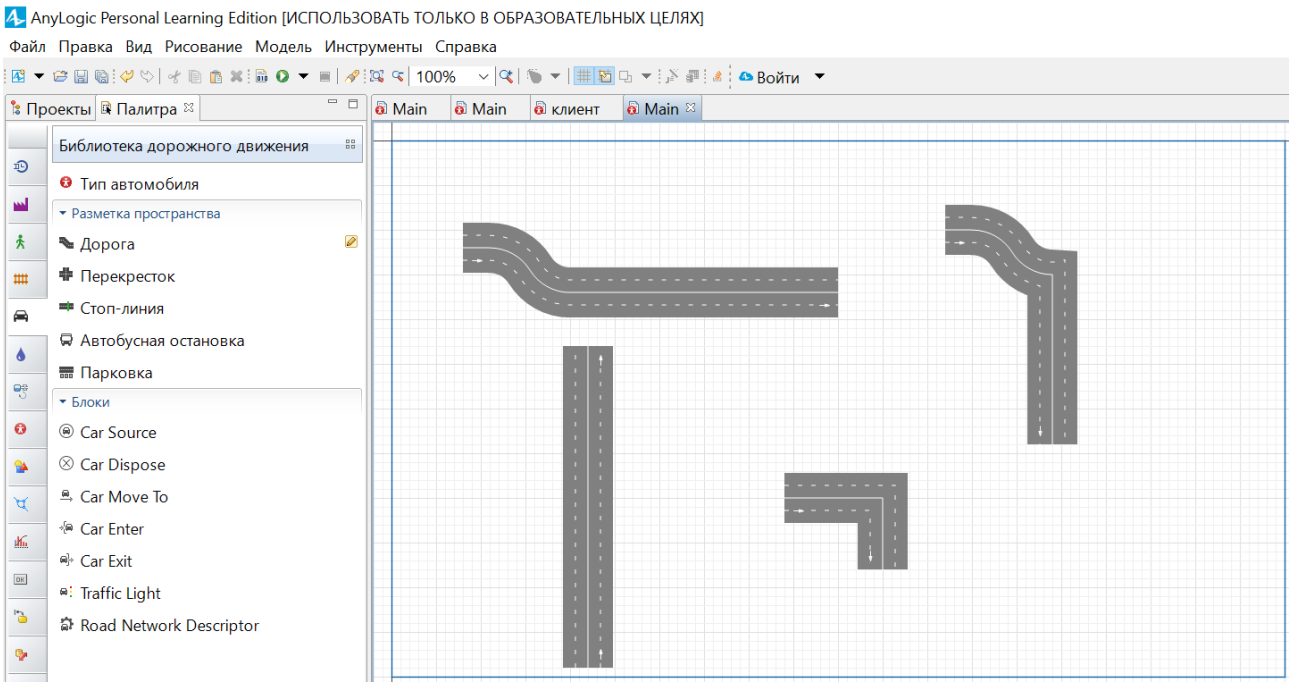
1. Запустите ПО
2. Создайте новый проект под названием Дорога



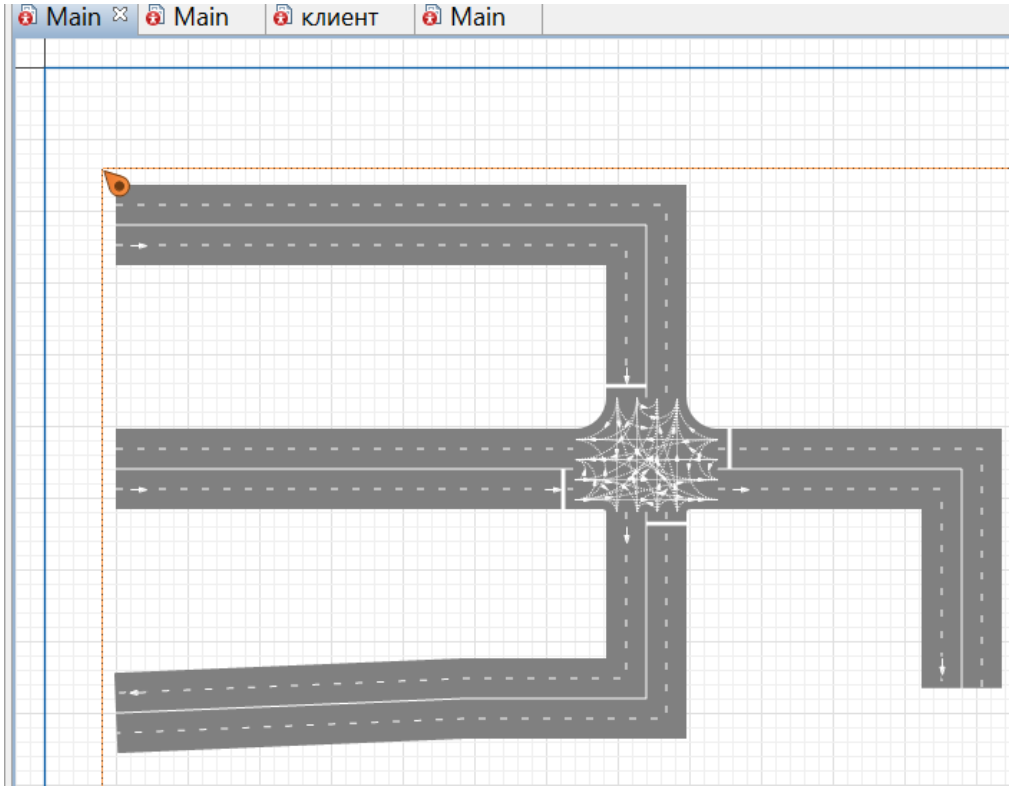
3. Перейдите в библиотеку Дорожного движения и выберите элемент дороги и перетяните на рабочую область. Попробуйте ее вида изменить и изогнуть.

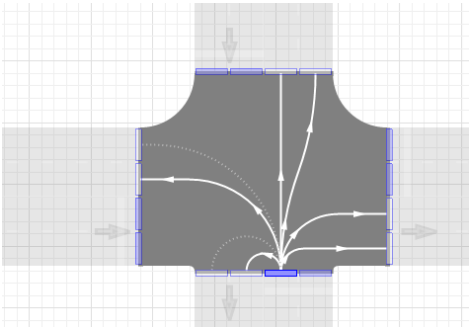


4. Создайте 4 шт элемента дороги

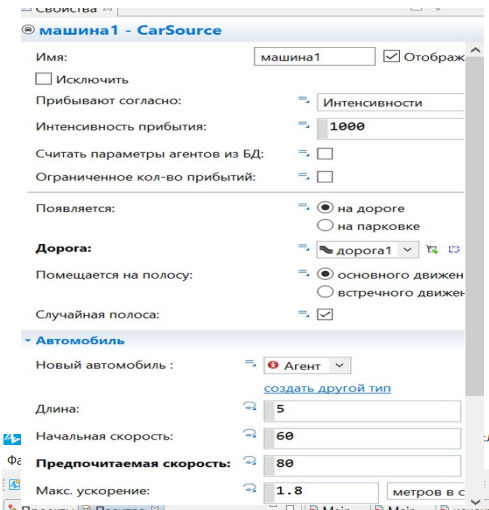


5. Измените в свойствах их имя, на дорога1, дорога2, дорога3, дорога4
6. Соедините дороги перекрестком. Из библиотеки дорожного движения, возьмите перетяните элемент перекресток



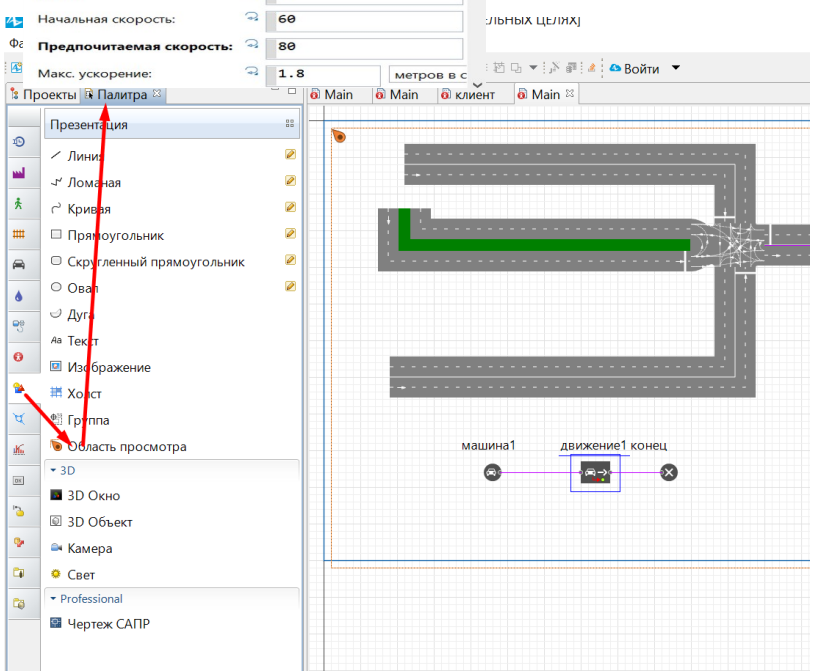
7. 

Нажмите на любую дорогу, и вы увидите направление ее движения. Наведите, нажмите на край полосы перед перекрестком, чтобы добавить или убавить количество возможных поворотов.



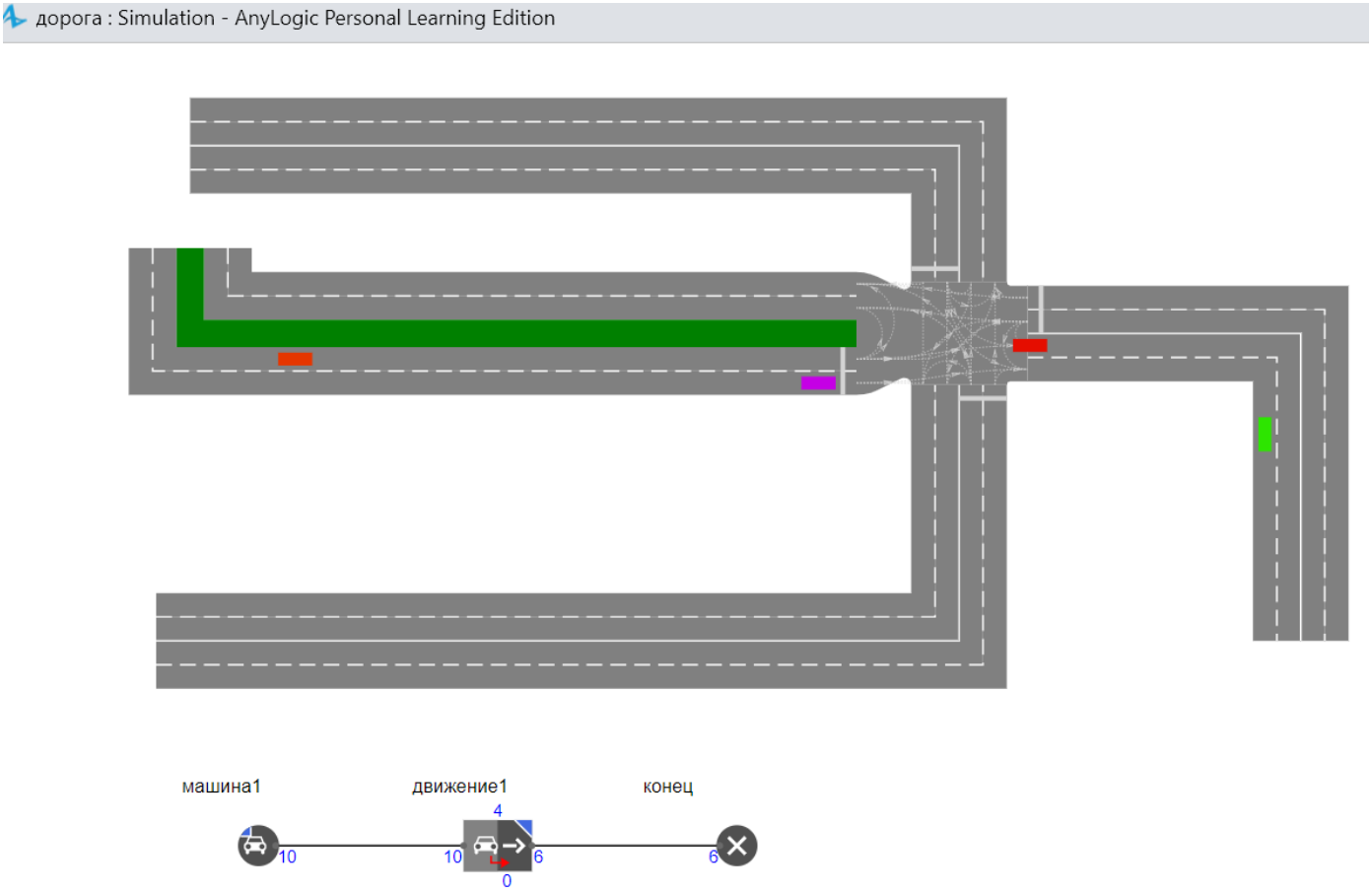
8. Добавьте источник движения- машину на рабочую область из библиотеки дорожного движения, переименуйте ее. Car Source и в свойствах укажите на какой дороге она у вас будет появляться и с какой скоростью.

