Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет информационных технологий и управления

Кафедра систем управления

В. А. Захарьев, С. В. Снисаренко

МОДЕЛИРОВАНИЕ В ПРОЕКТИРОВАНИИ СЛОЖНЫХ СИСТЕМ. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Рекомендовано УМО по образованию в области информатики и радиоэлектроники в качестве учебно-методического пособия для специальности 1-53 01 07 «Информационные технологии и управление в технических системах»

Минск БГУИР 2020

УДК [681.51-027.31]-047.58(076.5)
ББК 32.965.02я73
З-38

Рецензенты:

кафедра информационных систем и технологий Международного института дистанционного образования Белорусского национального технического университета (протокол №4 от 07.12.2019);

заведующий кафедрой информационных технологий Белорусского государственного университета кандидат физико-математических наук, доцент В. А. Нифагин

Захарьев, В. А.

З-38 Моделирование в проектировании сложных систем. Лабораторный практикум : учеб.-метод. пособие / В. А. Захарьев, С. В. Снисаренко. – Минск : БГУИР, 2020. – 64 с. : ил.

ISBN 978-985-543-582-3.

Содержит описание восьми лабораторных работ, которые выполняются в пакете моделирования GPSS World. Предназначено для студентов специальности 1-53 01 07 «Информационные технологии и управление в технических системах» при выполнении ими лабораторных работ по курсу «Моделирование в проектировании сложных систем».

УДК [681.51-027.31]-047.58(076.5)
ББК 32.965.02я73

ISBN 978-985-543-582-3

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Моделирование является основным методом исследований во всех областях знаний и научно обоснованным методом оценок характеристик сложных систем, используемым для принятия решений в различных сферах инженерной деятельности. Существующие и проектируемые системы можно эффективно исследовать с помощью математических моделей (аналитических и имитационных), реализуемых на современных ЭВМ, которые в этом случае выступают в качестве инструмента экспериментатора с моделью системы.

На стадии проектирования сложных технических систем обычно проводится их моделирование, позволяющее спрогнозировать поведение системы в определенных условиях, поскольку модель дает описание системы, отображающее совокупность ее свойств. Техническую систему можно описывать с различных точек зрения: функциональной, морфологической, информационной. Функциональное описание показывает изменение состояния системы во времени и определяет ее место по отношению к другим системам и к внешней среде. Морфологическое (топологическое) описание позволяет представить структуру системы, т. е. совокупность ее элементов и связей между ними. Информационное описание дает представление об организации системы, поскольку оно определяет зависимость ее морфологических и функциональных свойств от внутренней и внешней информации.

В настоящее время нельзя назвать область человеческой деятельности, в которой в той или иной степени не использовались бы методы моделирования. Особенно это относится к сфере управления различными системами, где основными являются процессы принятия решений на основе получаемой информации. Моделирование может быть определено как представление объекта моделью для получения информации об этом объекте путем проведения экспериментов с его моделью. Теория замещения одних объектов (оригиналов) другими объектами (моделями) и исследования свойств объектов на их моделях называется теорией моделирования. Если результаты моделирования подтверждаются и могут служить основой для прогнозирования процессов, протекающих в исследуемых объектах, то говорят, что модель адекватна объекту. При этом адекватность модели зависит от цели моделирования и принятых критериев.

Процесс моделирования предполагает наличие объекта исследования; исследователя, перед которым поставлена конкретная задача; модели, создаваемой для получения информации об объекте и необходимой для решения поставленной задачи. Причем по отношению к модели исследователь является, по сути дела, экспериментатором, только в данном случае эксперимент проводится не с реальным объектом, а с его моделью. Такой эксперимент для инженера есть инструмент непосредственного решения организационно-технических задач. Одна из проблем современной науки и техники – разработка и внедрение в практику проектирования новейших методов исследования характеристик сложных информационно-управляющих и информационно-вычислительных

систем различных уровней, например, автоматизированных систем научных исследований и комплексных испытаний, систем автоматизации проектирования, комплексов и сетей, информационных систем. При проектировании сложных систем и их подсистем возникают многочисленные задачи, требующие оценки количественных и качественных закономерностей процессов функционирования таких систем, проведения их структурного алгоритмического и параметрического синтеза.

Имитационное моделирование используется для структурного и параметрического синтеза технической системы с оптимизацией ее по важнейшим параметрам: производительность, надежность, экономическая эффективность. Имитационное моделирование технологических процессов и систем может проводиться с применением сетей Петри или теории систем массового обслуживания.

Лабораторный практикум включает в себя восемь лабораторных работ. В каждой лабораторной работе приводятся название работы, цель, краткие теоретические сведения, практические задания с набором вариантов исходных данных, примеры составления некоторых программ, требования к содержанию отчета и контрольные вопросы.

При подготовке к лабораторной работе необходимо изучить теоретический материал по имеющемуся в ЭУМКД [1] конспекту лекций, кратким теоретическим сведениям, представленным непосредственно в лабораторной работе, а также по приведенной литературе [2–4], и внимательно проработать примеры и задачи по соответствующей теме. После выполнения каждой лабораторной работы и оформления отчета проводится ее защита. При этом студент должен аргументированно ответить на все вопросы, относящиеся к выполнению работы и анализу полученных результатов. Студент допускается к экзамену после выполнения и защиты всех лабораторных работ.

Лабораторные работы выполняются в компьютерном классе на персональном компьютере с использованием пакета имитационного моделирования GPSS World.

Содержание отчета

Отчет выполняется в электронном виде каждым студентом индивидуально и после проверки студент допускается к защите лабораторной работы. В отчете должна содержаться следующая информация:

- выполненное задание согласно варианту;
- алгоритм программы;
- листинг программы;
- выходная статистика и ее анализ;
- выводы.

ЛАБОРАТОРНАЯ РАБОТА №1

СОЗДАНИЕ МОДЕЛЕЙ СИСТЕМ С ОДНОКАНАЛЬНЫМИ И МНОГОКАНАЛЬНЫМИ УСТРОЙСТВАМИ

Цель работы *– ознакомление со средой имитационного моделирования GPSS World, изучение базовых операторов языка, сбор и анализ статистики, оценка производительности одноканальных и многоканальных устройств.*

Теоретические сведения

Система имитационного моделирования GPSS в наибольшей степени подходит для моделирования реальных объектов, которые могут быть представлены в виде одного или нескольких узлов систем массового обслуживания (СМО). В языке моделирования GPSS имеются специальные средства для моделирования потоков заявок, одноканальных и многоканальных узлов СМО, очередей и т. п. Язык GPSS позволяет моделировать практически любые СМО: разомкнутые и замкнутые, одноканальные и многоканальные, с неограниченными очередями, отказами, ограничениями на очередь и др. Основные характеристики СМО (коэффициенты загрузки узлов, длины очередей и т. д.) автоматически определяются в процессе моделирования и выводятся в составе выходных данных модели. В то же время с помощью языка GPSS могут решаться задачи моделирования систем, для которых обычно не используется описание в виде СМО.

Работа языка GPSS основана на использовании метода Монте-Карло. В большинстве случаев операции метода Монте-Карло (обращения к генераторам случайных чисел, проверка условий и т. п.) выполняются в языке GPSS автоматически, т. е. они скрыты от пользователя. Однако при необходимости пользователь имеет возможность реализовать в программе на GPSS операции этого метода.

В данных лабораторных работах рассматривается применение одной из реализаций системы моделирования GPSS – система GPSS World.

Компиляция модели

По окончании подготовки текста модели необходимо выполнить его компиляцию, т. е. преобразование в машинные коды. Для этого используется команда «*COMMAND – CREATE SIMULATION*». Создается файл в машинных кодах. Его имя образуется автоматически на основе имени исходного файла (т. е. файла GPSS-модели); расширение – **.SIM*.

Если компиляция или моделирование прерываются из-за ошибок в модели, следует по выведенному сообщению определить допущенную ошибку, закрыть окно созданного **.SIM*-файла, перейти в окно модели, внести необходимые исправления и снова выполнить компиляцию модели.

Краткое описание объектов GPSS

Объект типа «Транзакт». Транзакты – это динамические объекты GPSS. Они создаются в определенных точках модели, продвигаются интерпретатором через блоки, а затем уничтожаются. Транзакты являются аналогами единиц-потоков в реальной системе.

Объект типа «Блок». Блоки языка GPSS выполняют соответствующие операции в модели. Блоки реализуют операции четырех основных типов: создание или уничтожение транзактов, изменение числового атрибута объекта, задержка транзактов на определенный период времени, изменение маршрута транзакта в модели.

Объект типа «Одноканальное устройство». Объект этого типа представляет собой оборудование, которое в данный момент времени может быть занято только одним транзактом.

Объект типа «Многоканальное устройство». Объекты этого типа представляют собой оборудование для параллельной обработки, которое может быть использовано несколькими транзактами одновременно.

Объект типа «Логический переключатель». Объект типа «ключ» может находиться в состоянии «включен», «выключен», «инвертирован». Состояние ключа может быть изменено некоторым транзактом, любой транзакт может использовать состояние ключа для выбора одного из двух возможных путей или ожидать момента изменения состояния ключа.

Объект типа «Арифметическая переменная». Арифметические переменные позволяют вычислять арифметические выражения, состоящие из стандартных числовых атрибутов (СЧА), которые описаны далее.

Объект типа «Булева переменная». Булевы переменные позволяют проверять в одном блоке одновременно несколько условий исходя из состояния объектов и значений их атрибутов.

Объект типа «Функция». Используя функции, программист может производить вычисление непрерывных или дискретных функциональных зависимостей между аргументом и значением функции.

Объект типа «Очередь». Объекты типа «Очередь» предназначены для сбора статистики об использовании оборудования транзактами.

Объект типа «Таблица». Таблица – объект, который определяет форму вывода статистической информации.

GPSS – язык интерпретируемого типа, он связан с пошаговым выполнением инструкций каждого отдельно взятого оператора. Программа на языке GPSS имеет блочную структуру, т. е. управление осуществляется с помощью инструкций, называемых блоками.

Формат инструкции GPSS

[<метка>] <операция> <операнды> [<;комментарий>]

Каждая инструкция языка записывается в отдельной строке и содержит следующие поля:

- поле метки, в котором записывается либо номер, либо идентификатор блока, представляющий собой алфавитно-цифровую последовательность длиной до 5 символов, начинающуюся с буквы (указание метки необязательно);

- поле операций, в котором указывается наименование действия над элементами языка, т. е. обозначение соответствующего блока;

- поле операндов, в котором записываются СЧА элементов; поле операндов состоит из подполей <A>, <B>, <C>, <D>, <E>, <F>, G>, содержимое которых отделяется друг от друга запятыми; если одно из подполей операндов необходимо опустить, пробел не ставится, вместо него ставится запятая.

Управление продолжительностью моделирования. Организация таймеров

Длительность моделирования в программе на GPSS можно задать двумя способами.

1. Определить в управляющей команде START количество транзактов, которые необходимо обработать в модели (этот способ используется для простых моделей, содержащих единственный процесс):

```
GENERATE…
<программа модели>
TERMINATE 1
START          100
```

2. С помощью *процесса-таймера* определить отрезок модельного времени, в течение которого должно осуществляться моделирование. Процесс-таймер должен быть единственным на всю модель, поэтому только в нем в блоке TERMINATE задается непустое поле операнда <A>. Во всех остальных процессах поле операнда <A> в блоке TERMINATE должно быть пустым:

```
; первый процесс
GENERATE…
<программа модели>
TERMINATE
…
; n-й процесс
GENERATE…
<программа модели>
TERMINATE
; процесс-таймер
GENERATE 480
TERMINATE      1
START          1
```

Процесс-таймер реализуется транзактом, который вводится в модель в начале моделирования, задерживается на 480 единиц модельного времени, затем выводится из модели, что приводит к завершению всех процессов модели.

Обработка результатов моделирования

По окончании моделирования создается файл-отчет с результатами моделирования. Его имя образуется автоматически на основе имени файла GPSS-модели; расширение – **.GPR*. Файл-отчет, созданный системой GPSS World, содержит информацию о различных объектах GPSS-модели (устройствах, очередях и т. п.). Кроме того, в файле-отчете содержатся некоторые внутренние данные о работе системы моделирования. Обычно следует сохранить этот файл (командой «*FILE – SAVE*»), а также скопировать его содержимое в окно текстового редактора Word для обработки и последующей печати. Сохранять файл в машинных кодах не требуется.

Основные блоки

В самом начале моделирования в GPSS-модели нет ни одного транзакта. В процессе моделирования транзакты входят в модель в определенные моменты времени, обрабатываются в модели и в конечном итоге покидают ее. В связи с этим над транзактами возможны следующие действия, которые реализуют соответствующие блоки:

- ввести транзакт в модель – блок GENERATE;
- вывести транзакт из модели – блок TERMINATE;
- задержать транзакт – блок ADVANCE;
- регистраторы очередей: блок QUEUE – войти в очередь, блок DEPART – выйти из очереди.