9. Определите процесс самого движения, добавив Car Move To, нажмите на элемент и укажите следующую дорогу по которой будет ехать машина в свойствах данного элемента.



10. Определите конец движения, для этого добавьте объект. Car Dispose

11. Соедините все элементы между собой
12. Перейдите в библиотеку, выберите библиотеку презентаций и выберите элемент область просмотра и добавьте на свою рабочую область.
13. Для соединения данных элементов в один процесс возьмите любой элемент и подведите его к тому элементу с чем его нужно соединить.
14. Запустите на иметацию вашу модель!



Задание2

1. Добавьте до созданной модели 3 источника движения, сделайте чтобы они появлялись на дороге 1, дороге 2 и дороге 3, так же укажите им разную скорость движения
2. Расширьте 2 и 3ю дорогу до 6м
3. Удалите 4-ю дорогу с карты

   Результат движения вставьте в отчет.

Контрольные вопросы:

1. Назовите основные свойства источника движения машины?
2. Как называется источник для выделения рабочей области на просмотр?
3. Как машине добавить или уменьшить возможностей поворота на другую дорогу?
4. Как называется блок управляющий движение машин?
5. Дайте определение дороге

Содержание отчета:

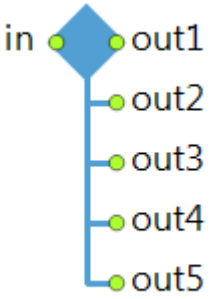
1. Тема, цель практической работы
2. Поэтапное описание выполнения практической работы
3. Скриншоты или результат практической
4. Краткие ответы на контрольные вопросы

Практическая 17-18

Тема: Имитационное моделирование. AnyLogic разработка типового проекта дорожного движения с перекрестком. Использование элемента SelectOutput с различными вероятностями появления машин и их движения.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

SelectOutput5



Блок направляет входящих агентов в один из пяти выходных портов в зависимости от выполнения заданных (детерминистических или заданных с помощью вероятностей) условий.

У блока есть три режима работы:

- **Условия** — Пользователю предлагается задать 4 условия. Когда в блок поступит новый агент, эти условия будут поочередно вычисляться одно за другим. Если будет выполнено условие 1, то агент покинет блок через порт 1, если нет, то будет проверяться условие 2, и так далее. Если не будет выполнено ни одно из условий, то агент покинет блок через последний порт. Каждое условие может зависеть как от агента, так и от каких-то внешних факторов.

• **Вероятности** — Пользователю предлагается задать 5 вероятностей для пяти выходных портов (если их сумма не равна 1, то они нормализуются). Агент будет перенаправляться на тот или другой выходной порт, выбор которого будет случайно определяться в соответствии с заданными вероятностями.

• **Номер выхода** — Пользователь должен задать выражение, возвращающее целое число в диапазоне от одного до пяти. Когда агент попадает в этот блок, выражение вычисляется, и результат означает номер выходного порта, через который агент должен покинуть этот блок. Выражение может зависеть как от агента, так и от каких-то внешних факторов.

Поступивший агент покидает блок **SelectOutput5** в тот же момент времени.

Блок может использоваться для сортировки агентов согласно заданному критерию, для разделения потока агентов на части и т.д.

Иногда требуется иметь более пяти выходов. Используя блоки **SelectOutputIn** и **SelectOutputOut**, вы можете создать один большой блок **SelectOutput** с требуемым количеством выходов.

SelectOutput

Блок направляет входящих агентов в один из двух выходных портов в зависимости от выполнения заданного (детерминистического или заданного с помощью вероятностей) условия. Условие может зависеть как от агента, так и от каких-то внешних факторов. Поступивший агент покидает блок **SelectOutput** в тот же момент времени.

Может использоваться для сортировки агентов согласно заданному критерию, для случайного разделения потока агентов на части и т.д. Предположим, например, что в вашей модели моделируются клиенты (с помощью агентов типа Customer, у которого есть параметр vip типа boolean). Тогда если вы захотите направлять VIP клиентов в верхний порт (True), а всех остальных — в нижний (False), то вы должны задать условие agent.vip и выбрать тип Customer в качестве типа агента блока **SelectOutput**. Более сложный случай: вы хотите перенаправить в верхний порт блока только 80% VIP клиентов, а оставшиеся 20% (и всех остальных) — в нижний порт. Тогда условие будет выглядеть как agent.vip && randomTrue(0.8).

Иногда требуется иметь более двух выходов. Мы предоставляем вам два блока для направления агентов в разные отделы диаграммы процесса: блоки **SelectOutput** и **SelectOutput5**. Блок **SelectOutput5** имеет пять выходных портов, соответственно, он может направлять агентов в пять выходов. Используя блоки **SelectOutputIn** и **SelectOutputOut**, вы можете создать один большой блок **SelectOutput** с требуемым количеством выходов.

Параметры

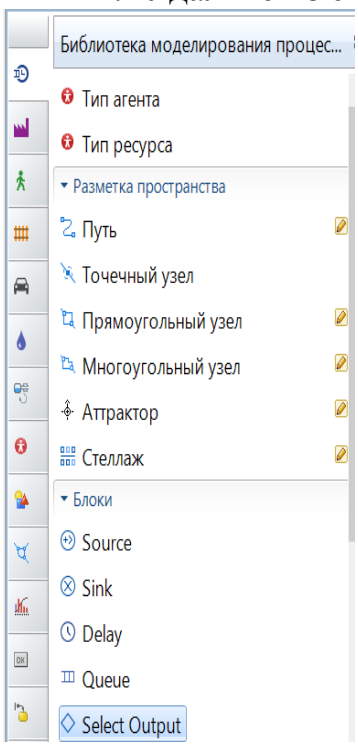
Выход true выбирается

Определяет, как будет производиться маршрутизация агентов: будут ли агенты направляться на выход true (верхний порт outT) случайно, с **Заданной вероятностью**, заданной в поле **Вероятность [0..1]** или же **При выполнении условия**, заданного в поле **Условие**.

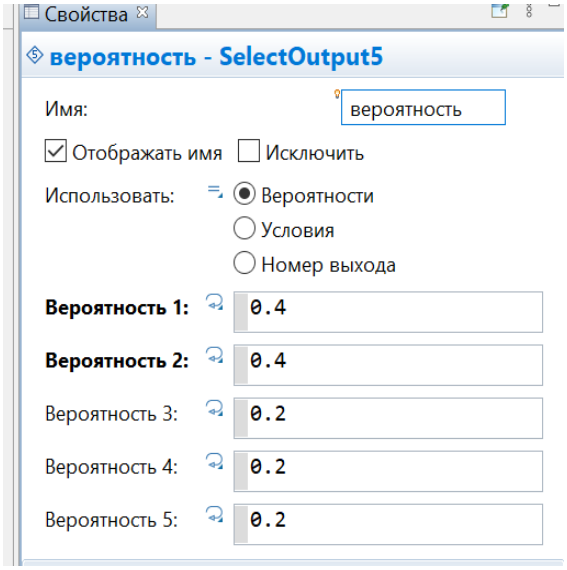
ХОД РАБОТЫ

Задание1

1. ЗАПУСТЕТЕ СОЗДАННУЮ ВАМИ МОДЕЛЬ
2. Удалите из схемы источник движения и все связи между ним и машинами.

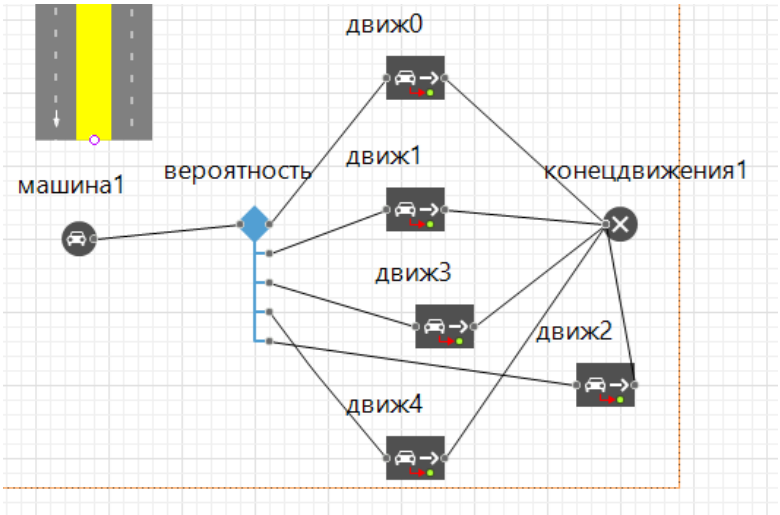


3. Перейдите в библиотек моделирования процессов и выберите элемент Select Output
4. После чего перетяните его на рабочую область
5. Укажите вероятности появления машин, общая сумма должна быть 1
6. Добавьте 5 шт блоков управляющих движением (carMoveTo) и переименуйте их.
7. Укажите по какой из дорог они будут ехать.
8. Свяжите их с элементом окончания движения

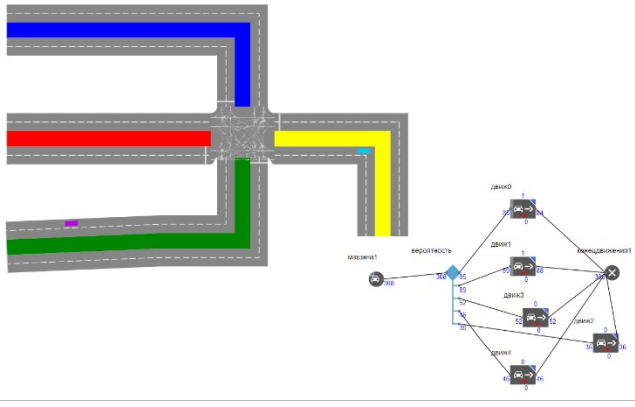


!для того чтобы подключить машины к элементу, нужно установить связь между элементом и входным портом элемента движения. Входной порт должен быть зеленым. Для того чтобы провести линию от машины к элементу конца движения машин, нужно два раза клацнуть правой кнопкой мыши на выходной порт и линия которая появится подвести к входному порту элемента.

9. Для установите направление проезд машины по дороге:

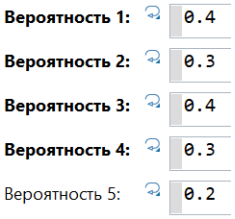


движ 0=дорога2
движ 1=дорога4
движ 2=дорога4
движ 3=дорога3
движ 4=дорога2



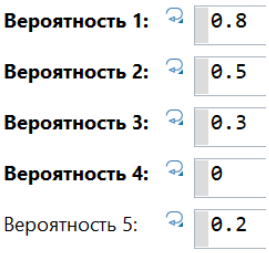
итогом у вас должна получится такая сеть:

задание 2



1. скопируйте созданную сеть дорожного движения, выделив всю сеть и вставив ее рядом с уже созданной.
2. задайте источнику движения (машине2 дорогу движения 4)
3. и укажите свойство встречного движения ей.
4. Установите интенсивность движения машин 500

задание 3



1. скопируйте созданную сеть дорожного движения, выделив всю сеть и вставив ее рядом с уже созданной.
2. задайте источнику движения (машине3 дорогу движения 2)
3. и укажите свойство встречного движения ей.
4. Измените вероятность распределения
5. Освободите порт 4,удалив связь и элемент движения, вероятность установите 0 этому порту.
6. Установите интенсивность движения машин 300

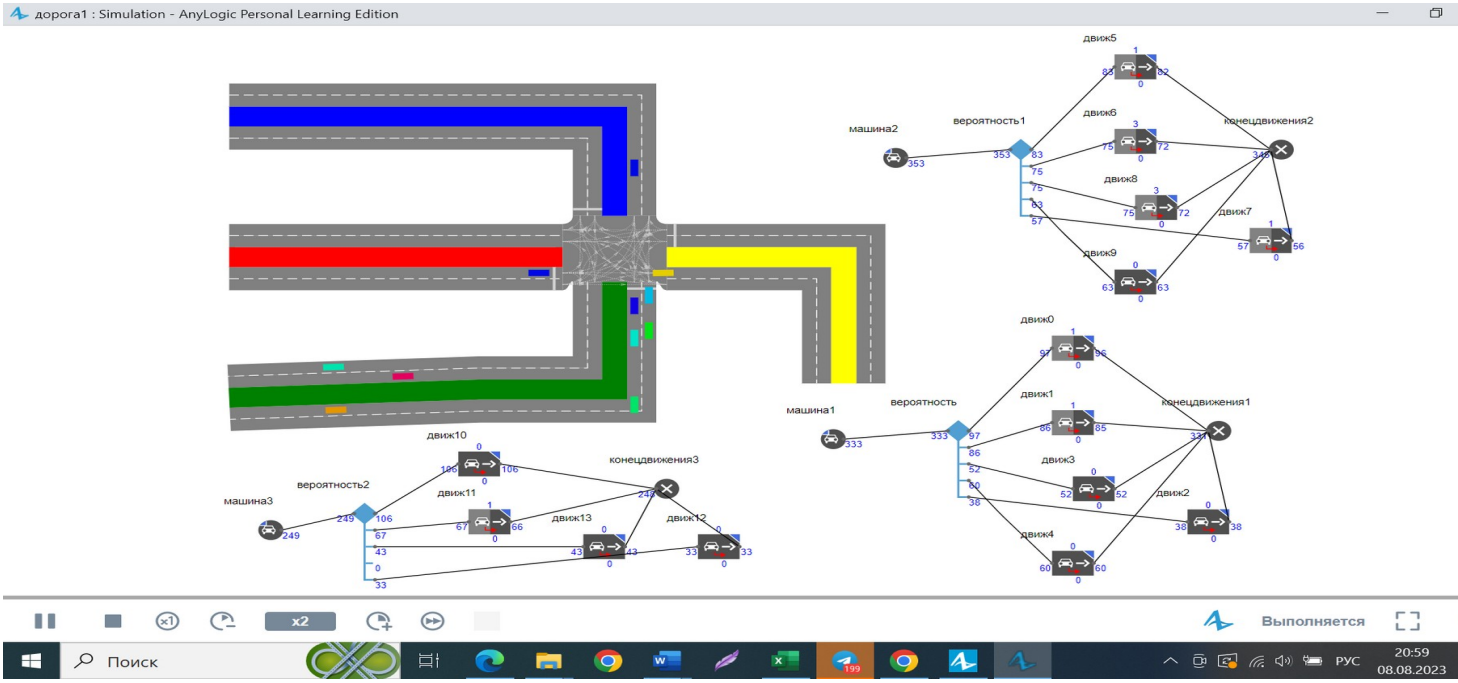
задание 4

Составьте дорожную карту используя элемент:

1. С двумя выходами **SelectOutput**,
2. Машину
3. Элемента движения 2шт
4. Элемента окончания поездки
5. Задайте ограничение на количество прибытий машин 100 шт
6. Установите направления движения и распределите вероятности 50 на 50.

Результат прикрепите и проанализируйте по какой дороге и из-за чего меньше всего проходит машин.

Результат трех заданий должен иметь такой вид:



Контрольные вопросы:

1. Как освободить порт элемента **SelectOutput?**
2. Можно ли оставить порт в элементе **SelectOutput5** открытым не присоединив к нему элемент движения, а только соединив их связью на прямую?

3. Чем отличается **SelectOutput5** от **SelectOutput?**
4. За что отвечает интенсивность в **CarMoveTo**?
5. Как скопировать всю сеть?

Содержание отчета:

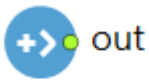
1. Тема, цель практической работы
2. Поэтапное описание выполнения практической работы
3. Скриншоты или результат практической
4. Краткие ответы на контрольные вопросы

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

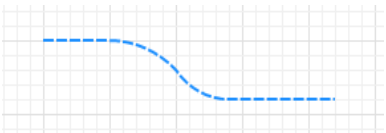
Уничтожает поступивших агентов. Обычно используется в качестве конечной точки потока агентов.

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.

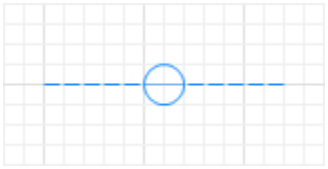
Узлы могут соединяться путями. Вместе они образуют сеть. Также AnyLogic автоматически создает отдельную сеть для каждого пути, не соединенного с узлами. В сети узел задает место, где агенты могут останавливаться, тогда как пути, соединяющие узлы, задают маршрут, по которому агенты следуют при движении 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

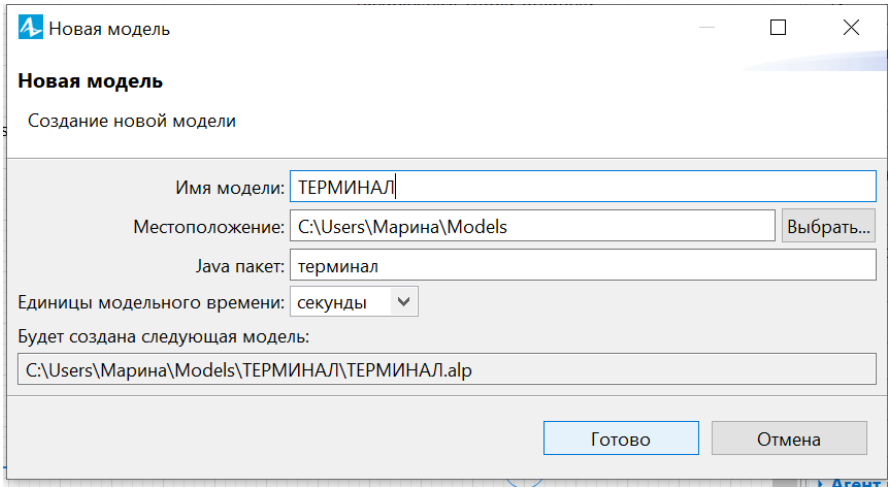
1. Перетащите элемент **Точечный узел** из палитры **Разметка Пространства** в графический редактор.

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**.

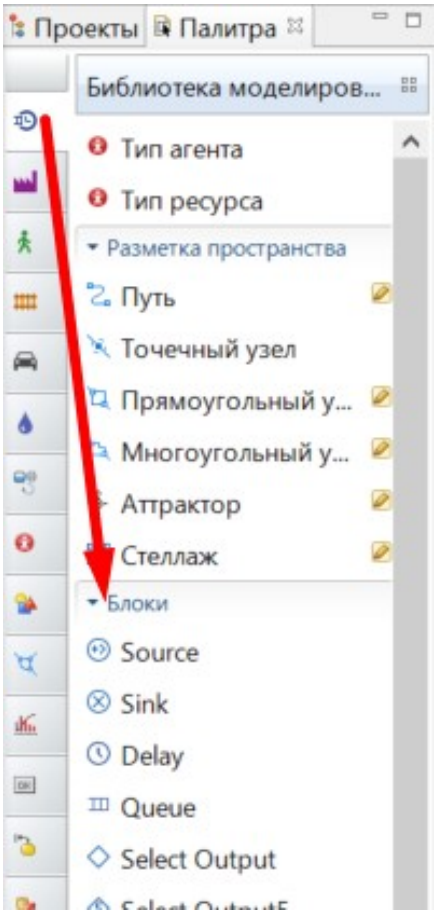
ХОД РАБРТЫ

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.

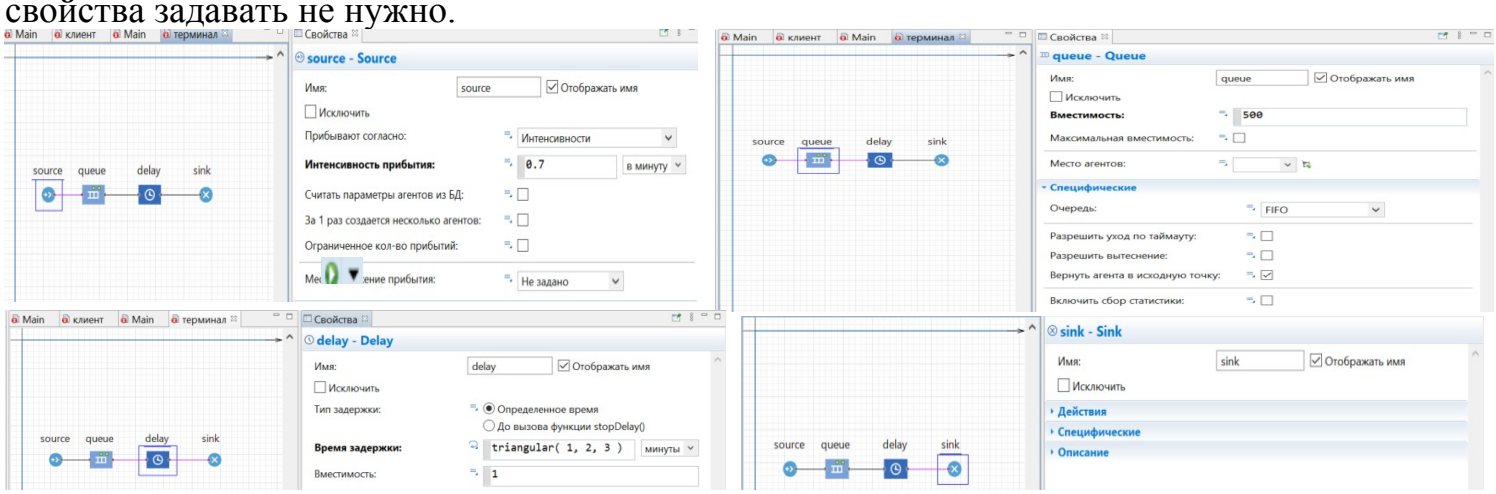
Для этого добавьте на рабочую область **терминал** из библиотеки моделирования процессов, элемент **source**, **queue, delay и sink** и соедините все элементы между собой.

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.

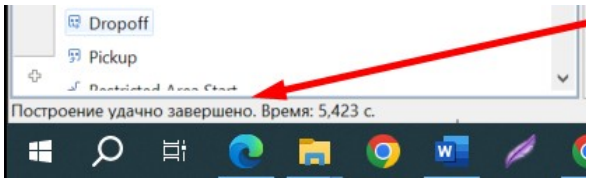
2.4 элементу уничтожения заявок (**sink**) клиента никакие свойства задавать не нужно.



П

е

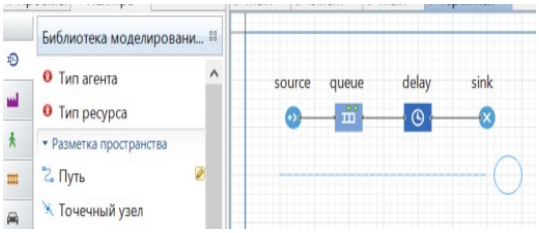
чего проверьте модель на ошибки. Для этого нажмите клавишу **F7,** для запуска модели нажмите клавишу **F5.** *Рекомендация перед запуском всегда проверять модель на ошибки! Либо можете нажимать в меню навигации*



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:

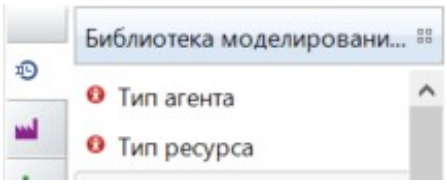


2.8.1 добавьте элемент **путь** и элемент **точечный узел (**банкомат наш**)**, перетащим их на рабочую область.