Блок генерации GENERATE

Генерирует транзакт и направляет его к следующему блоку программы, поэтому следующим за GENERATE должен быть блок, который всегда может принять транзакт.

Формат блока:

GENERATE <A>,<B>,<C>,<D>,<E>

Интервалы времени между транзактами, поступающими в модель, определяются содержимым полей операндов:

– <A> – среднее время между поступлениями транзактов в модель (по умолчанию равно нулю);

- <B> – модификатор времени;
- <C> – начальная задержка, т. е. момент времени появления в модели первого транзакта, по умолчанию начальная задержка равна нулю;
- <D> – общее число транзактов, которые должны быть введены в модель (по умолчанию, в модель вводится неограниченное число транзактов);
- <E> – приоритет транзакта (0…127), чем больше значение, тем выше приоритет транзакта (по умолчанию приоритет транзакта равен нулю).

Блок уничтожения транзакта TERMINATE

Формат блока:

TERMINATE <A>

Транзакты, попадающие в этот блок, выводятся из модели и больше не участвуют в процессе моделирования. В поле операнда <A> записывается либо целое число, либо ничего. Каждый раз, когда транзакт входит в блок TERMINATE, целое число, стоящее в поле операнда <A>, вычитается из счетчика завершений, который устанавливается управляющей командой START. Как только значение счетчика завершений обнулится, моделирование закончится. Например, конструкция

```
TERMINATE    1
START        100
```

обеспечивает такую длительность моделирования, при которой через программу модели пропускается 100 транзактов.

Если в поле операнда <A> ничего не указано, счетчик завершений не уменьшается и моделирование продолжается бесконечно.

Блок задержки движения транзакта ADVANCE

Формат блока:

```
ADVANCE      <A>,<B>
```

Задержка движения транзакта во времени имитируется при попадании транзакта в блок ADVANCE, для которого в полях операндов <A> и <B> указываются соответственно среднее время задержки и модификатор времени, использование которого аналогично блоку GENERATE. Например,

```
ADVANCE 9, 2
ADVANCE 2, FN$EXPON.
```

Одноканальное устройство

Используется для имитации любого элемента системы, функционирование которого во времени можно представить сменой двух состояний: «свободно» и «занято».

Устройство характеризуется следующим свойством: каждое устройство в любой момент времени может обслуживать только один транзакт. Если в процессе обслуживания появляется новый транзакт, его поведение определяется одним из трех вариантов:

- новый транзакт должен подождать своей очереди;
- новый транзакт должен направиться в другое место;
- если вновь пришедший транзакт имеет больший приоритет, устройство прерывает текущее обслуживание и начинает обслуживать новый транзакт.

Для работы с одноканальными устройствами используются следующие действия, которые реализуют соответствующие блоки:

- занять устройство – блок SEIZE;
- освободить устройство – блок RELEASE;
- прервать обслуживание на устройстве – блок PREEMPT;
- снять прерывание – блок RETURN.

Многоканальное устройство(МКУ)

Два или более обслуживающих устройства, работающих параллельно, могут моделироваться в GPSS двумя или более одноканальными устройствами. Обычно это необходимо, когда отдельные устройства являются разнородными, например, имеют различную интенсивность обслуживания.

Однако очень часто параллельно работающие устройства являются одинаковыми, и GPSS предоставляет для их моделирования объект, называемый многоканальным устройством (накопителем или памятью).

Количество устройств, которое моделируется каждым из накопителей, определяется пользователем. В этом смысле употребляют термин «емкость накопителя». Эта емкость заранее должна быть определена пользователем, чтобы интерпретатор знал, сколько устройств использует данный накопитель.

Все используемые в модели накопители должны быть заранее описаны, т. е. должно быть определено количество однотипных устройств, входящих в накопитель. Для этого используется оператор STORAGE, определяющий емкость накопителя.

Формат оператора задания емкости накопителя STORAGE:

<имя> STORAGE <A>

В операторе STORAGE содержатся следующие поля:

- поле метки – имя накопителя;
- поле операции – STORAGE;
- поле операнда <A> – емкость накопителя.

MEM STORAGE 100

Блоки ENTER (ВОЙТИ) и LEAVE (ВЫЙТИ). Использование МКУ аналогично использованию одноканального устройства. Элементом, который занимает и использует накопитель, является транзакт. При моделировании накопителя события происходят в следующем порядке:

1) транзакт ожидает своей очереди, если это необходимо;

2) транзакт занимает устройство;

3) устройство осуществляет обслуживание на протяжении некоторого интервала времени;

4) транзакт освобождает устройство.

Блоки – регистраторы очередей QUEUE и DEPART

Данные блоки обеспечивают в GPSS возможность автоматического сбора статистических данных, описывающих вынужденное ожидание, которое может происходить время от времени в различных точках модели.

Система моделирования GPSS обеспечивает возможность сбора статистики с помощью такого средства, как регистратор очереди.

При использовании регистратора очереди в тех точках модели, где число ресурсов ограничено, интерпретатор автоматически начинает собирать различную информацию об ожидании с помощью следующих СЧА:

1) число входов транзактов в очередь;

2) количество транзактов, которые фактически присоединились к очереди и сразу ее покинули, т. е. имели время ожидания, равное нулю;

3) максимальная длина очереди;

4) среднее число ожидавших транзактов;

5) среднее время ожидания тех транзактов, которым пришлось ждать.

В модели может быть несколько регистраторов очередей, различающихся именами. Правила присвоения имен те же, что и для устройств. Разработчик вносит регистратор очереди в модель с помощью пары взаимодополняющих блоков:

QUEUE <A>[,<B>]

DEPART <A>[,<B>]

При входе транзакта в блок QUEUE (СТАТЬ В ОЧЕРЕДЬ) выполняются четыре действия:

1) счетчик входов для данной очереди увеличивается на содержимое операнда <B>;

2) длина очереди (счетчик текущего содержимого) для данной очереди увеличивается на содержимое операнда <B>;

3) транзакт присоединяется к очереди с запоминанием ее имени и значения текущего модельного времени;

4) когда транзакт входит в блок QUEUE, то выполняется поиск очереди с именем, определенным операндом А. При необходимости очередь создается.

Блок QUEUE не поддерживает список членов очереди, он только добавляет единицы к длине очереди.

Использование регистратора очереди необязательно. С помощью него интерпретатор собирает лишь статистику об ожидании. Если же регистратор не используется, то статистика не собирается, но везде, где должна возникать очередь, она возникает. Ожидание является следствием состояния устройства, а не следствием использования регистратора. Если программист не планирует обработку статистических данных об очередях, то лучше не собирать статистику – это позволит уменьшить продолжительность моделирования.

Один и тот же транзакт может одновременно увеличить длину нескольких очередей.

Транзакт перестает быть элементом очереди только после того, как он переходит в блок DEPART (ПОКИНУТЬ ОЧЕРЕДЬ) соответствующей очереди. Когда это происходит, интерпретатор выполняет такие операции:

1) уменьшает длину соответствующей очереди на содержимое операнда <B>;

2) используя привязку к значению времени, определяет, является ли время, проведенное транзактом в очереди, нулевым; если да, то такой транзакт по определению является транзактом с нулевым пребыванием в очереди и одновременно изменяется счетчик нулевых вхождений;

3) ликвидирует «привязку» транзакта к очереди.

Практические задания

Задание 1. *Изготовление заданного количества деталей, моделирование таймера, использование очередей, списки событий, статистика.*

Базовые операторы: *generate, terminate, advance, seize, release, start, queue, depart.*

На прессе гибкого производственного модуля нужно изготовить *a* деталей. Заготовки к нему поступают через *b* минут. На изготовление одной детали уходит *c* минут. Время обработки детали неизменно.

Определить время, за которое будет изготовлено *a* деталей. Сделать вывод о загрузке пресса. Предложить варианты оптимизации работы, чтобы загрузка пресса составила более 90 %, но в то же время транзакты в модели не остались после завершения моделирования. Показать статистику повышения производительности. Задания выполняются согласно индивидуальным вариантам (таблица 1.1).

Выполнить данное задание, организовав работу пресса в течение одной смены. Предусмотреть статистику очереди. Определить среднюю и максимальную длину очереди, количество заготовок, которые сразу пресс начал обрабатывать, среднее время ожидания изготовления заготовки без учета заготовок, которые сразу попали на пресс. Оценить загрузку пресса и предложить способы повышения производительности труда.

Таблица 1.1

Вариант	*a*	*b*	*c*
1	50	7±3	5±2
2	70	5±2	4±2
3	100	8±2	6±2
4	80	9±1	7±3
5	75	3±1	4±1
6	40	4±1	6±2
7	30	7±3	6±1
8	150	5±3	7±2
9	200	5±2	4±1
10	120	5±1	6±2
11	60	8±3	5±1
12	35	3±1	5±2
13	90	6±2	5±3
14	110	3±2	5±1
15	130	10±3	6±2

Задание 2. *Моделирование одноканальных и многоканальных устройств.*

Базовые операторы: *seize, release, storage, enter, leave* .

В цех поступают заготовки через *a* минут. Вначале деталь обрабатывается на токарном станке в течение *b* минут. Далее деталь обрабатывается на фрезерном станке *c* минут и на шлифовальном станке *d* минут. Время перемещения между операциями составляет (1 ± 0,2) минут. Частота подачи заготовок может варьироваться в пределах 10 % от исходного значения.

Определить оптимальное количество токарных, фрезерных и шлифовальных станков. Провести моделирование в течение суток. Выполнить анализ выходной статистики. Задания выполняются согласно индивидуальным вариантам (таблица 1.2).

Таблица 1.2

Вариант	*a*	*b*	*c*	*d*
1	2	3	4	5
1	2±1	7±3	3±1	6±4
2	2±0,5	5±2	3±1	4±2
4	1±0,3	9±1	4±1	7±3
5	2±0,4	10±1	8±2	3±1
6	1,5±0,5	6±1	5±1	3±2
7	3±1	7±3	5±2	6±3
8	3±0,5	11±2	5±1	6±3
9	3±1	12±3	7±1	4±2
10	3±0,5	9±2	3±1	5±2
11	3±1,2	8±3	6±1	7±1
12	3±0,7	7±1	3±1	5±2
13	4±1,5	10±2	8±3	5±3
14	4±1	12±2	5±1	4±1
15	4±0,5	10±3	6±2	8±4

Контрольные вопросы

1. Что такое транзакт?
2. Какие существуют операторы занятия и освобождения одноканальных, многоканальных устройств?
3. Каков принцип работы таймера?
4. Какую информацию содержат операнды операторов storage, queue, depart?
5. На каком методе основана работа пакета GPSS World, в чем его суть?
6. Какие категории объектов присутствуют в пакете GPSS World?
7. Что представляет собой таймер относительного и абсолютного времени?
8. Какие существуют списки в пакете GPSS World, какие из них обязательны?
9. Какие операнды содержит оператор GENERATE?
10. Каково назначение оператора TERMINATE в программе и в блоке таймера?

ЛАБОРАТОРНАЯ РАБОТА №2

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ВЫЧИСЛИТЕЛЬНЫХ ОБЪЕКТОВ

Цель работы – *использование функций и различных законов распределения, моделирование последовательной работы оборудования.*

Теоретические сведения

Генератор случайных величин

Источником случайности в моделях являются генераторы случайных чисел. Их СЧА – *RNn*, где *n* – номер генератора, совпадающий с его начальным числом. Ограничения на количество генераторов нет. Первые семь генераторов (*RN1–RN7*) являются управляемыми. Пользователь может в необходимых случаях установить их начальные значения. В GPSS World количество генераторов случайных величин не ограничено. В них реализуется конгруэнтный мультипликативный метод. Генераторы выдают целочисленные значения, равномерно распределенные в диапазоне 0…999 999. Число, стоящее в начале генерируемой последовательности, равно номеру генератора. Если генератор используется в качестве аргумента функции или при вычислении переменной, генерируемые им значения находятся также в указанном диапазоне.

Законы распределения случайных величин

1) Распределение Вейбулла (Weibula) – WEIBULL (Stream, Locate, Scale, Shape), где Stream – номер генератора случайных чисел; Locate – величина сдвига (константа, добавляемая к значению моделируемой величины); Scale – параметр масштаба функции распределения; Shape – параметр, определяющий форму распределения. Распределение широко используется в теории надежности для описания времени безотказной работы систем на различных этапах их эксплуатации. Так, в период приработки, когда интенсивность отказов систем уменьшается, Shape <1; в период нормальной эксплуатации Shape =1; в период старения, когда интенсивность отказов системы со временем возрастает, Shape >1.