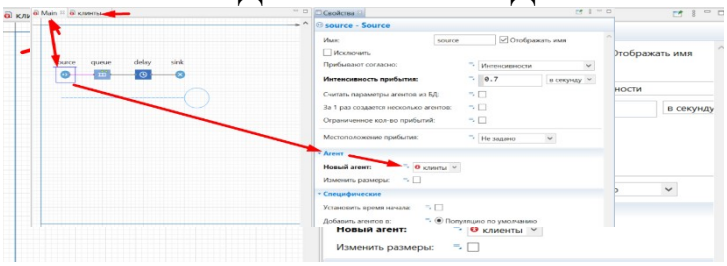
2.8.2 Нажмите на узел и в свойствах установите ему динамическое значение, это будет значить, что во время анимации цвет будет меняться с красного на зеленый. Напишите в поле такое значение:

```
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.**

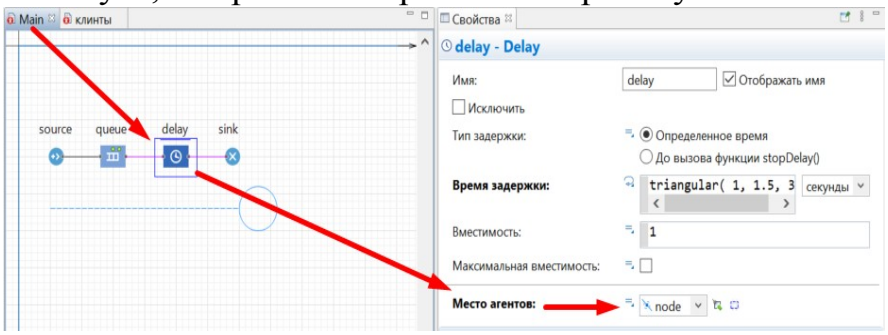


У ВАС ПОЯВИЛОСЬ ПУСТОЕ РАБОЧЕЕ ПОЛЕ, ДЛЯ ПРОДОЛЖЕНИЯ РАБОТЫ ПЕРЕЙДИТЕ НА ВКЛАДКУ main.



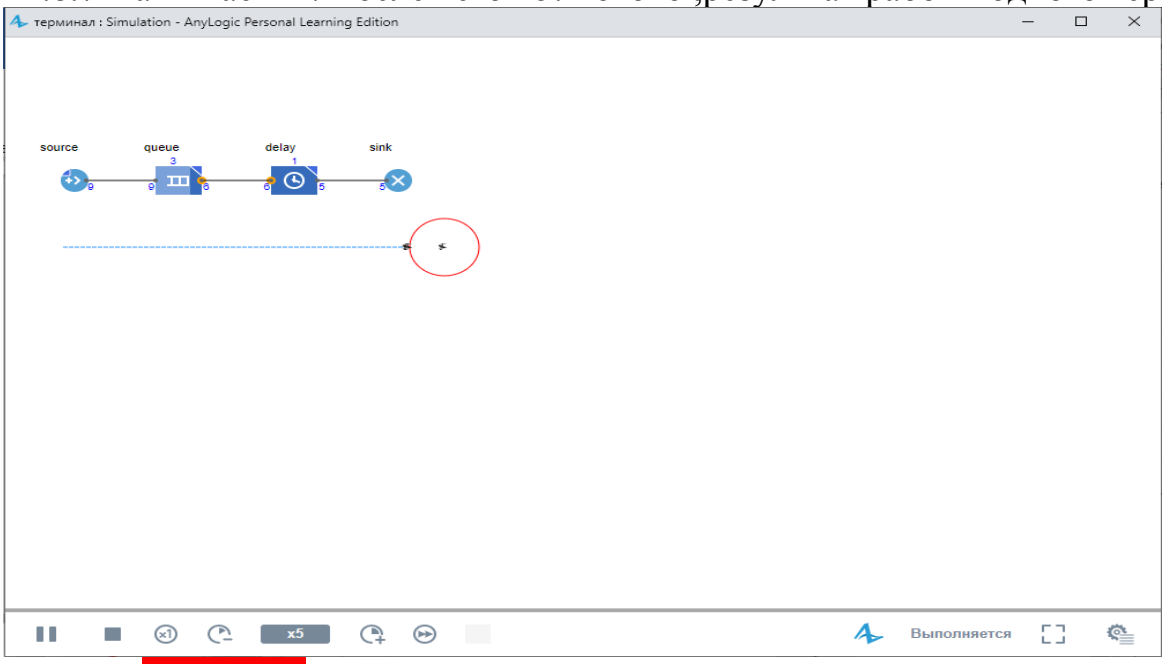
2.8.4 перейдите на вкладку **Main** нажмите на элесмент **sourse** , в свойствах выберите **Новый агент** и установите ему созданный тип агента, в нашем случае-это **клиенты** .После чего всей нашей схеме присвоится тип агента клинты.

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:

6. Когда срабатывает вытеснение? На какое свойство очереди оно опирается?
7. Что такое очередь? какие основные свойства знаете?
8. На что влияет элемент delay?
9. Назовите основные свойства элемента source.

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

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

Содержание отчета:

5. Тема, цель практической работы
6. Поэтапное описание выполнения практической работы
7. Скриншоты или результат практической
8. Краткие ответы на контрольные вопросы

Практическая 21-23

Тема: Имитационное моделирование. AnyLogic разработка типового проекта работы нескольких терминалов. Анализ проекта.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

SelectOutput

Блок направляет входящих агентов в один из двух выходных портов в зависимости от выполнения заданного (детерминистического или заданного с помощью вероятностей) условия. Условие может зависеть как от агента, так и от каких-то внешних факторов. Поступивший агент покидает блок **SelectOutput** в тот же момент времени.

Может использоваться для сортировки агентов согласно заданному критерию, для случайного разделения потока агентов на части и т.д. Предположим, например, что в вашей модели моделируются клиенты (с помощью агентов типа Customer, у которого есть параметр vip типа boolean). Тогда если вы захотите направлять VIP клиентов в верхний порт (True), а всех остальных — в нижний (False), то вы должны задать условие agent.vip и выбрать тип Customer в качестве типа агента блока **SelectOutput**. Более сложный случай: вы хотите перенаправить в верхний порт блока только 80% VIP клиентов, а оставшиеся 20% (и всех остальных) — в нижний порт. Тогда условие будет выглядеть как agent.vip && randomTrue(0.8).

Иногда требуется иметь более двух выходов. Мы предоставляем вам два блока для направления агентов в разные отделы диаграммы процесса: блоки **SelectOutput** и **SelectOutput5**. Блок **SelectOutput5** имеет пять выходных портов, соответственно, он может направлять агентов в пять выходов. Используя блоки **SelectOutputIn** и **SelectOutputOut**, вы можете создать один большой блок **SelectOutput** с требуемым количеством выходов.

Параметры

Выход true выбирается

Определяет, как будет производиться маршрутизация агентов: будут ли агенты направляться на выход true (верхний порт outT) случайно, с **Заданной вероятностью**, заданной в поле **Вероятность [0..1]** или же **При выполнении условия**, заданного в поле **Условие**.

еще

Вероятность

[Параметр виден, если **Выход true выбирается: Заданной вероятностью**] Выражение, вычисляющее вероятность того, что текущий агент покинет блок через порт outT. Значение вероятности должно лежать в пределах [0..1].

еще

Условие

[Параметр виден, если **Выход true выбирается: При выполнении условия**] Условие, вычисляемое для входящего агента. Если оно выполняется (равно true), то агент покидает блок через порт outT, если нет — через порт outF.

еще

Действия

Во всех этих действиях актуальный агент доступен как локальная переменная agent.

При входе Код, выполняемый, когда агент поступает в блок.

При выходе (true) Код, выполняемый, когда агент покидает блок через порт outT.

При выходе (false) Код, выполняемый, когда агент покидает блок через порт outF.

Порты

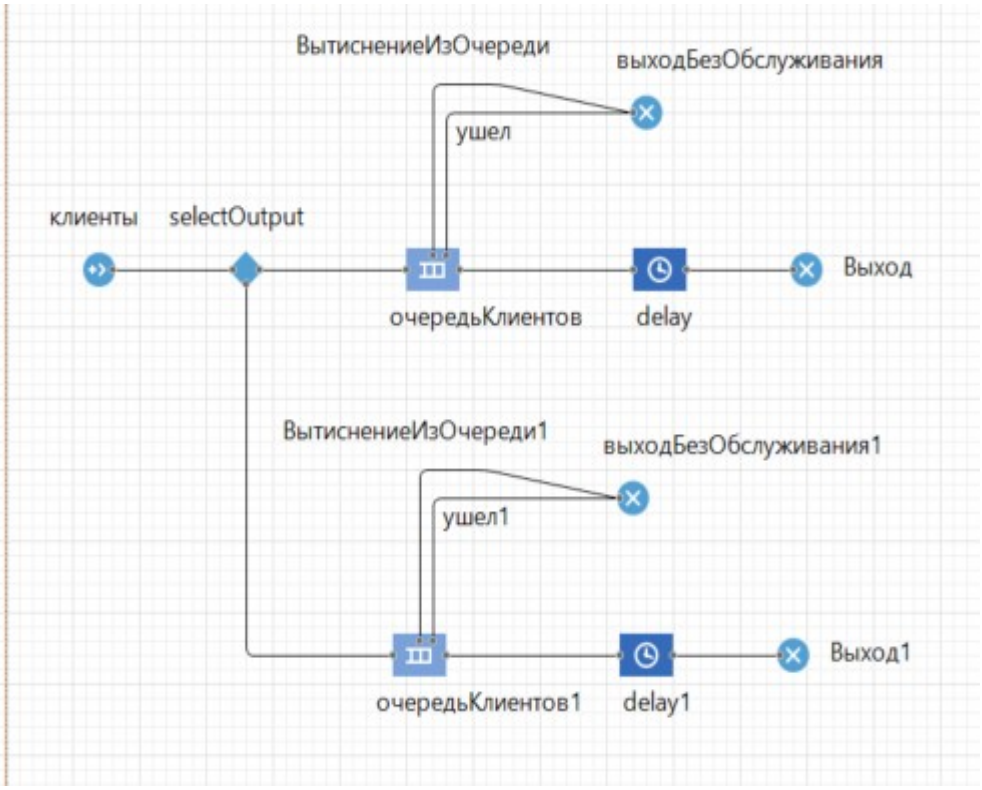
In Входной порт.

outT Выходной порт для агентов, для которых выбирается выход true.

outF Выходной порт для агентов, для которых выбирается выход false.

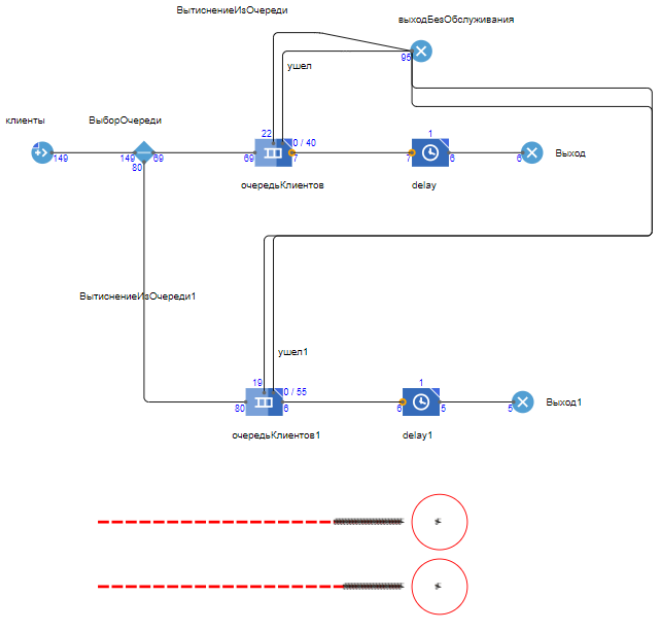
ХОД РАБОТЫ

Задание 1



1. Запустите проект терминал
2. Сдвиньте все элементы после очереди в право
3. Скопируйте и вставьте все эти элементы на схему
4. Добавьте элемент выбора **selectOutput** на схему, после элемента **source**, соедините схему линиями, если у вас разъединилась схема.
5. Соедините линией от элемента выбора к новой скопированной части вами схемы.
6. У вас должны получиться две очереди к терминалам.

7.переименуйте **selectOutput** на **выборОчереди** и установите равнозначную вероятность выбора ей.



8. Удалите из второй очереди элемент выходБезобслуживани1 и присоедините выходные данные двух портов второй очереди к ВыходБезОбслуживания.
9. Запустите схему, проанализируйте результат.
10. **скопируйте путь и точечный узел**, после чего вставьте на рабочую область и **измените** путь и точечный узел под **ОчередьКлиентов1** и **delay1.**

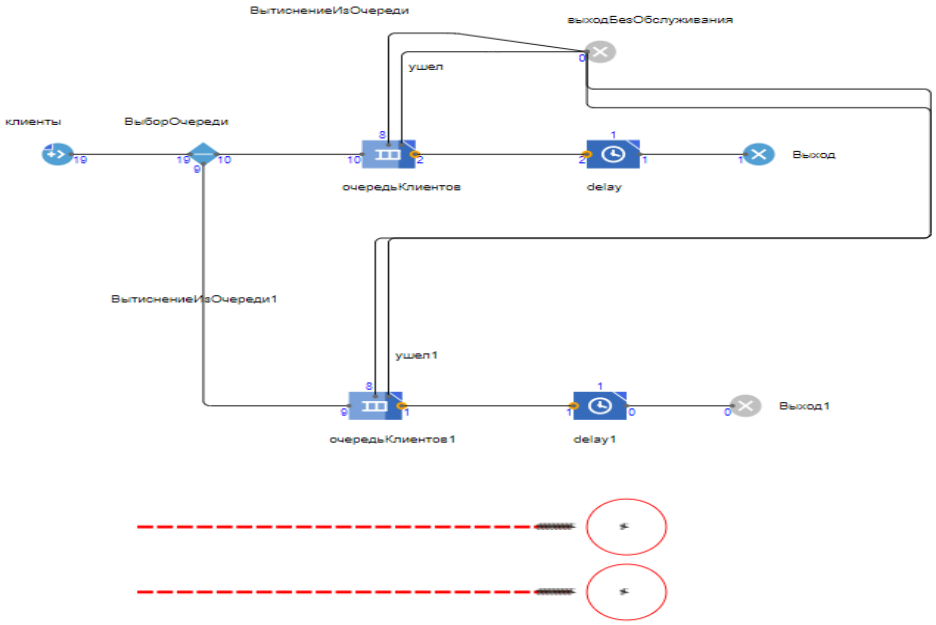
Результат вашей схемы проанализируйте при разных вариантах ускорения, с разными входными параметрами интенсивности и загрузки обработки заявок, сравните результаты. Сделайте вывод по вашему анализу.

Должно быть минимум 6 вариантов схем с разными параметрами входных данных.

Задание 2

1. Выберите элемент **Select Output (**Выбор очереди**),** после чего перейдите в его свойства, выберите 

и введите в поле условие, такое значение: `очередьКлиентов1.size()>очередьКлиентов.size()` , после чего запустите свою схему и сделайте вывод по распределению клиентов.



2. Измените условие так, чтобы распределение загрузки терминалов работало, таким образом, что если число очереди в первом терминале больше чем во втором, клиенты идут во второй терминал.
3. Дополним наше условие выбором очереди с вероятностью 50%, если наши очереди равны между собой по количество загрузки, то клиент пойдет в любую очередь, для этого допишем наше условие таким кусочком кода: `||`

```
((очередьКлиентов1.size()==очередьКлиентов.size())&&(randomTrue(0.5)))
```

Задание 3

1. **Откройте старый проект и выполните следующие действия**
2. Добавьте еще одну очередь с источником задержки и обработки запроса и выходом из очереди
3. Присоедините вытесненных и ушедших клиентов из очереди к ВыходуБезОбслуживания
4. Измените оператор выбора с двух условий оператор с 5 выходами (**Select Output5**).
5. Установите свойство **использовать вероятности** укажите распределения по 0.3 между терминалами с учетом не рабочих портов им мы ставим 0 вероятность.
6. Соедините схему логически так, чтобы она работала.
7. Добавьте 3ю визуальною очередь и точечный узел, подключите к 3й очереди.
8. Запустите схему и сделайте ее анализ на протяжении 10 секунд реального времени
9. Результат проанализировать и сделать вывод
10. Перейти на свойство **использовать Условия** и задать равнозначное распределениею.

10.1 условий 4, так как система считает 5му порту достаётся остаток, который не поделен между данными портами.

10.2 *Для того, чтобы указать, как порт не используем пишем в нем* параметр **False**

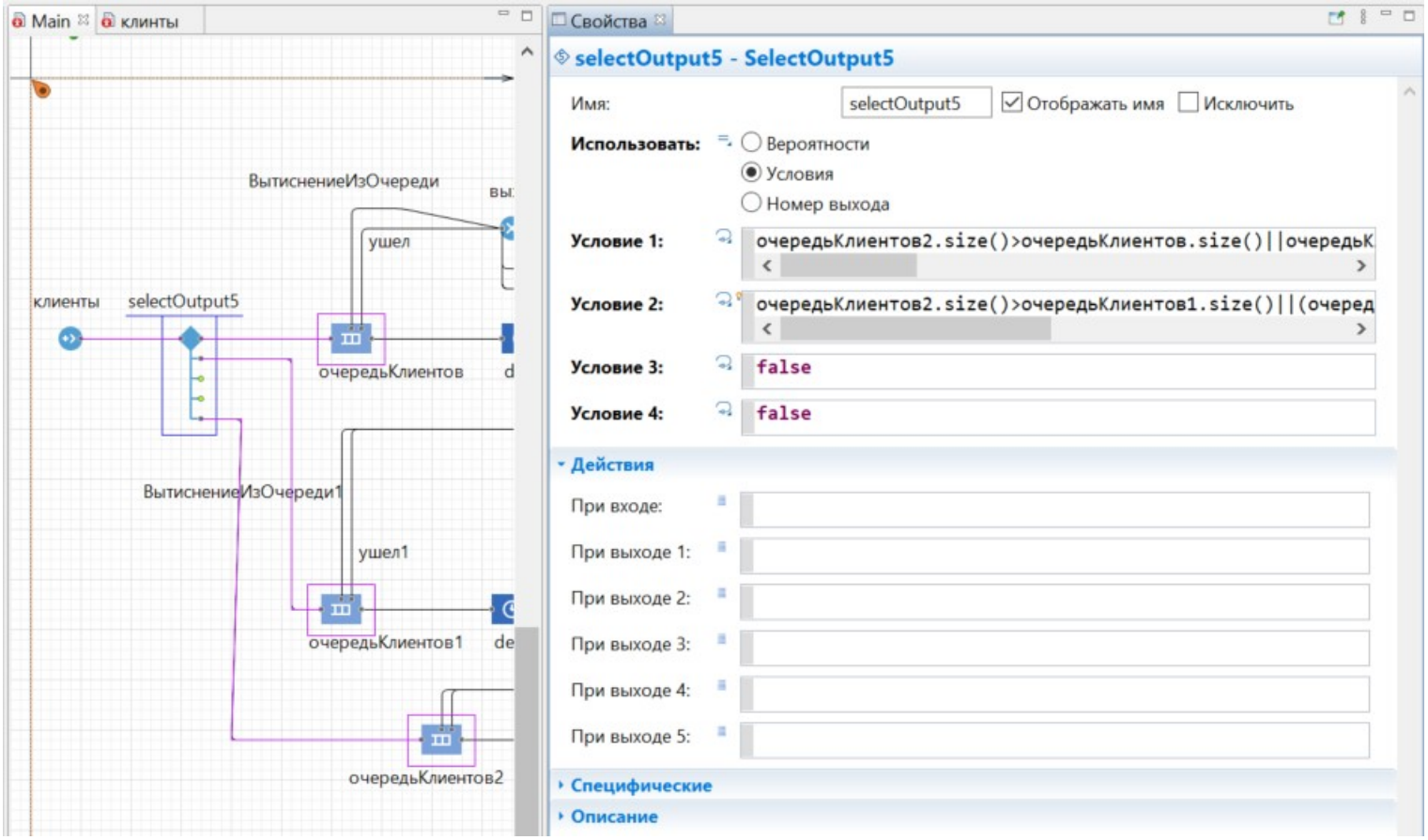
11. **Ведите данное условие в соответствующие порты, не используемым портам поставьте значение False :**

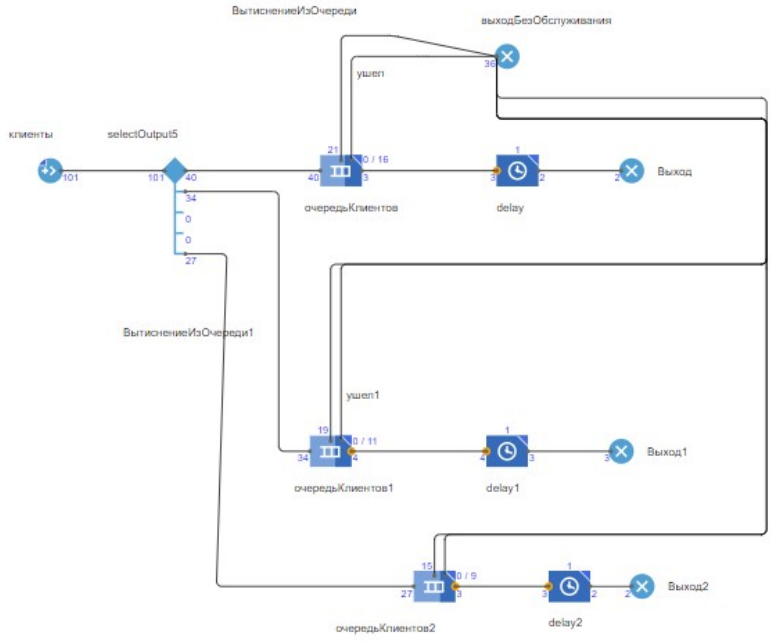
Порт 1

```
очередьКлиентов2.size()>очередьКлиентов.size()||
очередьКлиентов1.size()>очередьКлиентов.size()||
(очередьКлиентов2.size()==очередьКлиентов.size()&&(randomTrue(1/3)))||
(очередьКлиентов1.size()==очередьКлиентов.size()&&(randomTrue(1/3)))
```

Порт 2