2) Дискретно-равномерное распределение (Discrete Uniform) – DUNIFORM (Stream, Min, Max), где Stream – номер генератора случайных чисел; Min – наименьшее значение из выбранного интервала, Max – наибольшее значение из выбранного интервала.

3) Логистическое распределение (Logistic) – LOGISTIC (Stream, Locate, Scale), где Stream – номер генератора случайных чисел; Locate – величина сдвига (константа, добавляемая к значению моделируемой величины); Scale – параметр масштаба функции распределения, строго положительный.

4) Нормальное распределение (Normal) – NORMAL (Stream, Mean, StdDev), где Stream – номер генератора случайных чисел; Mean – математическое ожидание; StdDev – среднеквадратичное отклонение.

5) Распределение Пуассона (Poisson) – POISSON (Stream, Mean), где Stream – номер генератора случайных чисел; Mean – математическое ожидание.

6) Равномерное распределение (Uniform) – UNIFORM (Stream, Min, Max), где Stream – номер генератора случайных чисел; Min – наименьшее значение из выбранного интервала, Max – наибольшее значение из выбранного интервала.

7) Треугольное распределение (Triangular) – TRIANGULAR (Stream, Min, Max, Mode), где Stream – номер генератора случайных чисел; Min – наименьшее значение модели для генерации (должно быть меньше, чем Max); Max – наибольшее значение; Mode – наиболее частое значение распределения (должно быть больше Min и меньше Max).

8) Экспоненциальное распределение (Exponential) – EXPONENTIAL (Stream, Locate, Scale), где Stream – номер генератора случайных чисел; Locate – величина сдвига (константа, добавляемая к значению моделируемой величины); Scale – параметр формы распределения (математическое ожидание случайных величин при Locate = 0).

Переход транзакта в блок, отличный от последующего. Блок TRANSFER

В GPSS блок TRANSFER (передать) может быть использован в девяти разных режимах. Рассмотрим три основных режима:

1) *Режим безусловной передачи*. Формат блока TRANSFER в режиме безусловной передачи:

TRANSFER ,B

Поля операндов имеют следующий смысл:

- <A> – не используется;
- <B> – позиция блока, в который должен перейти транзакт.

Позиция блока – это номер или метка блока. Так как операнд <A> не используется, то перед операндом <B> должна стоять запятая. В режиме безусловной передачи блок TRANSFER не может отказывать транзакту во входе. Если транзакт входит в блок, то он сразу же пытается войти в блок <B>.

2) *Статистический режим*. В этом режиме осуществляется передача транзакта в один из двух блоков случайным образом. Формат блока TRANSFER в режиме статистической передачи:

TRANSFER <A>,[<B>],<C>

Поля операндов имеют следующий смысл:

- <A> – вероятность передачи транзакта в блок <C>, задаваемая в долях тысячи;
- <B> – позиция блока, в который должен перейти транзакт (с вероятностью (1 – <A>));
- <C> – позиция блока, в который должен перейти транзакт (с вероятностью <A>).

При задании вероятности (операнд <A>) используется не более трех цифр, первым символом записи частоты является «.» (десятичная точка), если используется действительное число, которое должно быть в пределах от 0 до 1,0 (например, 0,235).

Пример:

```
TRANSFER  .333,MET1,MET2
…
MET1  SEIZE       PR1
…
MET2  SEIZE       PR2
```

С вероятностью .333 транзакт переходит в блок с меткой MET2, а с вероятностью .667 – в блок с меткой MET1.

Если с вероятностью .667 транзакт должен перейти к следующему блоку, в блоке TRANSFER можно опустить операнд <B>.

```
TRANSFER  .333,,MET2
MET1  SEIZE       PR1
…
MET2  SEIZE       PR2
```

3) *Режим BOTH*. Если в операнде <A> указано зарезервированное слово BOTH, блок TRANSFER работает в режиме BOTH.

В этом режиме входящий транзакт сначала пытается перейти к блоку, указанному в операнде <B>. Если это сделать не удается, транзакт пытается перейти в блок, указанный в операнде <C>. Если транзакт не сможет перейти ни к тому, ни к другому блоку, то он остается в блоке TRANSFER и при каждом просмотре списка текущих событий будет в том же порядке повторять попытки перехода до тех пор, пока не сможет выйти из блока TRANSFER.

Пример:

```
TRANSFER  BOTH,MET1,MET2
…
MET1  SEIZE       PR1
…
MET2  SEIZE       PR2
```

Транзакт сначала пытается перейти в блок с меткой MET1. Если устройство PR1 занято, транзакт пытается войти в блок с меткой MET2. Если транзакт не может войти и в этот блок (устройство PR2 также занято), он остается в списке текущих событий и повторяет эти попытки при каждом просмотре списка до тех пор, пока не выйдет из блока TRANSFER.

Установка значений параметров транзакта. Блок ASSIGN

При входе транзакта в блок ASSIGN (назначить) значения параметров могут задаваться или изменяться.

Формат блока ASSIGN:

```
ASSIGN      <A>[±],<B>,[<C>]
```

Поля операндов имеют следующий смысл:

– <A> – номер или имя модифицируемого или задаваемого параметра;

– <B> – величина, используемая для модификации (число или СЧА);

– <C> – имя функции.

Блок ASSIGN может быть использован как в режиме замещения значения параметра (начальное значение всех параметров транзактов равно нулю), так и в режиме увеличения и уменьшения. В режиме увеличения предшествующее значение параметра увеличивается на значение, стоящее в операнде <B>. В режиме уменьшения оно уменьшается на величину, стоящую в операнде <B>. Режимы увеличения и уменьшения определяются введением соответственно знаков «плюс» и «минус» перед запятой, которая разделяет операнды <A> и <B>.

При использовании операнда <C> значение операнда <B> умножается на значение функции, указанной в операнде <C>. Параметр, заданный в операнде <A>, изменяется на величину полученного произведения (в режиме увеличения и уменьшения) или приобретает значение результата (в режиме замещения).

Пример:

ASSIGN 3,25

Параметру P3 присваивается значение 25.

ASSIGN P4,FR$BB

Параметру транзакта с номером, записанным в параметре P4, присваивается значение величины загрузки устройства BB (оба операнда заданы косвенным образом).

Блок ASSIGN в режимах накопления и уменьшения:

ASSIGN 4+,Q5

Параметр четыре увеличивается на значение, равное текущей длине очереди пять.

ASSIGN P2-,7

От значения параметра, номер которого задан параметром P2, вычитается семь.

Организация циклов. Блок LOOP

С помощью параметров транзактов в программе можно организовать циклы. Для этого используется блок LOOP. Он управляет количеством повторных прохождений транзактом определенной последовательности блоков модели. Формат блока LOOP:

LOOP <A>,[<B>]

Поля операндов имеют следующий смысл:

– <A> – параметр транзакта, используемый для организации цикла (переменная цикла); он может быть именем, положительным целым числом, СЧА;

– <B> – метка (имя блока) начального блока цикла.

Когда транзакт входит в блок LOOP, параметр, указанный в операнде <A>, уменьшается на единицу, а затем его значение проверяется на равенство нулю. Если значение не равно нулю, то транзакт переходит в блок, указанный в

операнде <B>. Если значение параметра равно нулю, транзакт переходит в следующий блок.

Пример:

```
        ASSIGN      1,3
MET   SEIZE         CHAN
          …
 RELEASE  CHAN
 LOOP       1,MET
```

Цикл организован согласно содержимому первого параметра транзакта. Его начальное значение равно трем. После освобождения устройства проверяется значение первого параметра. Если оно не равно нулю, то транзакт возвращается к блоку, помеченному меткой MET, т. е. занимает устройство с именем CHAN. Всего каждый транзакт будет занимать это устройство три раза.

Блок TEST

Формат блока TEST:

TEST <X> <A>,<B>,[<C>]

Поля операндов имеют следующий смысл:

– <A> – СЧА – левый операнд проверяемого отношения;

– <B> – СЧА – правый операнд проверяемого отношения;

– <C> – имя блока, в который переходит транзакт, если проверяемое отношение имеет значение «ложь».

В дополнительном операторе <X> задается один из следующих операторов отношения (операторы отношения записываются без кавычек):

G (Greater) – больше;

L (Less) – меньше;

E (Equal) – равно;

NE (Not Equal) – не равно;

LE (Less than or Equal) – меньше или равно;

GE (Greater than or Equal) – больше или равно.

Пример:

; режим отказа

TEST LE Ql,Q2

Проверяющий транзакт будет задержан в предыдущем блоке до тех пор, пока длина первой очереди не станет меньше или равна длине второй очереди.

; режим условного перехода

TEST LE Ql,Q2,MET

Проверяющий транзакт перейдет в следующий по порядку блок, если содержимое первой очереди меньше или равно содержимому второй очереди. Если это условие не выполняется, транзакт перейдет в блок с меткой MET.

Оператор EQU

Оператор EQU предназначен для присвоения числовых значений именам, которые используются в модели.

Формат оператора EQU:

<имя> EQU <A>

Поля оператора имеют следующий смысл:

– <имя> – имя переменной, которой присваивается числовое значение;

– <A> – выражение.

Когда интерпретатор обрабатывает оператор EQU, он вычисляет выражение, заданное операндом <A>, после чего создает или переопределяет имя переменной. Имени присваивается результат вычисленного выражения. Полученное значение заменяет ссылки на это имя в операндах или выражениях, применяемых в модели.

Значения имен могут использоваться как внутренние значения переменных пользователя, или они могут определять объекты, такие как метка. Именам, используемым как метки объектов, значения обычно не назначаются. Интерпретатор автоматически назначает индивидуальные значения именам, если они еще не появились в операторе EQU, в выражениях или операндах. Имена могут использоваться для определения объекта в СЧА.

Выражения, содержащиеся в операторе EQU, могут использовать любые из арифметических и логических операторов. Если в выражении применяются параметры, они вычисляются для активного транзакта.

Имена, которым не были явно назначены значения, не могут использоваться в выражении. Необходимо назначить значение для имени прежде, чем будет вычислено выражение. Переменные пользователя могут быть заданы операторами EQU.

Если значение имени определено, то оно сохраняет свое значение на протяжении всего прогона модели.

Переменные FVARIABLE и BVARIABLE используют одну и ту же область имен.

Если необходимо применить числовое имя для объекта, то оно должно быть назначено оператором EQU до определения объекта.

Пример:

```
TZA   EQU  10
GENERATE TZA,FN$EXPON
```

Переменной пользователя TZA назначено значение 10. Далее в программе можно использовать имя этой переменной.

Оператор INITIAL

Формат оператора инициализации ячейки:

INITIAL <A>,<B>

Поля операндов имеют следующий смысл:

- <A> – имя ячейки;
- <B> – начальное значение.

Пример:

```
INITIAL      X$TIMER,100000
```

Блок SAVEVALUE

Значение сохраняемой величины изменяется при входе транзакта в этот блок. Формат блока SAVEVALUE:

```
SAVEVALUE      <A>[±],<B>
```

Поля операндов имеют следующий смысл:

- <A> – имя ячейки;
- <B> – величина, применяемая для модификации.

Блок SAVEVALUE может быть использован как в режиме замещения величины, так и в режиме увеличения или уменьшения. В режиме увеличения предыдущее значение сохраняемой величины увеличивается на значение, заданное операндом <B>, а в режиме уменьшения – уменьшается на это значение. Режимы увеличения и уменьшения определяются введением соответственно знака «плюс» или «минус» перед запятой, разделяющей операнды <A> и <B>.

Пример:

```
SAVEVALUE      5+,100
```

При входе транзакта в блок SAVEVALUE величина ячейки номер пять (X5) увеличивается на 100.

```
SAVEVALUE      PROFIT-,FN$COSTS
```

При входе транзакта в блок SAVEVALUE величина ячейки PROFIT (X$PROFIT) уменьшается на значение функции FN$COSTS.

```
SAVEVALUE      P5,V$ALPHA
```

При входе транзакта в блок SAVEVALUE значение переменной ALPHA записывается в ячейку, номер которой записан в пятом параметре транзакта.

Практические задания

Задание 1. *Использование различных законов распределения.*

Базовые операторы: *exponential, normal, uniform, duniform, triangular, logistic, poisson, weibull, transfer.*

На станции техобслуживания работает *a* бригад по два мастера. Каждые *b* минут приезжает клиент. По виду ремонта обслуживание клиента имеет следующую статистику: 20 % - ремонт системы двигателя, 30 % - ремонт системы подвески, 30 % - ремонт кузова, 20 % - прочий ремонт. Время обслуживания одного клиента согласно виду ремонта составляет *c, d, e, f* часов. Промоделировать работу станции техобслуживания в течение рабочей смены, учитывая данную статистику. Проанализировать выходную статистику, если необходимо, оптимизировать работу станции техобслуживания, неизвестные параметры законов распределения выбрать по своему усмотрению. Задания выполняются согласно индивидуальным вариантам (таблица 2.1).