```
очередьКлиентов2.size()>очередьКлиентов1.size()||
(очередьКлиентов2.size()==очередьКлиентов1.size()&&(randomTrue(1/3)))
```



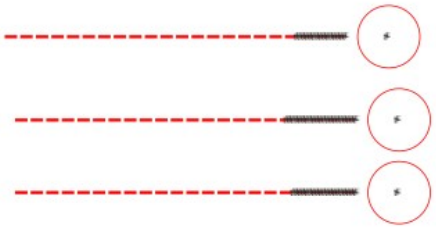


Контрольные вопросы:

1. От чего зависит распределение вероятностей между портами?
2. Как указать неиспользуемые порты в свойствах с использованием условий?
3. По какой причине в свойствах портов с распределением условных вероятностей нет 5-го порта? Можно ли установить вероятность всем портам по 1? Если можно, то это будет логически допустимо или нет? Если нет, то как лучше распределить числовую вероятность?

Содержание отчета:

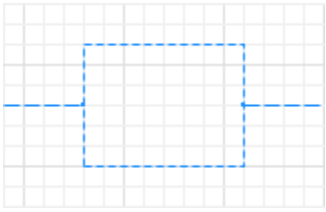
9. Тема, цель практической работы
10. Поэтапное описание выполнения практической работы
11. Скриншоты или результат практической
12. Краткие ответы на контрольные вопросы



Практическая 24-26

Тема: Имитационное моделирование. AnyLogic разработка типового проекта работы нескольких терминалов с участием нескольких менеджеров и разгрузки ресурсов.. Анализ проекта.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ



Прямоугольный узел

Узлы и пути являются элементами разметки пространства, которые задают местоположение агентов в моделируемом пространстве:

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

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.

Элемент разметки **Прямоугольный узел** задает область прямоугольной формы.

Существует еще два вида узлов, которые вы можете использовать:

Точечный узел. Используйте его для рисования узла транспортировки и переходов в сети. Точечный узел создается автоматически при соединении двух путей.

Многоугольный узел. Используйте его, чтобы нарисовать узел сложной формы.

Чтобы задать конкретные узлы ожидания внутри прямоугольного или многоугольного узла, используйте аттракторы.

Service

Захватывает для агента заданное количество ресурсов, задерживает их, а затем освобождает захваченные им ресурсы. Эквивалентен последовательности блоков **Seize**, **Delay**, **Release** (и сам реализован именно таким способом) и должен использоваться в тех случаях, когда все, что требуется — это задержать захваченные ресурсы на заданное время, а затем их отпустить. Большинство параметров этих вложенных блоков вынесены в интерфейс блока **Service**.

Один набор ресурсов может выполнять задачи с разными приоритетами, заданные в нескольких блоках.

- Если задачи имеют одинаковый приоритет, они будут выполняться последовательно в соответствии с настройками времени выполнения каждой из них. Если освободившихся единиц ресурса хватает, то такие задачи могут выполняться одновременно. Если у двух задач одинаковый приоритет, но одна из них по какой-то причине была ранее приостановлена, то выбирается приостановленная задача.[1]
- Если для задач не задан механизм вытеснения, они будут выполняться последовательно в соответствии с настройками времени выполнения каждой из них. Если освободившихся единиц ресурса хватает, то такие задачи могут выполняться одновременно.
- Начало выполнения задачи, заданной в одном блоке, не сбрасывает приоритеты задач, заданных в других блоках.

Существует несколько способов указать ресурсы, которые следует захватить в блоке **Service**. Более подробную информацию об этом вы можете найти в статье про использование ресурсов.

Параметры

Захватить

Здесь вы можете выбрать, требуются ли для сервиса **ресурсы одного типа** или ресурсы разных типов **((альтернативный) набор ресурсов)**. Детальное описание см. в статье про использование ресурсов.

Набор(ы) ресурсов

[Параметр виден, если **Захватить: (Альтернативный) набор ресурсов**]
Здесь вы можете задать требуемые наборы ресурсов (блоки **ResourcePool**). Вы можете добавить несколько наборов с помощью кнопки **Добавить список**. Ресурсы набираются согласно их доступности. Детальное описание см. в статье про использование ресурсов.

Тип ресурсов

[Параметр виден, если **Захватить: Ресурсы одного типа**]
Блок **ResourcePool**, задающий ресурсы, которые требуются для обслуживания агента. Детальное описание см. в статье про использование ресурсов.

Количество ресурсов

[Параметр виден, если **Захватить: Ресурсы одного типа**]

Выражение, возвращающее требуемое количество ресурсов для агента.

Вместимость очереди

[Параметр виден, если не выбрана опция **Максимальная вместимость**]

Вместимость вложенной очереди queue.

Максимальная вместимость

Если опция выбрана (true), то вместимость очереди queue будет максимально возможной (ограничена константой Integer.MAX_VALUE).

Время задержки

Выражение, вычисляющее время задержки для агента.

Пересылать захваченные ресурсы

Если опция выбрана (true), захваченные ресурсы будут пересылаться в указанное местоположение.

Место назначения

[Параметр виден, если выбрана опция **Пересылать захваченные ресурсы**]

Задает место, куда будут пересылаться ресурсы. Ресурсы могут быть отправлены в следующее место назначения:

К агенту — ресурсы пересылаются в местоположение указанного агента

Узел сети — ресурсы пересылаются в указанный узел сети

Аттрактор — ресурсы пересылаются в указанный аттрактор

Узел

[Параметр виден, если **Место назначения: Узел сети**]

Узел сети, куда отправляются агенты, созданные этим блоком.

Аттрактор

[Параметр виден, если **Место назначения: Аттрактор**]

Аттрактор, куда отправляются агенты, созданные этим блоком.

По окончании, движущиеся ресурсы

Выберите, движущиеся ресурсы **Возвращаются в базовую точку** (если их сразу же не захватывает другой агент) или **Остаются на месте**.

Место агентов (queue)

Фигура разметки (узел или путь), где располагаются агенты, пока они находятся во вложенном блоке queue.

Место агентов (delay)

Фигура разметки (узел или путь), где располагаются агенты, пока они находятся во вложенном блоке delay.

ResourcePool

Задает набор ресурсов, которые могут захватываться и освобождаться агентами с помощью блоков **Seize**, **Release**, **Assembler** и **Service**.

Типы ресурсов

Ресурсы могут быть трех типов:

- **Статические** ресурсы привязаны к определенному местоположению (например, узлу) внутри сети и не могут быть перемещаться или быть перемещены. Примером статического ресурса может быть рентгеновский кабинет или весы-платформа.
- **Движущиеся** ресурсы могут перемещаться сами по себе, они могут представлять персонал, транспорт, и т.д.
- **Переносные** ресурсы могут быть перемещены агентами или движущимися ресурсами. Переносное ультразвуковое устройство или кресло-каталка могут быть примерами переносных ресурсов.

Движущиеся и переносные ресурсы имеют базовое местоположение, куда они могут при необходимости вернуться или быть возвращены. Ресурсы одного типа могут иметь индивидуальные свойства, отображаться на презентации, хранить статистику своего использования и т.д. Вы можете создать свои собственные типы ресурсов. Агент использует имя типа, чтобы ссылаться на ресурсы, и может выбирать конкретный ресурс, анализируя его свойства.

Любой ресурс может быть либо **свободен**, либо **занят**. Блок собирает статистику занятости ресурсов (непрерывная статистика процента занятых ресурсов от их общего числа). Отдельные ресурсы собирают индивидуальную статистику загруженности. Также вы можете собрать статистику времени, которое отдельный ресурс потратил на обслуживание, перерывы, поломки и дополнительные задачи.

Движущиеся ресурсы не служат препятствиями друг для друга, поэтому их движение не в полной мере соответствует реальности. Если при создании модели вы хотите учитывать возможные столкновения и разрешение навигационных конфликтов, более подходящим инструментом для этого будут транспортеры либо двигающиеся по заданному пути, либо с произвольной навигацией.

Блоки библиотеки, которые работают с ресурсами

Библиотека Моделирования Процесса поддерживает разнообразные операции с ресурсами: агент может захватить один или несколько ресурсов (блок **Seize**), освободить ресурсы (блок **Release**), отправить захваченные ресурсы в определенное местоположение (блок **ResourceSendTo**), прикреплять ресурсы так, чтобы они перемещались вместе с агентом (блок **ResourceAttach**), и отсоединять прикрепленные ресурсы (блок **ResourceDetach**). Совершая операции над множеством ресурсов, укажите их список в параметре **ResourcePool**; например, чтобы захватить двух медсестер и один рентгеновский кабинет, укажите: Nurse, Nurse, XRay.

Если запросы от блоков **Seize** и **Service** не могут быть удовлетворены в текущий момент времени, эти запросы помещаются в очередь блока **ResourcePool**. Эта очередь может быть либо обычной очередью FIFO, либо учитывать приоритеты запросов.

Один набор ресурсов может выполнять задачи с разными приоритетами, заданные в нескольких блоках.

- Если задачи имеют одинаковый приоритет, они будут выполняться последовательно в соответствии с настройками времени выполнения каждой из них. Если освободившихся единиц ресурса хватает, то такие задачи могут выполняться одновременно. Если у двух задач одинаковый приоритет, но одна из них по какой-то причине была ранее приостановлена, то выбирается приостановленная задача.[1]
- Если для задач не задан механизм вытеснения, они будут выполняться последовательно в соответствии с настройками времени выполнения каждой из них. Если освободившихся единиц ресурса хватает, то такие задачи могут выполняться одновременно.
- Начало выполнения задачи, заданной в одном блоке, не сбрасывает приоритеты задач, заданных в других блоках.

[1] Каждый ресурс получает запросы из разных источников. Когда у ресурса есть несколько задач с одинаковым приоритетом (новых или ранее приостановленных), при выборе задачи к исполнению учитываются следующие правила:

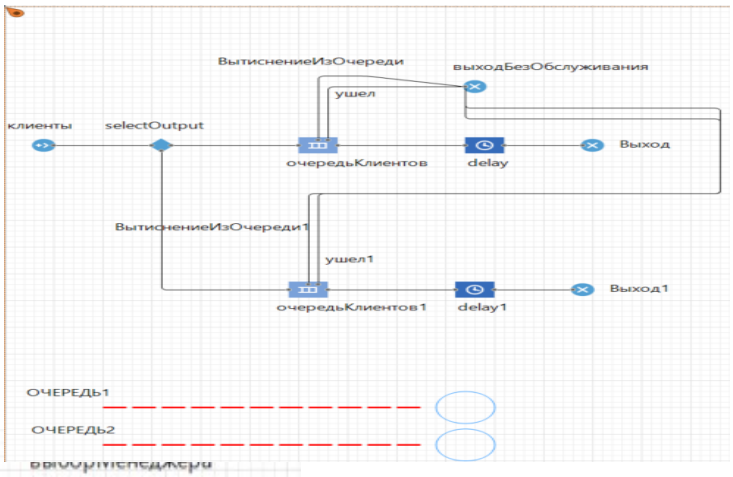
- Самый высокий приоритет — у приостановленных задач из «индивидуальной» очереди единицы ресурса.