Таблица 2.1

Вариант	*a*	*b*	*c*	*d*	*e*	*f*
1	2	3	4	5	6	7
1	3	Экспоненциальная величина со средним значением 50	Равномерное распределение в диапазоне 1–2	Распределение Вейбулла (ГСЧ, 0,5; 1; 0,2)	Дискретно-равномерное распределение в диапазоне 0,5–3	Гауссовское распределение с матожиданием 1,8 и СКО 1
2	3	Равномерное распределение в диапазоне 50–70	Экспоненциальная величина со средним значением 2	Нормальное распределение (ГСЧ, 2; 0,5)	Логистическое распределение (ГСЧ, 2,1; 0,2)	Гауссовское распределение с матожиданием 2,1 и СКО 0,5
3	4	Гауссовское распределение с матожиданием 160 и СКО 20	Дискретное равномерное распределение в диапазоне 2–3	Распределение Пуассона со средним значением 1,5	Экспоненциальная величина со средним значением 0,7	Нормальное распределение (ГСЧ, 1; 0,5)
4	4	Распределение Пуассона со средним значением 70	Дискретное равномерное распределение в диапазоне 3–4	Распределение Пуассона со средним значением 1,5	Гауссовское распределение с матожиданием 1,1 и СКО 0,5	Распределение Вейбулла (ГСЧ, 0,5; 1,5; 0,1)
5	2	Дискретное равномерное распределение в диапазоне 30–80	Экспоненциальная величина со средним значением 1	Гауссовское распределение с матожиданием 1 и СКО 0,2	Распределение Вейбулла (ГСЧ, 0,3; 1,2; 0,1)	Треугольное распределение (ГСЧ, 0,2; 1,5; 0,4)
6	2	Экспоненциальная величина со средним значением 80	Гауссовское распределение с матожиданием 2 и СКО 0,5	Логистическое распределение (ГСЧ, 2,1; 0.1)	Треугольное распределение (ГСЧ, 0,2; 1; 0,5)	Распределение Вейбулла (ГСЧ, 0,2; 1,3; 0,1)
7	4	Равномерное распределение в диапазоне 30–50	Экспоненциальная величина со средним значением 1,3	Треугольное распределение (ГСЧ, 0,3; 1; 0,4)	Гауссовское распределение с матожиданием 1,4 и СКО 1	Дискретное равномерное распределение в диапазоне 0,7–1,2
8	3	Гауссовское распределение с матожиданием 40 и СКО 10	Равномерное распределение в диапазоне 0,5–1,5	Распределение Вейбулла (ГСЧ, 0,3; 1,5, 0,1)	Треугольное распределение (ГСЧ, 0,1; 1,5; 0,3)	Логистическое распределение (ГСЧ, 1,1; 0,3)

Продолжение таблицы 2.1

1	2	3	4	5	6	7
9	5	Распределение Пуассона со средним значением 120	Гауссовское распределение с матожиданием 1 и СКО 0,3	Нормальное распределение (ГСЧ, 1,2; 0,2)	Распределение Вейбулла (ГСЧ, 0,2; 1,3, 0,1)	Дискретное равномерное распределение в диапазоне 0,6–1,4
10	4	Дискретное равномерное распределение в диапазоне 120–150	Экспоненциальная величина со средним значением 1,5	Дискретное равномерное распределение в диапазоне 0,5–2,3	Распределение Пуассона со средним значением 1,2	Треугольное распределение (ГСЧ, 0,2; 1,7; 0,4)
11	3	Экспоненциальная величина со средним значением 90	Дискретное равномерное распределение в диапазоне 1–2,5	Логистическое распределение (ГСЧ, 1,7; 0,4)	Нормальное распределение (ГСЧ, 1, 0,3)	Распределение Вейбулла (ГСЧ, 0,3; 1,3, 0,1)
12	2	Равномерное распределение в диапазоне 60–10	Гауссовское распределение с матожиданием 2 и СКО 1	Треугольное распределение (ГСЧ, 0,1; 1,5; 0,3)	Гауссовское распределение с матожиданием 1,5 и СКО 0,1	Дискретное равномерное распределение в диапазоне 1–2
13	5	Гауссовское распределение с матожиданием 200 и СКО 30	Экспоненциальная величина со средним значением 2,5	Распределение Вейбулла (ГСЧ, 0,9; 1; 0,3)	Экспоненциальная величина со средним значением 1,5	Логистическое распределение (ГСЧ, 1,2; 0,1)
14	4	Распределение Пуассона со средним значением 100	Равномерное распределение в диапазоне 1,2–2	Нормальное распределение (ГСЧ, 2; 0,5)	Гауссовское распределение с матожиданием 1 и СКО 0,3	Распределение Вейбулла (ГСЧ, 0,4; 1,2, 0,1)
15	3	Дискретное равномерное распределение в диапазоне 90–130	Гауссовское распределение с матожиданием 1,7 и СКО 1	Распределение Пуассона со средним значением 1,3	Треугольное распределение (ГСЧ, 0,1; 1,6; 0,3)	Распределение Вейбулла (ГСЧ, 0,2; 1; 0,1)

Задание 2. *Организация циклов, применение стандартных числовых атрибутов.*

Базовые операторы: *equ, initial, assign, loop, test.*

На автоматизированную линию по изготовлению печатных плат поступают платы в кассете. Каждая кассета может содержать от одной до трех бракованных плат (с одинаковой вероятностью). Поток кассет – пуассоновский с заданным средним интервалом поступления кассет. На линии задействованы два

робота-манипулятора. Замена бракованных плат включает следующие операции:

– распознавание брака с помощью технического зрения – от *e* до *f* секунд;

– замена брака: время замены одной платы – гауссовская случайная величина со средним значением *a* секунд и стандартным отклонением 1,5 с.

В начале работы имеется *c* запасных плат. Каждые 24 ч этот запас пополняется до *d* шт.

Разработать модель для анализа работы автоматизированной линии изготовления печатных плат в течение 30 сут. Задания выполняются согласно индивидуальным вариантам (таблица 2.2)

Таблица 2.2

Вариант	*a*	*c*	*d*	*e*	*f*
1	2	200	240	1,5	3
2	3	180	220	2	4
3	3	100	150	1	3
4	3	80	160	3	5
5	4	120	140	1,5	4
6	4	260	280	2,2	5
7	5	230	250	1,6	2,5
8	5	170	200	3	4
9	5	300	300	1,1	2,2
10	2	140	170	2,4	3,5
11	3	165	180	0,5	1
12	5	190	200	1,2	1,6
13	5	145	170	0,5	1,4
14	4	90	100	0,6	1,4
15	4	140	160	0,7	0,9

Контрольные вопросы

1. Каковы основные законы распределения, которые можно использовать при моделировании сложных систем?
2. Какой смысл несут генераторы случайных чисел при использовании в законах распределения случайных величин?
3. Как организовать проверку условий в GPSS World?
4. Какие операторы необходимо использовать при организации цикла?
5. Как реализуется режим отказа в блоке test?
6. Как реализовать режим замещения в блоке assign?
7. Какая логическая связь прослеживается между блоками loop и assign?
8. Что задается в дополнительном операторе <X> блока test?
9. В каких режимах может работать блок transfer?
10. Какова суть работы статистического режима блока transfer?

ЛАБОРАТОРНАЯ РАБОТА №3

ИСПОЛЬЗОВАНИЕ СРЕДСТВ РАЦИОНАЛЬНОГО ПОСТРОЕНИЯ МОДЕЛЕЙ

Цель работы – *организация работы модели с взаимосвязанными процессами, а также управления движением транзактов в зависимости от состояния элементов модели.*

Теоретические сведения

Создание копий транзактов. Блок SPLIT

Кроме блока GENERATE для создания транзактов может использоваться блок SPLIT (разделить), который выполняет функцию копирования транзакта, входящего в него. Этот транзакт называется начальным или порождающим. Все копии формируются в момент входа начального транзакта в блок SPLIT. Каждая новая копия становится членом семейства (ансамбля) транзактов, порожденных одним начальным транзактом, который был создан блоком GENERATE.Формат блока SPLIT:

SPLIT <A>[,<B>][,<C>]

Поля операндов имеют следующий смысл:

– <A> – число создаваемых копий транзакта;

– <B> – метка блока, к которому направляются копии;

– <C> – параметр, в котором запоминаются номера копий транзактов.

Операнд <A> может быть положительным целым числом, СЧА. Если вычисленное значение операнда <A> равно нулю, блок SPLIT не выполняет никаких операций. После создания копий начальный транзакт пытается перейти к очередному блоку.

Операнд <B> задает блок, в который переходят копии начального транзакта. Операнд <B> может быть именем (меткой), положительным целым числом, СЧА (в двух последних случаях операнд <B> задает номер блока). Значение операнда <B> вычисляется для каждой копии отдельно.

Операнд <C> задает параметр транзакта, который используется для присвоения копиям последовательных номеров. Операнд <C> может быть именем, положительным целым числом, СЧА.

Транзакты, принадлежащие одному семейству, объединяются интерпретатором в список. По связям внутри семейства транзактов невозможно установить, какой из транзактов является начальным. Если копия транзакта входит в блок SPLIT, то повторная копия становится членом того же семейства, что и первичная копия. Таким образом, каждый транзакт является членом одного и только одного семейства. Семейство может состоять из произвольного числа транзактов. Когда транзакт уничтожается, интерпретатор автоматически ис-

ключает его из членов соответствующего семейства. Таким образом, семейство существует до тех пор, пока из модели не удалится последний из его членов.

В модели одновременно может присутствовать произвольное число семейств, оно все время меняется, поскольку каждый транзакт, генерируемый блоком GENERATE, может создать свое семейство.

Пример:

```
SPLIT  1,MET
ADVANCE 10
…
MET SEIZE  CHAN
…
```

Основной транзакт, порождающий копию, переходит в блок ADVANCE, а транзакт-копия переходит к блоку с меткой MET.

Блок ASSEMBLE

Собирает начальный транзакт и все транзакты-копии из одного семейства, удаляет копии и выдает один начальный транзакт. После сборки из блока ASSEMBLE выходит только один транзакт, который переходит в следующий по номеру блок. Формат блока ASSEMBLE:

ASSEMBLE <A>

Операнд <A> задает счетчик сборки, указывающий сколько членов одного семейства должны быть объединены. Операнд <A> может быть именем, положительным целым числом, СЧА. Первоначальное значение операнда <A> должно быть больше единицы.

Блок GATHER

Накапливает заданное количество транзактов, принадлежащих одному семейству. Он задерживает их до тех пор, пока не соберется необходимое число, указанное операндом <A>. Затем накопленные транзакты одновременно попытаются войти в следующий по номеру блок. Формат блока GATHER:

GATHER <A>

Операнд <A> задает число транзактов, принадлежащих к одному семейству, которое нужно накопить. Операнд <A> может быть именем, положительным целым числом, СЧА.

Блок GATE

Блок GATE управляет потоком транзактов с помощью логических операторов. Блок GATE, как и блок TEST, не изменяет никаких атрибутов транзактов. Он определяет номер следующего блока, к которому должен перейти транзакт из блока GATE. Блок GATE может задержать транзакт на входе, если не задан альтернативный выход. Формат блока GATE:

GATE <X> <A>,[<B>]

Поля операндов имеют следующий смысл:

– <A> – имя или номер объекта, для которого производится проверка; операнд <A> может быть именем, положительным целым числом или СЧА;

– <B> – номер следующего блока для входящего транзакта, если логический оператор имеет значение «ложь». Операнд <B> может быть именем, положительным целым числом или СЧА. Если операнд <B> определен, то он должен содержать номер блока, допустимый для текущей модели.

В дополнительном операторе <X> задается один из следующих логических операторов:

1) Логические операторы, связанные с устройствами:

– NU – устройство j, заданное в операнде <A>, свободно;

– U – устройство j, заданное в операнде <A>, занято (в результате выполнения транзактом блока SEIZE или PREEMPT);

– NI – устройство j, заданное в операнде <A>, не прервано;

– I – устройство j, заданное в операнде <A>, обслуживает прерывания;

– FV – устройство j, заданное в операнде <A>, доступно;

– FNV – устройство j, заданное в операнде <A>, недоступно.

2) Логические операторы, связанные с многоканальными устройствами:

– SE – накопитель j, заданный в операнде <A>, пуст (S[j]=0);

– SNE – накопитель j, заданный в операнде <A>, непуст (S[j]<>0);

– SF – накопитель j, заданный в операнде <A>, заполнен (R[t]=0);

– SNF – накопитель j, заданный в операнде <A>, не заполнен (R[j]<>0);

– SV – накопитель j, заданный в операнде <A>, доступен;

– SNV – накопитель j, заданный в операнде <A>, недоступен.

3) Логические операторы, связанные с транзактами:

M – в блоке j, заданном в операнде <A> блока GATE, находится в состоянии синхронизации транзакт, принадлежащий тому же семейству, что и транзакт, который находится в блоке GATE или пытается войти в этот блок;

NM – в блоке j, заданном в операнде <A> блока GATE, в состоянии синхронизации нет ни одного транзакта, принадлежащего тому же семейству, что и транзакт, который пытается войти в блок GATE.

4) Логические операторы, связанные с логическими ключами:

LS – логический ключ j, заданный в операнде <A>, включен;

LR – логический ключ j, заданный в операнде <A>, выключен.

Блок GATE может работать в режиме отказа и условного перехода.

В режиме условного перехода, если заданный логический оператор имеет значение «истина», транзакт пытается перейти к следующему блоку. Если логический оператор имеет значение «ложь», то транзакт будет пытаться перейти к блоку, номер которого задан в операнде <B> блока GATE. Выбор следующего блока производится один раз в момент вхождения транзакта в блок GATE.

В режиме отказа, если операнд <B> блока GATE пустой (альтернативный выход не задан), транзакты не смогут войти в блок GATE до тех пор, пока указанный в этом блоке логический оператор не будет иметь значение «истина». Интерпретатор не проверяет значение логических операторов, за исключением

операторов M и NM. В режиме условного перехода задержанные транзакты находятся в списках задержки и, таким образом, исключаются из числа транзактов, обрабатываемых интерпретатором, до тех пор, пока соответствующий логический оператор не примет значение «истина».

Пример:

```
QUEUE      CHAN
GATE SNF MEM
DEPART     CHAN
ENTER      MEM
```

В данном примере транзакт помещается в список задержки, если многоканальное устройство MEM заполнено в тот момент, когда транзакт пытается войти в блок GATE. Когда накопитель MEM становится незаполненным, все транзакты выводятся из списка и делают попытку занять накопитель MEM.

Логический переключатель (LOGIC)

Логический переключатель (ключ) используется для моделирования объектов, имеющих всего два положения: «включен» (set или 1) и «выключен» (reset или 0). Блок LOGIC используется для включения, выключения или инвертирования положения ключа. Положение ключа можно проверить любым транзактом в любой части модели. Формат блока LOGIC:

LOGIC <X> <A>

В поле операнда <A> указывается номер или имя ключа.

Когда транзакт входит в блок LOGIC, положение ключа, номер которого задан в операнде <A>, изменяется в зависимости от значения вспомогательного оператора <X> следующим образом:

– S – ключ устанавливается в положение «включен»;

– R – ключ устанавливается в положение «выключен»;

– I – ключ инвертируется, т. е. положение его изменяется на противоположное.

Пример:

LOGIC S KEY

По умолчанию перед началом моделирования ключи находятся в положении «выключен». Установить ключ в состояние «включен» перед началом моделирования можно с помощью инструкции:

INITIAL LS KEY.

Практические задания

Задание 1. *Организация модели взаимосвязанных процессов.*

Базовые операторы: *test, loop, assign, initial.*

На склад прибывают грузовые автомобили с контейнерами (от 4 до 10 шт.). В среднем на склад прибывает *a* автомобилей в час (интервалы между момен-

тами их прибытия – экспоненциальные случайные величины). Одновременно на складе могут разгружаться не более чем три автомобиля. Выгрузка одного контейнера занимает от 4 до 12 мин. Склад вмещает *b* контейнеров. При заполнении склада разгрузка приостанавливается.

Примерно *c* процентов грузов доставляются заказчикам автомобилями, принадлежащими складу. Склад имеет *e* автомобилей. Доставка груза заказчику занимает от 1 до 5 ч. Остальные грузы вывозятся автомобилями заказчиков. Интервал от поступления груза до прибытия за ним автомобилей заказчика составляет от 5 до 20 ч.

Одновременно на складе могут загружаться не более пяти автомобилей. Затраты времени на погрузку примерно такие же, как и на выгрузку.

Разработать имитационную модель для анализа работы склада в течение суток. Определить количество контейнеров, которое проходит через склад. Определить оптимальный объем склада. Определить минимальное и максимальное время доставки груза заказчику с момента прихода машины с грузом на склад своими силами и машинами заказчика. Предложить варианты повышения эффективности работы склада. Задания выполняются согласно индивидуальным вариантам (таблица 3.1).

Таблица 3.1

Вариант	*a*	*b*	*c*	*e*
1	8	200	10	4
2	10	150	15	5
3	12	300	20	3
4	14	100	25	8
5	15	180	30	10
6	7	140	35	12
7	8	220	40	8
8	9	260	45	10
9	10	180	50	8
10	11	270	55	12
11	12	210	60	8
12	13	130	65	9
13	14	280	70	11
14	15	290	30	13
15	16	240	40	7

Задание 2. *Управление движением транзактов в зависимости от состояния элементов модели.*

Базовые операторы: *gate, logic, split, assemble.*

В ремонтную службу предприятия поступают приборы для ремонта согласно закону распределения *b*. Каждый прибор состоит из *a* блоков, требующих ремонта. Блоки, входящие в один прибор, могут ремонтироваться независимо друг от друга на разном оборудовании.

В ремонтной службе имеется три вида оборудования для ремонта. Время ремонта одного блока – экспоненциальная величина со средним значением *c* минут.

После ремонта всех блоков, входящих в прибор, требуется регулировка прибора на специальном стенде, занимающая от *d* до *e* минут.

Приборы поступают в ремонтную службу круглосуточно. Ремонтная служба работает согласно расписанию, т. е. рабочую смену 8 ч. Разработать модель для анализа работы ремонтной службы в течение 30 сут. Задания выполняются согласно индивидуальным вариантам (таблица 3.2).

Таблица 3.2

Вариант	*a*	*b*	*c*	*d*	*e*
1	2	Uniform(8,10)	Exponential(6)	4	10
2	3	Uniform(10,14)	Exponential(8)	5	12
3	3	Uniform(8,12)	Exponential(12)	6	14
4	3	Uniform(10,12)	Exponential(7)	7	8
5	4	Uniform(6,10)	Exponential(10)	8	9
6	4	Uniform(6,8)	Exponential(11)	10	11
7	5	Uniform(9,13)	Exponential(5)	9	13
8	5	Uniform(9,11)	Exponential(9)	8	12
9	5	Uniform(7,10)	Exponential(6)	7	14
10	2	Uniform(7,11)	Exponential(10)	6	15
11	3	Uniform(12,14)	Exponential(8)	11	20
12	5	Uniform(12,16)	Exponential(7)	12	16
13	5	Uniform(10,16)	Exponential(9)	5	14
14	4	Uniform(14,16)	Exponential(12)	6	10
15	4	Uniform(8,11)	Exponential(5)	7	9

Контрольные вопросы

1. В чем состоит назначение оператора test при реализации модели с взаимосвязанными процессами?
2. Для чего используется команда initial? Каков ее формат объявления?
3. В каких состояниях может находиться логический переключатель?
4. В чем заключается суть использования логического ключа?
5. Какой оператор используется для проверки состояния переключателя?
6. В чем заключается отличие оператора test от оператора gate?
7. Для чего служит оператор split?
8. Чем отличаются операторы assemble и gather?
9. Какие значения может принимать внутренний операнд <X> в блоке GATE?
10. Какие значения может принимать внутренний операнд <X> в блоке LOGIC?

ЛАБОРАТОРНАЯ РАБОТА №4

ОРГАНИЗАЦИЯ СИНХРОННОЙ РАБОТЫ ПОДРАЗДЕЛЕНИЙ

Цель работы – *организация синхронной работы подразделений, применение табличных величин для сбора статистики и ввода исходных данных.*

Теоретические сведения

Блок ADOPT

Изменяет семейство активного транзакта. Синтаксис:

ADOPT <A>

<A> – число либо СЧА.

Особенности выполнения в том, что блок всегда принимает транзакты.

Значение операнда <A> округляется. Если результат меньше или равен нулю, то происходит останов по ошибке, в противном случае значение результата становится новым номером семейства активного транзакта.

Пример:

ADOPT 150

В результате выполнения блока активный транзакт становится членом семейства с именем 150.

Блок синхронизации MATCH

Блок MATCH синхронизирует движение транзактов с другим блоком MATCH. Формат блока MATCH:

MATCH <A>

Операнд <A> указывает имя сопряженного блока. Сопряженным блоком является также блок MATCH.

Пример:

В локальной сети рабочая станция опрашивается каждые 30 мс. Если на рабочей станции есть сообщение для передачи, то оно занимает канал.

```
MET1  MATCH    MET2
      SEIZE    CHAN
…
MET2  MATCH    MET1
      ADVANCE 30
```

При входе транзакта-сообщения в блок MATCH с меткой MET1он будет ждать (в списке синхронизации) момента, когда другой опрашиваемый транзакт, принадлежащий тому же семейству, не войдет в сопряженный блок MATCH с меткой MET2. Только после этого сообщение займет канал CHAN, а опрашиваемый транзакт перейдет в блок ADVANCE.

Формирование GPSS-таблиц

Блок GPSS-таблиц служит для сбора статистической информации о любом параметре модели, накопления выборочных значений случайных величин, статистической обработки этих выборок и построения гистограммы распределения вероятностей значений указанного параметра. Графическим аналогом GPSS-таблицы является гистограмма выборочных значений случайной величины, которую можно просмотреть в окне таблицы. Прежде чем использовать таблицу, ее нужно определить, а затем указать, для какого параметра модели следует собрать статистику.

Оператор TABLE

Таблица определяется оператором TABLE. Формат оператора TABLE:

<имя> TABLE <A>,<B>,<C>,<D>

Поля операндов имеют следующий смысл:

- <A> – СЧА параметра, для которого выполняется сбор статистики,
- <B> – верхняя граница самого левого интервала таблицы;
- <C> – ширина интервалов таблицы, за исключением самого левого и самого правого;
- <D> – общее число интервалов таблицы, включая самый левый и самый правый интервалы.

Пример:

MET1 TABLE M1, 10, 100, 8

Задана таблица с именем MET1 для табулирования времени пребывания в модели активного транзакта M1 при верхней границе первого значения интервала, равного 10, ширине интервалов в 100 единиц и общем числе интервалов таблицы, равном 8.

Блок QTABLE

Для сбора статистических данных об очередях используется оператор QTABLE. Его формат совпадает с форматом оператора TABLE, за исключением того, что операнд <A> задает имя очереди.

Блок TABULATE

Механизм сбора статистики в таблицу включается, когда транзакт входит в блок TABULATE (табулировать), у которого в поле операнда <A> указано имя или номер таблицы, описанной оператором TABLE.

Формат блока TABULATE:

TABULATE <A>

Одну и ту же таблицу можно использовать в нескольких блоках модели оператора TABULATE.

Пример:

Пример использования таблицы для сбора статистической информации о значениях, которые принимает в модели величина, хранящаяся в первом параметре транзактов.

TAB TABLE P1,4,3,5
…
TABULATE TAB

Атрибут M1

При каждом входе транзакта в модель интерпретатор фиксирует для него текущее значение времени, которое называется *отметкой времени*. Она может быть интерпретирована как время «рождения» транзакта или время входа транзакта в модель. В явном виде отметка времени недоступна. Однако существует СЧА, связанный со значением времени входа транзакта в модель. Его имя M1, а значение определяется следующим образом: M1 = текущее значение таймера абсолютного времени – значение времени входа транзакта в модель. Значение M1 для каждого транзакта изменяется в процессе моделирования. Сразу после входа транзакта в модель M1=0, через 10 единиц модельного времени M1=10 и т. д.

Стандартный числовой атрибут M1 измеряет время, которое прошло с момента входа транзакта в модель. Однако очень часто требуется знать время, затраченное на перемещение транзакта между двумя произвольными точками модели (т. е. транзитное время). Для этого используется блок MARK. При входе транзакта в блок MARK значение таймера абсолютного времени записывается в качестве одного из его параметров. Такую запись называют отметкой транзакта.

Блок MARK

Формат блока MARK:

MARK <A>

Поля операндов имеют следующий смысл:

– <A> – номер параметра, в который записывается значение абсолютного времени (целое число или СЧА).

Пусть необходимо определить интервал времени, на протяжении которого транзакт проходит от точки T1 к точке T2. Для этого нужно выполнить два действия:

1) в точку T1 поместить блок MARK j, где j – номер параметра, в который записывается значение абсолютного времени в момент записи;

2) в точке T2 обратиться к СЧА с именем MPj, где j – номер параметра, в котором сделана отметка времени транзакта; СЧА MPj будет иметь следующее значение: MPj = текущее значение таймера абсолютного времени – значение j-го параметра.