  Такие очереди формируются агентами, находящимися в блоках, для которых значение свойства **Правило вытеснения задач** установлено на **Ожидать оригинал ресурса**.
- После этого обрабатываются прерванные задачи из очереди блока **ResourcePool**. Эта очередь формируется агентами, находящимися в блоках, для которых значение свойства **Правило вытеснения задач** установлено на **Захватывать любой ресурс**.

После выполнения задач из этих очередей ресурс сначала выполняет задачи из «индивидуальной очереди», а затем — из очереди блока **ResourcePool**.

Если этот порядок выполнения вам не подходит, вы можете вручную уменьшить приоритет определенной задачи с помощью кода: для этого используйте действие блока **При остановке задачи**.

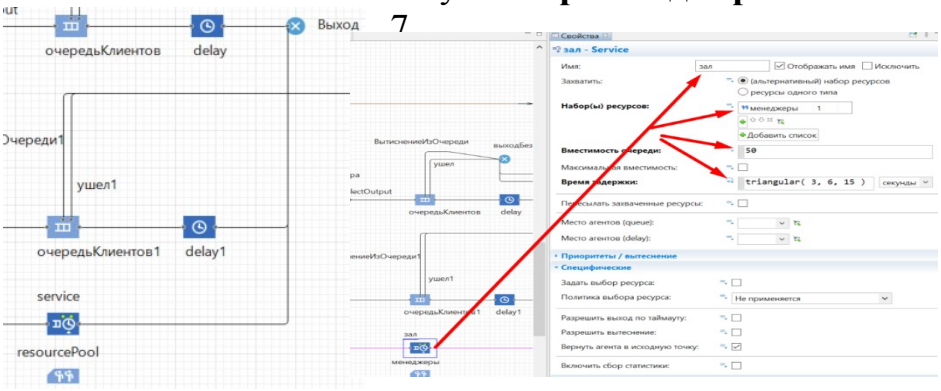
ХОД РАБОТЫ

Задание1

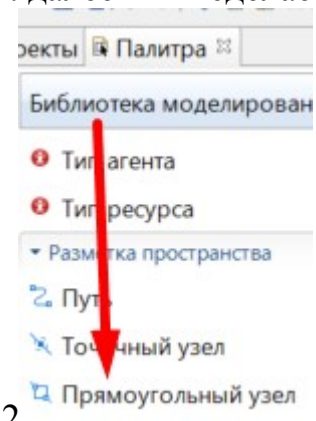


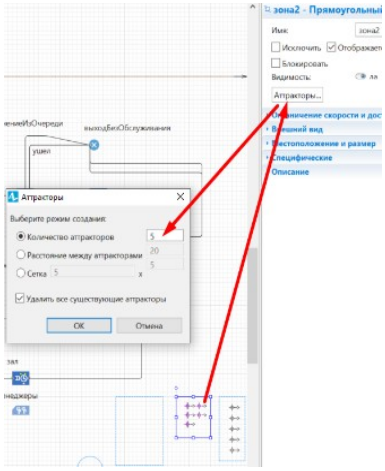
1. **запустите ваш проект, который я вам говорила сохранить под новым названием** или
2. удалите элемент выбора с 5 портами и добавьте элемент с двумя портами, как делали ранее.
3. Соедините элемент ветвления с очередями, путям и точечным узлам укажите очереди и элементы delay
4. Установите равнозначное распределение элементу.

5. **Добавьте еще один элемент ветвления с двумя выходами выборМнеджера.** Создаем для того, чтобы клиенты, те которые приходили, могли выбрать пойти к менеджеру или сразу к терминалу.
6. Установим элементу **ВыборМенеджера** в свойствах **вероятность 0.6.**



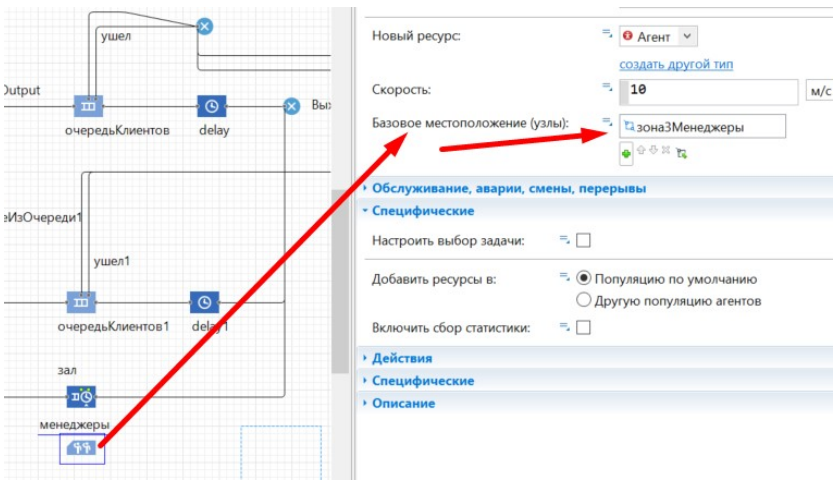
7. Добавим новый элемент **service** Service **,** он вам поможет распределить ресурсы с разными задержками по времени.
Добавим элемент **resourcePool** Resource Pool ,задает ресурсы определённого типа (движущиеся, статичные, переносные), соединим его с элементом и выход соединим с элементом Выход.
9. Выход 1, удалите элемент и связь от dilay1 и установите связь на Выход
10. **resourcePool** переименуем в **Менеджеры**, количество в свойствах данного элемента укажем **5**
11. **service** переименуем в **зал** и в свойствах данного элемента зададим **Наборы ресурсов-Менеджеры, вместимость очереди** укажем **50, время задержки укажите** `triangular(36,15)`
12. далее сделаем анимацию для данных элементов:



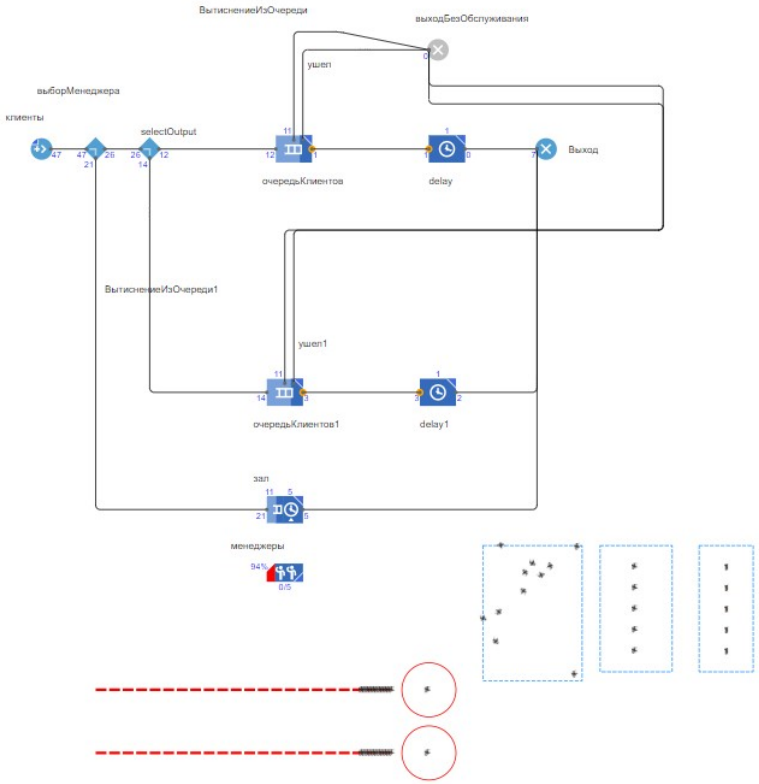


добавим элемент **прямоугольный узел** Многоугольный узел (это зоны различные), мы создадим 3 зоны, а значит добавьте 3 элемента.
Переименуйте их на Зона1, Зона2, Зона3.
У данного элемента, есть свойство Аттракторы, они создают место положение агентов.
Установите по 5 Аттракторы зоне2 и зоне3
12.1 Аттракторы из зоны 3 поверните в левую сторону (так как мы создаём имитацию общения клиентов и менеджеров, а они общаются лицом к лицу.) Для этого, нажмите на каждый элемент внутри зала3 и стрелку разверните в лево.
12.5 Переименуйте **зона1** в **зону1Ожидание**, **зона2**-в **зона2Обслуживание и зона3** в **зону3Менеджеры**, для более понятного поиска элементов из списка, для переименование используем CTRl+Enter

12.6 Выберите элемент **Зал** и в свойствах установите ему: место агентов очередь -Зона1Ожидание, место агентов заявки – зона2Обслуживание



12.7 Далее нажмите элементменеджеры и перейдите в его свойтва и укажите Базовое местоположение(узлы-Зона3Менеджеры.)
12.8 Добавьте новый тип ресурса на схему и назовите его **персонал**,нажмите далее из списка



картинок выберите **Служащий** и нажмите далее,**готово**

12.9 **Перейдите на основную рабочую**

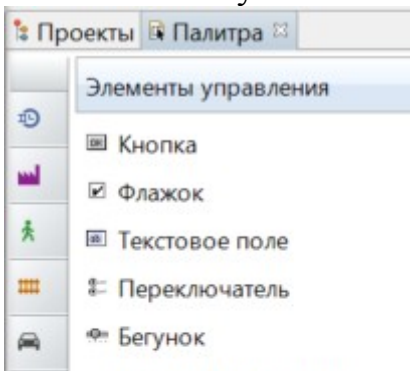
область

12.10 Нажмите на элемент **менеджеры** и в его свойствах **новый ресурс** укажите из списка **персонал**

12.11 Результат**.**

Задаине2

1. **Добавьте** новый элемент **Бегунок** из **библиотеке элементов управления**
2. В свойствах элементА **Бегунок** нажмите галочку (связать с) и укажите **менеджеры,** максимальное значение установите **6, добавьте метки,** данный элемент позволит в системе увеличивать и уменьшать количество менеджеров.

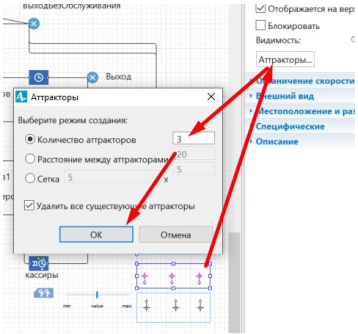


3. ЗАПУСТИТЕ МОДЕЛЬ И ПРОКОМЕНТИРУЙТЕ МОДЕЛЬ, СДЕЛАЙТЕ ПОЛНЫЙ АНАЛИЗ И ВЫАОДЫ ПО МОДЕЛИ, РЕЗУЛЬТАТЫ ЗАПИШИТЕ.
4. При анализе попробуйте изменять входные параметры интенсивности ресурсов и их обработки
5. Выводы и результаты разных входных параметров запиши в вывод и прикрепите скриншоты.

Задание 3

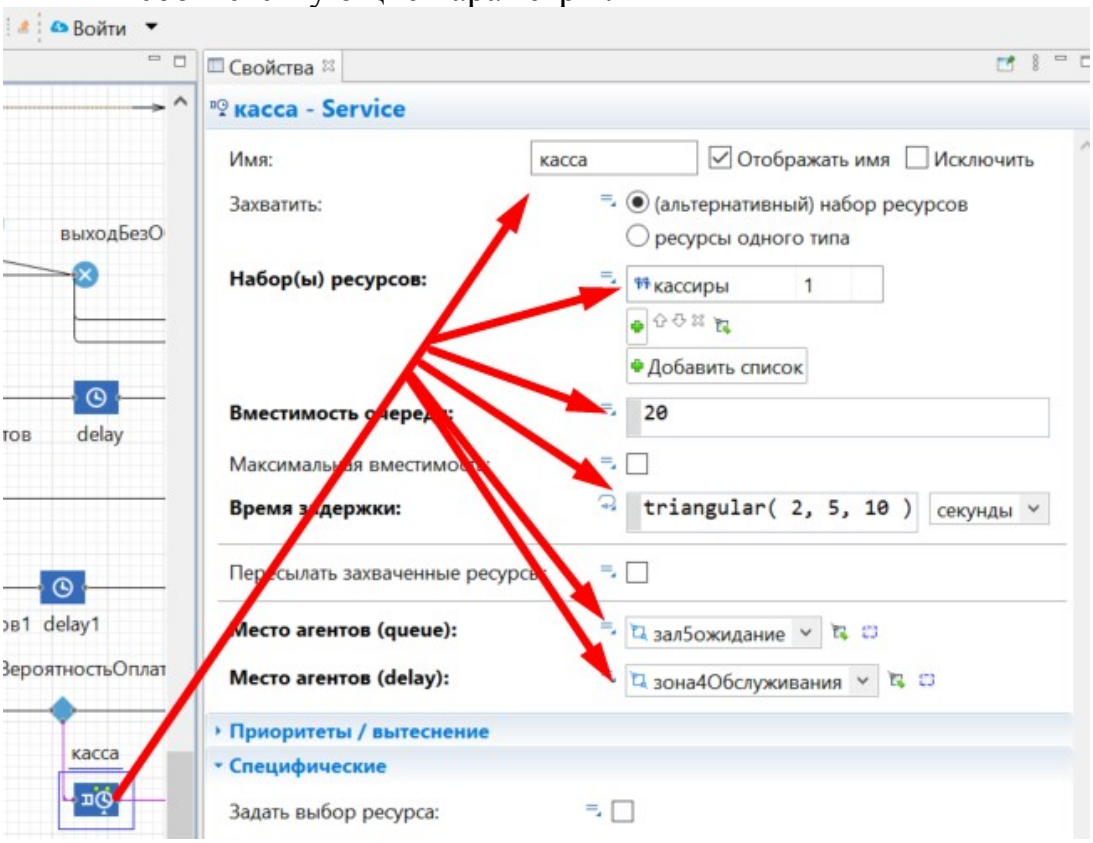
Создать еще один узел обслуживания клиентов, по типу кассиров.

1. Для этого добавьте еже один элемент сервис **service** на рабочую область
   1.1 Назовите элемент **service** Service **касса**
2. Добавьте на рабочую область еще один элемент **resourcePool** Resource Pool
   2.1 переименуйте элемент **resourcePool** на **кассиры.**
   2.1 установите в свойствах элемента **кассиры** количество ресурса **3**
3. Добавьте элемент выбора с двумя выходами **selectOutput**
   3.1 задайте элементу **selectOutput вероятность** оплаты на кассе **30%**
   3.2 свяжите один выход из элемента **selectOutput** напрямую с **Выход** схемы напрямую.
   3.3 Второй выход из элемента **selectOutput** свяжите с добавленным вашим элементом **кассиры** и выход уже из элемента кассиры присоедините к **Выход** из схемы.

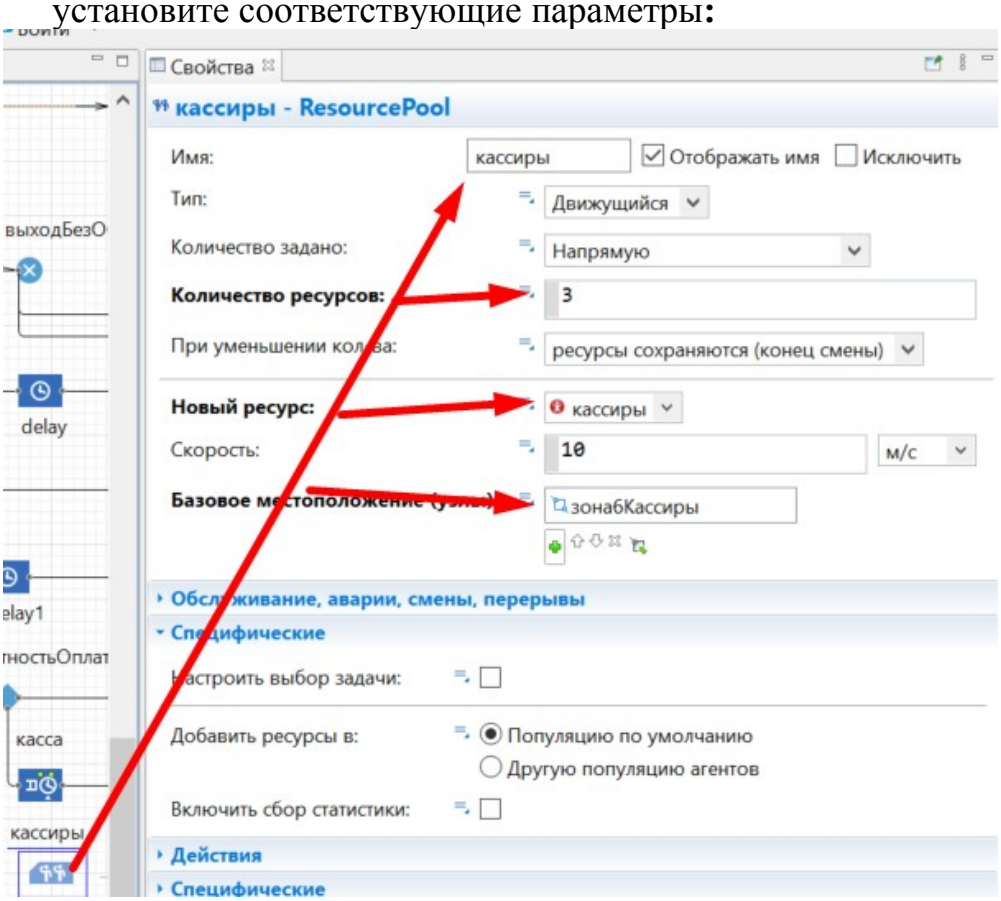


4. Добавьте 3 элемента **прямоугольный узел** Многоугольный узел на рабочую область
   4.1 двум из них добавьте по 3 **аттрактора** в каждую зону
   4.2 переименуйте каждую зону логическим названием

5. добавьте на рабочую область новый **тип ресурса** Тип ресурса , дайте название **Кассиры**.
6. Нажмите элемент **service** Service **касса** перейдите в его свойства и установите соответствующие параметры**:**



7. Нажмите элемент **resourcePool** Resource Pool **кассиры** перейдите в его свойства и установите соответствующие параметры**:**

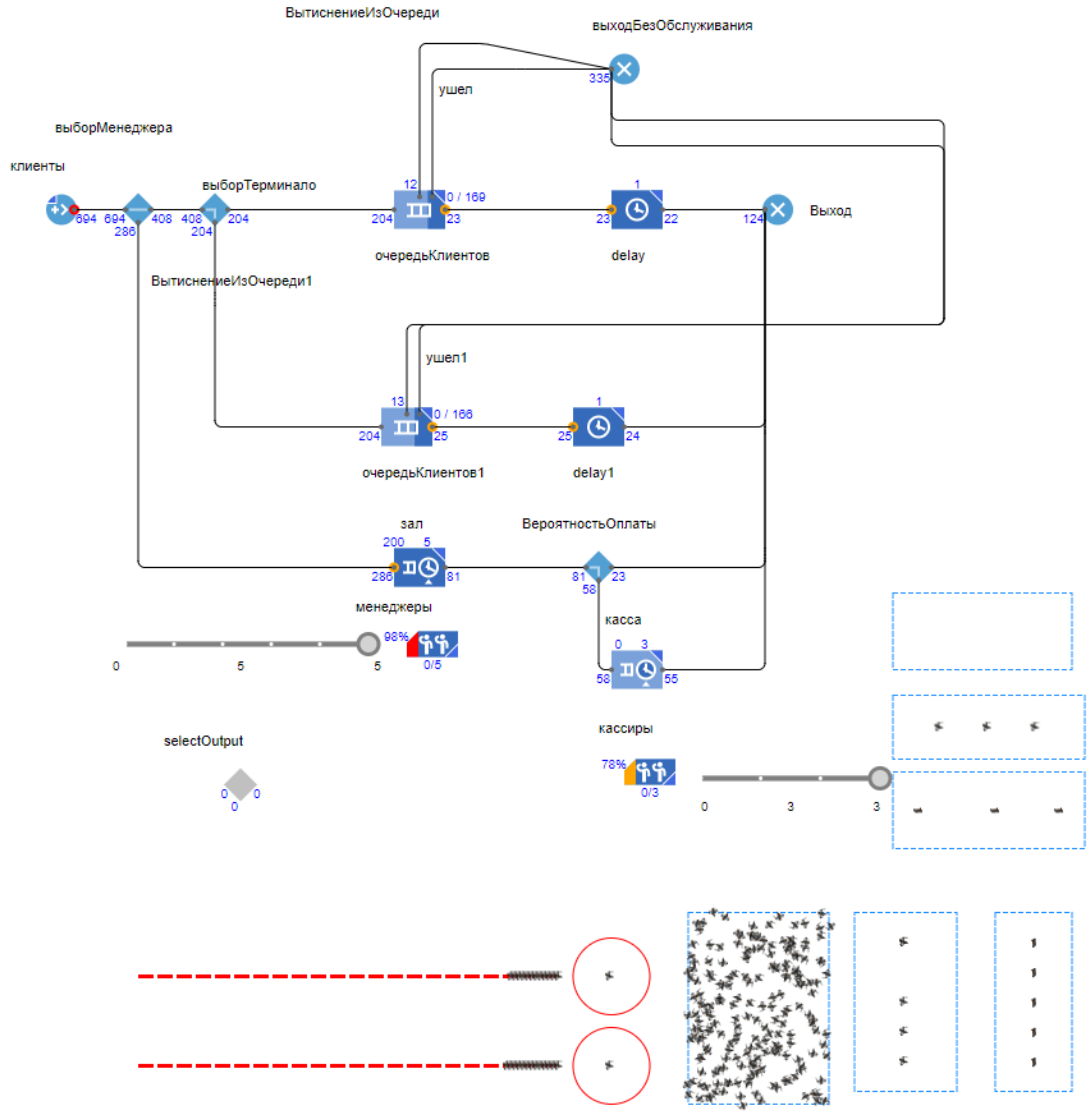


8. Добавьте элемент **бегунок** на рабочую область и установите в свойствах данного элемента

   8.1 связь с **кассирами**

8.2 максимальное значение **3**

9. **запустите схему и проанализируйте результат, выводы и свои наблюдения запишите в отчет иллюстрируя скринами.**
10. *При запуске у вас будет ошибка на каком-то моменте работы, ошибка связана будет с ограниченностью вместимости очереди, измените вместимость элемента Менеджеры и Кассиры на 500 и 200.*
11. *Запустите и продолжите анализ схемы.*
результат должен выглядеть таким образом



Контрольные вопросы:

1. Чем отличается тип ресурса от тип агента?
2. Что такое многоугольный узел для чего его применяют?
3. Для чего нужен элемент resource Pool?
4. Для чего нужен элемент Service?
5. Как указать элементу resource Pool 5 attractor ?
6. Назовите основные свойства элемента resource Pool, Service, многоугольный узел
7. Для чего нужен бегунок? На что он указывает в системе?

Содержание отчета:

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

Практическая 27-31

Тема: Работа с кодом на языке программирования PHP. Анализ кода. На основе кода построение: диаграммы последовательности, диаграммы состояния, use-case диаграммы, бизнес-модели в нотации IDEF3 и диаграммы потоков в нотации DFD.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Диаграмма прецедентов или диаграмма вариантов использования (англ. use case diagram) в UML — диаграмма, отражающая отношения между акторами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

Диаграмма состояний -она показывает, как объект переходит из одного состояния в другое. Диаграммы состояний служат для моделирования динамических аспектов системы. Данная диаграмма полезна при моделировании жизненного цикла объекта. От других диаграмм диаграмма состояний отличается тем, что описывает процесс изменения состояний только одного экземпляра определенного класса - одного объекта, причем объекта реактивного, то есть объекта, поведение которого характеризуется его реакцией на внешние события.

Диаграмма последовательности используются для уточнения диаграмм прецедентов, более детального описания логики сценариев использования. Это отличное средство документирования проекта с точки зрения сценариев использования! Диаграммы последовательностей обычно содержат **объекты**, которые **взаимодействуют в рамках сценария**, **сообщения**, которыми они обмениваются, и **возвращаемые результаты**, связанные с сообщениями. Впрочем, часто возвращаемые результаты обозначают лишь в том случае, если это не очевидно из контекста.

В отличие от IDEF0, представляющего моделируемую систему как совокупность видов деятельности, IDEF3 представляет собой технику моделирования деятельности как последовательности событий, а также участвующих в этих событиях объектов. Модели IDEF3 могут использоваться для детализации функциональных блоков IDEF0, они более низкого уровня . IDEF3 удобен для подробного моделирования деятельности отдельных подразделений, сотрудников, описания техпроцессов и т.д.

DFD (data flow diagram) — диаграмма потоков данных, один из основных инструментов структурного анализа и проектирования информационных систем, существовавших еще до широкого распространения UML DFD диаграммы в отличии от других нотаций позволяют визуально показать все процессы с точки зрения данных. Это может быть полезно:

- при разработке информационной системы;
- при интеграции системы;
- при миграции данных и функционала с одной системы на другую;
- в проектах, связанных с Data Management;
- в процессе построения аналитического хранилища, BI-решения.

Диаграмма позволяет визуализировать как движение данных между объектами системы, так и преобразования данных, которые могут применяться на разных шагах процесса.

ХОД РАБОТЫ

Ссылка на где можно онлайн запустить код и редактировать! PHP Sandbox - Execute PHP code online through your browser (onlinephp.io)

Код написанный в процедурном стиле!