Практические задания

Задание 1. *Разработка имитационной программы для анализа работы участка технологического процесса производства.*

Базовые операторы: *split, assemble, gather.*

На участке цеха по выпуску напитков выполняются следующие операции: заполнение бутылок напитком и закупоривание, наклейка этикеток, установка бутылок в ящики.

Пустые бутылки по одной поступают в цех в среднем через каждые *a* секунд (экспоненциальная случайная величина). По мере поступления бутылки устанавливаются в поддон, вмещающий 25 шт. Поддон с бутылками поступает к машине, выполняющей заполнение и закупоривание. Эти операции выполняются для всех бутылок в поддоне одновременно и занимают *b* секунд на поддон (обе операции вместе). На закупоренные и заклеенные бутылки наклеиваются этикетки; эта операция занимает *c* секунд на бутылку (включая извлечение ее из поддона, наклеивание этикетки и установку обратно в поддон). По окончании всей обработки бутылки из поддона перегружаются в ящики, вмещающие по 6 шт.

Всего на участке используется *d* поддонов. Перемещение поддона от места подачи пустых бутылок к машине для заполнения и закупоривания, от нее – к месту наклейки этикеток, и оттуда – к месту перегрузки бутылок в ящики занимает *e* секунд; возвращение пустого поддона к месту подачи пустых бутылок занимает 20 с.

Разработать имитационную программу для анализа процесса работы участка в течение недели (5 д. по 3 смены). Предложить возможные методы повышения выпуска продукции при минимальных изменениях технологического процесса производства. Задания выполняются согласно индивидуальным вариантам (таблица 4.1).

Таблица 4.1

Вариант	*a*	*b*	*c*	*d*	*e*
1	2	21±2	Uniform(8,10)	4	10
2	3	22±1	Uniform(10,14)	5	12
3	3	24±5	Uniform(8,12)	6	14
4	3	29±6	Uniform(10,12)	7	8
5	4	30±2	Uniform(6,10)	8	9
6	4	35±4	Uniform(6,8)	10	11
7	5	38±1	Uniform(9,13)	9	13
8	5	39±5	Uniform(9,11)	8	12
9	5	36±6	Uniform(7,10)	7	14
10	2	25±2	Uniform(7,11)	6	15
11	3	34±3	Uniform(12,14)	11	20
12	5	38±4	Uniform(12,16)	12	16
13	5	40±1	Uniform(10,16)	5	14
14	4	27±3	Uniform(14,16)	6	10
15	4	29±5	Uniform(8,11)	7	9

Задание 2. *Разработка имитационной программы для процесса работы мастерской.*

Базовые операторы: *split, assemble, gather*.

Мастерская по наладке устройств получает задания каждые *a* минут. Устройство состоит из трех частей. Бригада ремонтников (3 чел.) после получения устройства в течение (5 ± 2) мин разбирает его на части, и каждый из рабочих занимается своей частью в течение *b* минут. Затем части отправляются на тестирование, а бригада берется за новое устройство, но только после отправки всех трех частей. Тестирование занимает по *c* минут на часть, после чего устройство собирают в течение (6 ± 3) мин и отправляют на склад.

Разработать имитационную программу для анализа процесса работы мастерской в течение дня (две смены). Предложить способы повышения эффективности работы ремонтников. Задания выполняются согласно индивидуальным вариантам (таблица 4.2).

Таблица 4.2

Вариант	*a*	*b*	*c*
1	29±6	15±2, 12±5, 17±2	6±2, 5±3, 2±1
2	25±2	10±3, 14±1, 12±5	3±1, 2±2, 2±1
3	34±3	15±5, 18±3, 13±3	10±2, 5±3, 2±1
4	38±4	15±3, 10±5, 17±2	12±2, 9±3, 10±4
5	22±1	12±4, 8±6, 9±3	6±4, 4±3, 5±3
6	24±5	10±3, 10±1, 10±3	5±1, 7±1, 9±1
7	30±2	13±2, 12±5, 12±2	10±2, 5±3, 2±1
8	35±4	20±1, 14±2, 15±4	12±2, 9±3, 10±4
9	36±6	12±3, 16±2, 14±2	6±2, 5±3, 2±1
10	38±1	15±2, 12±5, 17±2	6±4, 4±3, 5±3
11	39±5	13±2, 12±5, 12±2	5±1, 7±1, 9±1
12	21±2	10±3, 14±1, 12±5	6±2, 5±3, 2±1
13	40±1	15±5, 18±3, 13±3	12±2, 9±3, 10±4
14	27±3	15±3, 10±5, 17±2	10±2, 5±3, 2±1
15	29±5	15±2, 12±5, 17±2	12±2, 9±3, 10±4

Задание 3. *Синхронизация работы, формирование таблиц.*

Базовые операторы: *match, adopt, table, tabulate*.

В цех поступают заготовки двух типов. Заготовки первого типа поступают через *a* минут и обрабатываются на станке в течение *b* минут. Брак на операции составляет *c* процентов. Заготовки второго типа поступают на другой станок с интервалом *d* минут, обрабатываются *e* минут. Брак составляет *f* процентов. После этого обе детали попадают одновременно на третий станок, где собираются в одну деталь в течение *g* минут.

Определить минимальное и максимальное время сборки одной детали – от поступления заготовки до собранной детали, определить количество брака на каждом станке. Занести время обработки деталей в таблицу с интервалом 1 мин. Задания выполняются согласно индивидуальным вариантам (таблица 4.3).

Таблица 4.3

Вариант	*a*	*b*	*c*	*d*	*e*	*f*	*g*
1	6–8	Normal(6,1)	2	Exponential(10)	Normal(8,1)	1	10–12
2	7–9	Normal(6,1)	2	Normal(8,1)	Exponential(7)	2	7–12
3	5–6	Normal(5,1)	3	Exponential(6)	Poisson(6)	3	8–10
4	5–7	Normal(6,1)	3	Poisson(7)	Normal(6,1)	4	8–12
5	6–10	Normal(8,1)	4	Exponential(7)	Exponential(7)	5	8–10
6	7–10	Exponential(7)	4	Normal(8,1)	Poisson(7)	1	10–12
7	8–10	Exponential(7)	5	Normal(10,1)	Normal(8,1)	2	8–12
8	8–9	Exponential(7)	5	Poisson(10)	Exponential(8)	3	10–14
9	8–11	Exponential(7)	1	Poisson(7)	Poisson(7)	4	8–12
10	10–12	Exponential(7)	1	Poisson(10)	Normal(8,1)	5	11–12
11	4–6	Poisson(5)	1	Normal(5,1)	Exponential(4)	5	6–8
12	4–8	Poisson(5)	2	Exponential(5)	Poisson(5)	4	7–8
13	5–9	Poisson(5)	3	Poisson(8)	Normal(8,1)	3	7–9
14	6–9	Poisson(7)	4	Normal(8,1)	Exponential(7)	2	7–9
15	6–8	Poisson(7)	5	Exponential(10)	Poisson(9)	1	8–9

Контрольные вопросы

1. Что такое транзакт?
2. В чем заключается назначение оператора split?
3. В чем заключается назначение оператора gather?
4. Для чего служит оператор match?
5. Как синхронизировать работу двух видов оборудования?
6. Когда применяется оператор adopt?
7. Как организовать вывод необходимой статистики в таблицу?
8. Что записывается в операнд <A> блока TABLE?
9. Как определить время пребывания активного транзакта в модели?
10. Для чего используется блок MARK?

ЛАБОРАТОРНАЯ РАБОТА №5

ОБРАБОТКА ВНЕШТАТНЫХ СИТУАЦИЙ ПРИ ИМИТАЦИОННОМ МОДЕЛИРОВАНИИ

Цель работы – *организация прерываний программы, работы одноканальных и многоканальных устройств, использование приоритетов.*

Теоретические сведения

Организация обслуживания с прерыванием. Блоки PREEMPT и RETURN

Ситуацию обслуживания транзакта в устройстве с прерываниями можно смоделировать, считая, что отказ оборудования представляет собой транзакт, приоритет которого выше, чем у транзакта, обрабатываемого устройством. В этом случае более приоритетный транзакт должен прервать обслуживание менее приоритетного транзакта, т. е. выгрузить его из устройства. Дословный перевод с английского языка слова preempt – выгрузить, но с точки зрения работы одноканальной СМО принято использовать термин «захватить» устройство. Для организации обслуживания в устройстве с прерываниями используют пару блоков PREEMPT (захватить) – RETURN (вернуть).

Формат блока:

PREEMPT <A>,<B>,<C>,<D>,<E>

Поля операндов имеют следующий смысл:

– <A> – номер или имя устройства;
– <B> – возможность захвата по приоритету;
– <C> – имя блока, в который переходит прерванный транзакт;
– <D> – номер параметра у прерванного транзакта;
– <E> – возможность снятия транзакта с обслуживания.

Операнд <A> определяет номер или имя устройства, на котором генерируется прерывание.

Операнд <B> задает приоритетный режим (если <B>=PR) или режим прерывания (если этот операнд опущен). При работе в приоритетном режиме транзакт, уже занимающий устройство или генерирующий на нем прерывание, может быть прерван только транзактом, приоритет которого выше приоритета данного транзакта. Прерванные транзакты претендуют на дополнительное использование устройства, когда прервавший их транзакт войдет в соответствующий блок RETURN. Прерванные транзакты помещаются в список задержки в порядке приоритета.

Операнд <C> задает номер или имя блока, в который в этот же момент времени должен попытаться войти прерванный транзакт. Прерванный транзакт теряет управление устройством, но претендует на право его использования, если только не задан аргумент операнда <E>. В приоритетном режиме работы

желательно задавать операнд <C>, если прерывающий транзакт имеет более высокий приоритет, чем прерываемый.

Операнд <D> задает номер параметра, связанного с прерванным транзактом. Если прерываемый транзакт в момент прерывания направляется в список будущих событий, то остаток времени записывается в заданный параметр. Если такой параметр не существует, то он создается. В приоритетном режиме работы операнд <D> задают только в том случае, если прерывающий транзакт имеет более высокий приоритет, чем прерываемый.

Операнд <E> задает либо не задает режим удаления (RE), в котором прерванный транзакт более не претендует на использование устройства и пытается войти в блок, заданный операндом <C> (если в операнде <E> стоит RE, то должен быть определен и операнд <C>). В приоритетном режиме работы режим RE используется только в том случае, если приоритет прерывающего транзакта больше приоритета прерываемого. При использовании RE прерванный транзакт не должен входить в блоки RELEASE или RETURN, связанные с устройством, в котором обслуживался прерванный транзакт. Если режим RE не задан (операнд <E> опущен), то прерванный транзакт по возвращении в список текущих событий будет вновь пытаться занять устройство.

Блок RETURN

Предназначен для освобождения ранее захваченного устройства.

Формат блока:

RETURN <A>

В операнде <A> задается номер устройства, с которого снимается прерывание. Прерывание может быть снято в блоке RETURN только тем транзактом, которым оно было сгенерировано.

Пример:

В качестве примера использования блоков PREEMPT и RETURN смоделируем обработку отказа оборудования.

```
; первый процесс – нормальная работа оборудования
GENERATE …
…
SEIZE CHAN
…
RELEASE    CHAN
TERMINATE
; второй процесс – поток отказов оборудования
GENERATE ….,1
PREEMPT    CHAN,PR,ERR,5,RE
ADVANCE   …
RETURN     CHAN
TERMINATE
; обработка события отказа оборудования
ERR    …
```

Во втором процессе блок GENERATE в определенный момент времени генерирует транзакт, имитирующий отказ оборудования, приоритет которого равен единице. Он выше, чем приоритет транзакта, моделирующего нормальную работу оборудования. Этот факт отмечается наличием значения PR в операнде <B> транзакта, моделирующего отказ оборудования. Значение RE в операнде <E> означает, что прерванный транзакт снимается с устройства CHAN и переходит на метку ERR, по которой находится группа операторов, моделирующих обработку отказа оборудования. Прерванный транзакт более не претендует на занятие устройства CHAN после того, как прерывающий транзакт освободит его.

Блок SUNAVAIL

Формат блока:

SUNAVAIL <A>

Операнд <A> – название МКУ (накопитель).

Переводит накопитель в состояние недоступности, при котором транзакты не могут войти в накопитель. Уменьшение содержимого накопителя в этот период может происходить путем прохождения транзактами блока LEAVE. Номер или диапазон номеров накопителей, переводимых в состояние недоступности, записывается в поле A.

Блок SAVAIL

Формат блока:

SAVAIL <A>

Операнд <A> – название МКУ.

Переводит заданный накопитель из состояния недоступности в состояние доступности. Если данный накопитель уже доступен, то блок SAVAIL никаких действий не выполняет.

Пример:

```
SUNAVAIL PRESS
TEST    NE   PHI,O,NEXT
SAVAIL       PRESS
```

(т. е. недоступные накопители становятся доступными при появлении транзакта с ненулевым параметром первого типа «полуслово»).

Блок FUNAVAIL

Блоком FUNAVAIL (символом F обозначает одноканальное устройство (ОКУ), UNAVAIL – недоступный) моделируется недоступность ОКУ. При использовании этого блока статистика ОКУ не искажается.

Формат блока:

FUNAVAIL A, B, C, D, E, F, G, H

Блок делает недоступным ОКУ с именем или номером, указываемым операндом <A>.

Все транзакты, обрабатываемые блоком FUNAVAIL, разделяются на три класса, которые и определяют назначение операндов:

– транзакт, занимающий ОКУ (по SEIZE или PREEMPT) в момент перевода его в недоступное состояние (операнды <B>, <C>, <D>);

– ранее прерванные транзакты, находящиеся в списке прерываний (операнды <E>, <F>);

– транзакты, находящиеся в списке отложенных прерываний и в списке задержки ОКУ (операнды <G>, <H>).

Операндом <B> задаются режимы обработки транзакта, занимающего ОКУ в момент перевода его в недоступное состояние, а именно:

– CO – режим продолжения: продолжить обработку занимающего ОКУ транзакта во время недоступности;

– RE – режим удаления: удалить и направить занимающий ОКУ транзакт к блоку, метка которого должна быть указана операндом <C>;

– по умолчанию – прервать обработку и поместить в список прерываний ОКУ, после восстановления доступности этот транзакт может занять ОКУ и «дообслужиться».

Операнд <C> – метка блока, в который будет направлен в режиме удаления транзакт, занимавший ОКУ в момент перевода его в недоступное состояние.

Операнд <D> – номер или имя параметра транзакта, занимавшего ОКУ в момент перевода его в недоступное состояние; если он будет удален (режим RE), т. е. исключен из списка будущих событий, в этот параметр будет записано время, оставшееся удаленному транзакту до конца обслуживания.

Операндом <E> задаются режимы обработки транзактов, находящихся к моменту перевода ОКУ в недоступное состояние в списке прерываний, т. е. тех транзактов, обслуживание которых на данном ОКУ было ранее прервано, а именно:

– CO – режим продолжения: продолжить работу ОКУ во время недоступности – обслуживать транзакты из списка прерываний;

– RE – режим удаления: удалить и направить транзакты из списка прерываний к новому блоку, метка которого должна быть указана операндом <F>;

– по умолчанию – оставить ранее прерванные транзакты в списке прерываний ОКУ и запретить им занимать его во время недоступности.

Операнд <F> указывает метку блока, к которому будут направлены транзакты из списка прерываний ОКУ, вследствие чего они не могут находиться в СБС, поэтому для них нет возможности занесения в их параметры времени, оставшегося до конца обслуживания.

Операнд <F> может использоваться и тогда, когда отсутствует операнд <E> (по умолчанию). В этом случае для перемещенных к новому блоку транзактов прерывание обслуживания сохраняется.

Операндом <G> задаются режимы обработки транзактов, находящихся к моменту перевода ОКУ в недоступное состояние в списке отложенных преры-

ваний, т. е. ожидающих выполнения с прерыванием, и в списке задержки, а именно:

– CO – режим продолжения: продолжить работу ОКУ во время недоступности – обслуживать транзакты из списка отложенных прерываний и списка задержки;

– RE – режим удаления: удалить и направить транзакты из списка отложенных прерываний и списка задержки к новому блоку, метка которого должна быть указана операндом <H>;

– по умолчанию – оставить транзакты в списке отложенных прерываний и списке задержки ОКУ и запретить им занимать его во время недоступности.

Операндом <H> указывается метка нового блока, к которому в режиме удаления (RE) направляются транзакты из списка отложенных прерываний и списка задержки. Когда операнд <G> не используется, нельзя использовать и операнд <H>.

Блок FAVAIL

Изменяет состояние ОКУ на доступное, т. е. восстанавливает обычный режим вхождения транзактов в ОКУ.

Блок имеет следующий формат:

FAVAIL A

Операнд <A> – имя или номер ОКУ.

Все транзакты, ожидающие доступного состояния ОКУ, указанного операндом <A>, активизируются и могут попытаться занять его.

Установка приоритета транзакта. Блок PRIORITY

Блок PRIORITY (назначить приоритет) присваивает или изменяет приоритет транзакта, если он был задан блоком GENERATE (по умолчанию приоритет транзакта равен нулю).

Формат блока PRIORITY:

PRIORITY <A>

Поле операндов <A> задает новое значение приоритета (число или СЧА).

Новое значение приоритета может быть меньше, больше или равно текущему значению приоритета транзакта. Приоритет влияет на порядок выбора транзакта для обслуживания устройствами и на порядок просмотра транзактов в списке текущих событий.

Стандартный числовой атрибут этого блока – PR. Поскольку уровень приоритета транзакта может изменяться от 0 до 127, то PR будет выдавать значение в диапазоне 0–127.

Пример:

PRIORITY 100

Вошедшему в этот блок транзакту присваивается приоритет 100.

DELAY FUNCTION PR,D3

1,4/2,7/3,10

…

ADVANCE FN$DELAY

…

Задержка в блоке ADVANCE зависит от приоритета транзакта. Транзакт с наиболее низким приоритетом (1) задерживается на четыре единицы модельного времени, транзакт с наиболее высоким приоритетом (3) задерживается на 10 единиц модельного времени.

Практические задания

Задание 1. *Моделирование профилактических работ на производственном участке.*

Базовые операторы: *savail, sunavail.*

В цехе установлено устройство обработки деталей, которое может обрабатывать по *a* деталей одновременно за *e* минут. Детали на обработку поступают каждые *b* минут. Однако устройство необходимо останавливать для профилактического обслуживания каждые *c* минут, перерыв в работе длится *d* минут.

Необходимо скорректировать интервал поступления деталей так, чтобы детали, накапливающиеся за время перерыва, успевали обработаться до следующего перерыва. Построить график изменения количества деталей в очереди на обработку. Задания выполняются согласно индивидуальным вариантам (таблица 5.1).

Таблица 5.1

Вариант	*a*	*b*	*c*	*d*	*e*
1	2	5±2	6	40	7
2	2	3±1	2	50	8
3	1	5±2	3	40	6
4	3	7±1	2	20	8
5	3	5±1	1	10	5
6	3	9±2	5	10	5
7	1	5±2	2	40	9
8	4	6±2	4	30	7
9	3	5±1	3	20	5
10	4	8±2	6	50	10
11	2	7±2	5	30	8
12	1	5±2	3	20	4
13	2	6±1	4	40	12
14	4	5±2	3	20	7
15	3	4±1	2	30	14

Задание 2. *Моделирование прерываний работы устройств.*

Базовые операторы: *preempt, return, priority.*

В цехе установлен станок по обработке деталей. Обработка длится *a* минут. Заказы на детали бывают трех видов: обычные, срочные и сверхсрочные, они приходят каждые *b, c, d* минут соответственно. Более срочный заказ прерывает выполнение менее срочного.

Промоделировать работу станка в течение недели (5 д., 2 смены). Показать графически (plot) как происходят прерывания выполнения заказов соответственно приоритетам. Задание выполняется согласно индивидуальному варианту (таблица 5.2).

Таблица 5.2

Вариант	*a*	*b*	*c*	*d*
1	30	60	90	120
2	20	50	70	80
3	10	40	45	50
4	5	10	15	20
5	40	55	65	70
6	30	35	45	55
7	5	7	10	12
8	50	70	90	100
9	35	40	50	55
10	17	20	22	30
11	20	35	60	70
12	18	27	35	40
13	25	38	45	60
14	45	55	75	85
15	5	15	20	30

Задание 3. *Обработка внештатных ситуаций и их профилактика.*

Базовые операторы: *preempt, return, favail, funavail.*

В цехе установлен станок для обработки деталей. Детали поступают каждые *a* минут, обработка длится *b* минут. Каждые *c* минут станок останавливают и в течение *d* минут осматривают на наличие неисправностей. Кроме того, станок выходит из строя (обнаруживается поломка при осмотре) каждые *e* минут. На ремонт тратится *f* минут. После ремонта обрабатывавшуюся в момент поломки деталь необходимо подвергнуть действию *g* (забраковать, обработать заново, продолжить обработку), а осмотр станка – действию *h* (не прекращается во время ремонта, продолжается после, проходит заново).

Промоделировать работу станка в течение недели (5 д., 2 смены). Задания выполняются согласно индивидуальным вариантам (таблица 5.3).

Таблица 5.3

Вариант	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*
1	2	1	10	2	52	3	Забраковать	Не прекращать
2	3	1	25	2	45	3	Обработать заново	Не прекращать
3	4	2	20	1	40	3	Продолжить	Не прекращать
4	5	4	40	5	50	2	Забраковать	Продолжить после
6	3	2	32	3	52	3	Продолжить	Продолжить после
7	2	1	15	1	28	3	Забраковать	Заново
8	4	3	10	5	32	5	Обработать заново	Заново
9	3	1	14	3	44	1	Продолжить	Заново
10	5	1	32	5	58	3	Забраковать	Не прекращать
11	5	3	28	2	56	2	Обработать заново	Не прекращать
12	4	1	35	1	64	3	Продолжить	Не прекращать
13	3	2	18	2	36	1	Забраковать	Продолжить после
14	4	2	33	4	62	3	Обработать заново	Продолжить после
15	6	3	35	1	48	2	Продолжить	Продолжить после

Контрольные вопросы

1. Какой смысл имеют операнды в блоке PREEMPT?
2. Каково назначение блока RETURN?
3. В чем состоит отличие блоков SANAVAIL и FANAVAIL?
4. Для чего служат блоки FAVAIL, SAVAIL?
5. Для чего нужны списки задержанных и прерванных транзактов?
6. На какакие классы делятся транзакты, обрабатываемые блоком FANAVAIL?
7. Какие режимы можно задать в операнде <B> блока FANAVAIL?
8. Какую информацию содержат операнды <G> и <H> блока FANAVAIL?
9. Как установить уровень приоритета транзакта?
10. Как располагаются транзакты в списке текущих событий?

ЛАБОРАТОРНАЯ РАБОТА №6

МОДЕЛИРОВАНИЕ ВЫБОРА УСТРОЙСТВ ПО ОПРЕДЕЛЕННОМУ КРИТЕРИЮ

Цель работы – *организация выбора устройств по заданному критерию, подсчет подходящих устройств.*

Теоретические сведения

Функции

В GPSS рассматриваются пять типов функций:

1) дискретная числовая (D);
2) непрерывная числовая (C);
3) табличная числовая (L);
4) дискретная атрибутивная (E);
5) табличная атрибутивная (M).

Дискретная функция представляет собой кусочно-постоянную функцию, которая состоит из горизонтальных ступеней (рисунок 6.1). *Непрерывная функция* представляет собой кусочно-непрерывную функцию. Непрерывная функция в GPSS состоит из соединенных между собой прямых отрезков и представляет ломаную линию (рисунок 6.2). Чтобы задать дискретную функцию, необходимо задать координаты крайних правых точек горизонтальных отрезков. Для непрерывной функции необходимо задать координаты всех точек, которые являются концами отрезков.

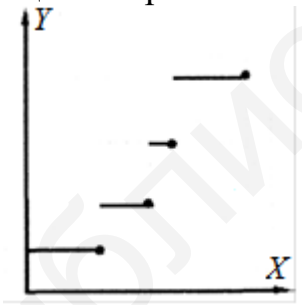

Рисунок 6.1 – Дискретная функция

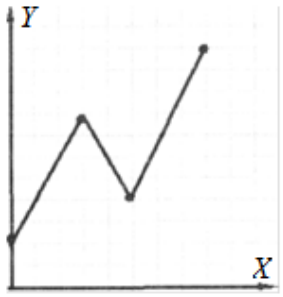

Рисунок 6.2 – Непрерывная функция

Действия, необходимые для определения GPSS-функции:

1) Присвоить функции имя. Имя может быть числовым либо символьным.
2) Задать аргумент функции. Аргументом могут быть:

- ссылка на генератор случайных чисел, используемый для розыгрыша в соответствии с распределением, заданным функцией;
- стандартный числовой атрибут;
- ссылка на любую другую функцию.

3) Задать тип функции и число пар аргумент/значение функции.
4) Задать значения пар аргумент/значение функции.

Оператор FUNCTION

Для задания GPSS-функции используются два оператора:

- определения функции;
- описания координат функции.