```
<?php
$searchByAuthor = 'Marina2';//входящий поиск по автору 1

//проверка входящего запроса поиска 1.1
if ($searchByAuthor === '') {//если строка поиска пустая тогда вывести ошибку
    echo 'Ошибка, вы не ввели автора!';//вывод ошибки на экран 1.3
    return;//завершение выполнения поиска
}

//Начало имитации получения книг из базы данных 1.1.1
$databaseBooks = [];//инициализация массива
//добавление в массив книг по автору, где ключ - это автор, а значение книга. Связь 1 ко многим
$databaseBooks['Marina1'][] = 'Name Book 1';//Marina1 - автор, Name Book 1 - название книги
$databaseBooks['Marina2'][] = 'Name Book 2';
$databaseBooks['Marina2'][] = 'Name Book 3';
```

```
//Конец имитации получения кнги из базы данных

//Начало поиска автора 1.2
$booksByAuthor = [];//результат поиска книг по автора
foreach ($databaseBooks as $author => $book) {//поиск по нашей базе данной
    if ($author === $searchByAuthor) {//если поиск по автору совпадает с автором из базы данных то:
        $booksByAuthor[$author] = $book;
    }
}
//Конец поиска автора

//Начало создания вывода результата поиска 1.2.1
$searchResult = '';//инициализация строки результата поиска
foreach ($booksByAuthor as $author => $book) {//поиск по нашей базе данной
    $searchResult .= $author . ':' . implode(',', $book) . "\n"; //добавить в результат Автора и его книги
}
//Конец создания вывода результата поиска

echo $searchResult;//вывод результата поиска 1.2.1.1
```

Код написанный в стиле ООП!

```
<?php

$searchByAuthor = 'Marina3';

$databaseBook = getDatabaseBook();
$validationService = new ValidationService();
$mapperLibrary = new MapperLibrary();

$customerRequest = new CustomerRequest($searchByAuthor);
if (!$validationService->valid($customerRequest)) {
    echo 'Ошибка, вы не ввели автора!' . PHP_EOL;
    return;
}

$library = $mapperLibrary->getLibrary($databaseBook);
$searchBookService = new SearchBookService($library);
$bookListByAuthor = $searchBookService->searchByAuthor($customerRequest->author);

$bookFormatter = new BookFormatter();

echo $bookFormatter->format($bookListByAuthor);


class SearchBookService
{
    public Library $library;

    /**
     * @param Library $library
     */
    public function __construct(Library $library)
    {
        $this->library = $library;
    }

    /**
     * @param string $searchByAuthor
     * @return array<int, Book>
     */
    public function searchByAuthor(string $searchByAuthor): array
    {
        $resultSearchBooksByAuthor = [];
        foreach ($this->library->getBooks() as $book)
        {
            if ($book->getAuthor() == $searchByAuthor) {
                $resultSearchBooksByAuthor[] = $book;
            }
        }

        return $resultSearchBooksByAuthor;
    }
```

```
}

class BookFormatter
{
    /**
     * @param array<int, Book> $books
     * @return void
     */
    public function format(array $books): string
    {
        $output = '';

        foreach ($books as $book)
        {
            $output .= $book->getAuthor() . ':' . $book->getName() . PHP_EOL;
        }

        return $output;
    }
}

class CustomerRequest
{
    public string $author;

    /**
     * @param string $author
     */
    public function __construct(string $author)
    {
        $this->author = $author;
    }

    /**
     * @return string
     */
    public function getAuthor(): string
    {
        return $this->author;
    }
}

class ValidationService
{
    public function valid(CustomerRequest $customerRequest): bool
    {
        if ($customerRequest->getAuthor() === '') {
            return false;
        }

        return true;
    }
}

class MapperLibrary
{
    private array $books;

    public function getLibrary(array $books): Library
    {
        $booksObj = [];
        foreach ($books as $book) {
            $booksObj[] = new Book($book['author'], $book['nameBook']);
        }

        return new Library($booksObj);
    }
}

class Library
{
    /** @var array<int, Book> */
    private array $books;

    /**
     * @param Book[] $books
```

```
     */
    public function __construct(array $books)
    {
        $this->books = $books;
    }

    /**
     * @return array
     */
    public function getBooks(): array
    {
        return $this->books;
    }
}

class Book
{
    private string $author;
    private string $name;

    /**
     * @param string $author
     * @param string $name
     */
    public function __construct(string $author, string $name)
    {
        $this->author = $author;
        $this->name = $name;
    }

    /**
     * @return string
     */
    public function getAuthor(): string
    {
        return $this->author;
    }

    /**
     * @return string
     */
    public function getName(): string
    {
        return $this->name;
    }
}

function getDatabaseBook(): array
{
    $databaseBook = [];
    $databaseBook[] = ['author' => 'Marina1', 'nameBook' => 'Name Book 1'];
    $databaseBook[] = ['author' => 'Marina2', 'nameBook' => 'Name Book 2'];
    $databaseBook[] = ['author' => 'Marina3', 'nameBook' => 'Name Book 3'];
    $databaseBook[] = ['author' => 'Marina3', 'nameBook' => 'Name Book 3_2'];
    $databaseBook[] = ['author' => 'Marina3', 'nameBook' => 'Name Book 3_3'];
    $databaseBook[] = ['author' => 'Marina4', 'nameBook' => 'Name Book 4'];
    $databaseBook[] = ['author' => 'Marina5', 'nameBook' => 'Name Book 5'];
    $databaseBook[] = ['author' => 'Marina6', 'nameBook' => 'Name Book 6'];

    return $databaseBook;
}
```

Задание1

1. выберите один из вариантов предложенного кода, идея кода одинаковая, только написана в разных стилях!
2. Откройте онлайн редактор, вставьте код, запустите его и проверите результат.
3. Сделайте анализ кода и по коду напишите техническое задание на разработку, чтобы оно соответствовало написанному коду.
4. Результат задания должно быть написанное ТЗ

Задание2

По коду и ТЗ составьте диаграмму последовательности.

Задание3

По коду и ТЗ составьте диаграмму состояния

Задание4

По коду и ТЗ составьте use-case диаграмму

Задание5

По коду и ТЗ составьте бизнес-модель в нотации IDEF3

Задание6

По коду и ТЗ составьте диаграмму потоков в нотации DFD.

P.S. для тех кто не смог составить самостоятельно, но пролистал методичку до конца)!
ТЗ:
есть два сценария:
1. Основной
2. Альтернативный
1.Основной: Клиент вводит ФИО автора и нажимает кнопку поиска. После чего система проверяет параметры ввода пользователя, если валидация пройдена- система ищет Автора по каталогам, когда поиск окончен - система выдает результат
2. Если клиент не ввел имя автора, но нажал на кнопку поиска, то система выводит сообщение об ошибке.

Контрольные вопросы:

1. Для чего нужна диаграмма последовательности?
2. Для чего нужна диаграмма состояний?
3. Для чего нужна диаграмма use-case?
4. Что показывает нотация бизнес-модели IDEF3?
5. Зачем нужна нотация DFD?
6. Чем отличается код ООП и код процедурный?

Результатом каждого задания должна быть схема, кроме задания 1.

Содержание отчета:

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