Формат оператора определения функции:

<имя> FUNCTION <A>,<B>

Поля операндов имеют следующий смысл:

- <A> – датчик случайных чисел или СЧА;
- <B> – Dn либо Cn, где D определяет дискретную функцию, C – непрерывную функцию; n – для дискретной функции – число различных значений, получаемых функцией (количество горизонтальных отрезков), для непрерывной функции – число, на единицу большее числа отрезков, составляющих функцию (количество точек). Формат оператора описания координат функции: X1, Y1/X2, Y2/… Xn, Yn, где Xi и Yi – координаты i-й точки функции (в случае моделирования случайной величины Xi является i-й кумулятивной частотой, Yi – соответствующим значением случайной величины). Особенности оператора описания координат функции:
- основной единицей информации оператора описания координат функции является пара значений Xi, Yi (координаты точки i);
- значения координат Xi и Yi одной точки функции разделяются запятой;
- последовательные наборы координат разделяются знаком «/»;
- координаты Xi и Yi, относящиеся к одной точке, задаются одним оператором, т. е. пара координат одной точки не должна разрываться;
- все строки описания координат функции должны начинаться с первой позиции;
- во всех случаях значения аргумента должны удовлетворять следующим неравенствам: X1<X2<…<Xi<…<Xn.

Значение функции является ее стандартным числовым атрибутом. Способ ссылки на этот атрибут зависит от того, как задано имя функции: в символьном или числовом виде. Если имя числовое, то к значению функции можно обратиться через FNj (где j – номер функции), если имя символьное – через FN$<имя функции>.

Пример:

Необходимо смоделировать дискретную случайную величину, заданную параметрами, приведенными в таблице в таблице 6.1.

Таблица 6.1

Интервал	Относительная частота	Кумулятивная частота	Диапазон	Значение случайной переменной
1	0,20	0,20	[0,0–0,20]	2
2	0,20	0,40	[0,20–0,40]	5
3	0,20	0,60	[0,40–0,60]	8
4	0,22	0,82	[0,60–0,82]	9
5	0,18	1,00	[0,82–1,00]	12

GPSS-функцию можно определить таким образом:

```
SERV        FUNCTION        RN4,D5
0.20,2/0.40,5/0.60,8/0.82,9/1,12
```

Графическая интерпретация функции показана на рисунке 6.3.

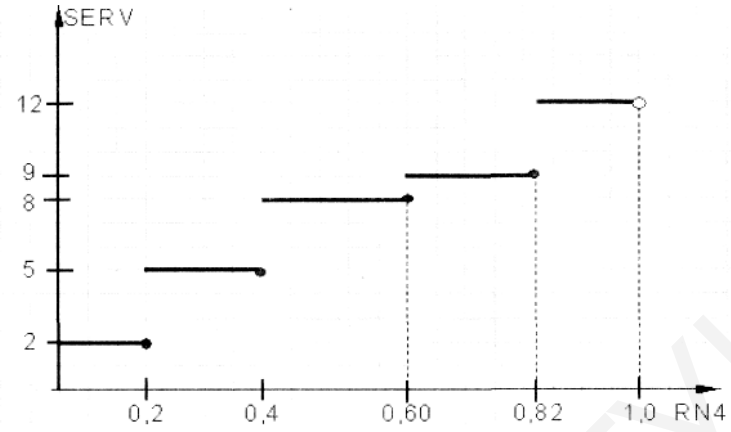

Рисунок 6.3 – Графическая интерпретация дискретной функции

Особенности вычисления дискретных и непрерывных GPSS-функций:

1. В начальной фазе выполняемые действия при вычислении дискретной и непрерывной функций одинаковы. При обращении к функции определяется значение ее аргумента. Потом просматривается упорядоченный ряд значений X1<X2<...<Xi<...<Xn для определения интервала, в который попало значение аргумента (пусть это будет интервал между точками i–1 и i).

2. Если функция дискретная, то второй элемент соответствующей пары Xi, Yj является значением функции. Если функция непрерывная, выполняется линейная интерполяция для пары точек i–1 и i, находящихся на концах интервала значений функции, в который попало значение аргумента. Целая часть результата интерполяции и является значением функции.

3. Если значение аргумента функции больше значения координаты Xn последней точки, то в обоих случаях (дискретной и непрерывной функции) значениями функции являются значения Yn.

Табличная числовая функция определяется так, что аргументы функции образуют непрерывный ряд натуральных чисел. Аргумент функции используется для прямого обращения к массиву значений функции.

Пример:

```
FUN1        FUNCTION        P1,L5
1,10/2,12/3,5/4,20/5,25
```

Дискретная атрибутивная функция подобна дискретной числовой с тем отличием, что значением функции может быть любой СЧА.

Пример:

```
FUN2        FUNCTION        X$CAR,E4
21,V$RED/25,X35/30,10/36,S$RED
```

Значение функции FUN2 вычисляется по правилу вычисления значения дискретной числовой функции:

X$CAR<=21 FN$FUN2=V$RED
22<=X$CAR<=25 FN$FUN2=X35
26<=X$CAR<=30 FN$FUN2=10
31<=X$CAR<=36 FN$FUN2= S$RED

Табличная атрибутивная функция подобна табличной числовой с тем отличием, что значением функции может быть любой СЧА.

Пример:

```
FUN3        FUNCTION        P1,M5
1,V1/2,P12/3,X35/4,Q$QUICK/5,99
```

Матрицы ячеек сохраняемых величин

С матрицами связан стандартный числовой атрибут MXj(m, n) – значение, записанное в строке m и в столбце n матрицы j или MX$<имя матрицы> (m, n), если матрица имеет символьное имя.

Оператор описания матрицы MATRIX

В GPSS World максимальная размерность матрицы равна 6. Каждая матрица должна быть объявлена до ее использования, т. е. должна иметь оператор описания MATRIX.

Формат оператора MATRIX:

<имя> MATRIX ,< B>,<C>,<D>,<E>,<F>,<G>

Поля операндов имеют следующий смысл:

- <A> – не используется;
- <B> – максимальное значение индекса для первой размерности;
- <C> – максимальное значение индекса для второй размерности;
- <D> – максимальное значение индекса для третьей размерности;
- <E> – максимальное значение индекса для четвертой размерности;
- <F> – максимальное значение индекса для пятой размерности;
- <G> – максимальное значение индекса для шестой размерности.

Пример:

```
MATR MATRIX    ,10,5
```

Блок MSAVEVALUE

Используется для записи значений в матрицы, а также для увеличения или уменьшения значений элементов матриц.

Формат блока MSAVEVALUE:

MSAVEVALUE <A>[±],<B>,<C>,<D>

Поля операндов имеют следующий смысл:

- <A> – имя матрицы;
- <B> – имя строки матрицы;
- <C> – имя столбца матрицы;
- <D> – величина, используемая для модификации.

Подобно блоку SAVEVALUE блок MSAVEVALUE может быть использован как в режиме замещения величины, так и в режиме увеличения или уменьшения.

Когда транзакт входит в блок MSAVEVALUE, анализируется операнд <A> и выполняется поиск матрицы с указанным именем. Если матрица не найдена, возникает ошибка. Соответствующий элемент матрицы определяется содержимым операндов <B> и <C>. Если такого элемента не существует, то также возникает ошибка.

Пример:

MSAVEVALUE MATR-,8,4,FN$COSTS

При входе транзакта в блок MSAVEVALUE величина ячейки матрицы MATR на пересечении восьмой строки и четвертого столбца (MX$MATR(8,4)) уменьшается на значение функции FN$COSTS.

Блок SELECT (выбрать)

Выбор элементов по их состояниям. Оператор SELECT может использоваться в следующих режимах:

1) Режим оператора отношения.

Формат оператора:

SELECT X A, B, C, D, E, F

Поля операндов имеют следующий смысл:

- <X> – вспомогательный внутренний операнд – оператор отношения, определяющий способ сравнения СЧА, указанного операндом <E>, и значения операнда <D> (принимает значения, аналогичные значениям такого же операнда блока TEST);
- <A> – имя или номер параметра, в который записывается номер члена группы, отвечающего установленному условию атрибута, обязательный параметр;
- <B> – наименьший номер (имя) из множества членов просматриваемой группы, обязательный параметр;
- <C> – наибольший номер (имя) из множества членов просматриваемой группы, обязательный параметр;
- <D> – значение, с которым должен сравниваться атрибут, указанный в операнде E, обязательный параметр;
- <E> – групповое имя стандартного числового атрибута, обязательный параметр;
- <F> – указывает номер блока, в который передается транзакт, если ни один член группы не отвечает установленному условию, необязательный параметр.

Примеры:

1) SELECT E 7, 1, 1, 0, F.

При поступлении транзакта в этот блок проверяется, равен ли нулю стандартный числовой атрибут F (Facility) устройства с номером 1, т. е. свободно ли устройство; если устройство свободно, то его номер заносится в параметр с но-

мером 7 данного транзакта и транзакт переходит к следующему блоку; в противном случае номер устройства не заносится в параметр транзакта.

2) SELECT L Packet, 2, 6, 25, SM, EXI_T2.

Проверяется, меньше ли 25 стандартный числовой атрибут SM (Storage) устройств с номерами от двух до шести, т. е. меньше ли 25 максимально занятая емкость многоканального устройства; номер первого многоканального устройства, для которого выполняется заданное условие, заносится в параметр с именем Packet данного транзакта и транзакт переходит к следующему блоку; в противном случае, если ни для одного из проверяемых многоканальных устройств не выполняется заданное условие, транзакт переходит к блоку с именем EXI_T2.

2) Режим MIN или MAX.

Формат оператора:

SELECT X A, B, C,, E.

Поля операндов имеют следующий смысл:

- <X> – внутренний вспомогательный операнд. Он является оператором MIN или MAX в зависимости от того, определяется ли элемент, имеющий минимальное или максимальное значение атрибута;
- <A> – номер параметра, значением которого становится номер соответствующего элемента, имеющего минимальное или максимальное значение атрибута, обязательный параметр;
- <B> – нижняя граница номеров членов соответствующей группы, обязательный параметр;
- <C> – верхняя граница номеров членов соответствующей группы, обязательный параметр;
- <E> – групповое имя стандартного числового атрибута, обязательный параметр.

Оператор count

Для определения количества объектов (из заданного множества объектов), удовлетворяющих заданному условию, используется оператор COUNT, он относится к блокам, имеющим дополнительные коды.

Формат оператора:

COUNT X A, B, C, D, E.

Поля операндов имеют следующий смысл:

- <X> – вспомогательный внутренний операнд – оператор отношения, определяющий способ сравнения СЧА, указанного операндом <E> и значение операнда <D>. Для устройств – U, NU, I, NI, FV, FNV; для памяти – SE, SNE, SF, SNF, SV, SNV; для переключателей – LR, LC; для операций сравнения – G, GE, L, LE, E, NE;
- <A> – не имеет значения по умолчанию, определяет имя или номер параметра с указанием формата (PF, PH, PB), используемого для организации счетчика числа объектов;

– <B> – нижняя граница диапазона изменения номеров объектов, для которых проверяется заданное условие, обязательный параметр;

– <C> – верхняя граница диапазона изменения номеров объектов, для которых проверяется заданное условие, обязательный параметр;

- <D> – не используется для устройств, памяти и переключателей, для операции сравнения представляет собой СЧА, значение которого сравнивается со значением СЧА объекта, указанного в поле E; в тех случаях, когда сравнение истинно, вычисленное значение присваивается параметру, определенному операндом <A>;

– <E> – не имеет значения по умолчанию, не используется для устройств, памяти, переключателей, определяет атрибуты анализируемых объектов, может использоваться любой одноразмерный СЧА, кроме матричных. При использовании устройств, памяти, переключателей поля операндов D, E должны быть пустыми.

Примеры:

1) COUNT SF 5PH,10,20

В результате выполнения данного оператора количество заполненной памяти из множества выбора с номерами от 10 до 20 будет занесено в параметр 5 формата «байт» транзакта.

2) COUNT LE 1PB, 1, 5, X10, FC

Операция сравнения (при этом поля <D> и <E> всегда должны быть заданы). На множестве устройств с номерами 1–5 подсчитывается количество устройств, у которых счетчик числа входов (FC) меньше либо равен текущему значению ячейки 10 формата «слово». Полученное число записывается в параметр 1 формата «байт», вошедшего в блок транзакта.

Косвенная адресация

Идея косвенной адресации состоит в том, что каждый транзакт в некотором своем параметре содержит номер того или иного объекта, а в операндах блоков, адресующихся к объектам, записывается ссылка на этот параметр транзакта. Например, *SAVEVALUE* 1, x*P2 означает: поместить в ячейку с номером 1 значение, содержащееся в ячейке, номер которой определяется значением параметра 2 транзакта.

Под косвенной адресацией будем понимать ссылку на СЧА некоторого объекта модели, номер которого записан в одном из параметров транзакта. При использовании косвенной адресации имя объекта модели (очереди, устройства и т. д.) должно быть числовым. Ссылка на СЧА имеет следующий вид:

имя_СЧА*номер_параметра.

Пример:

```
BBB   FUNCTION   *4,D6
1,2/2,5/3,11/4,20/5,18/6,12/7,9.
```

Дискретная (D) функция BBB задана шестью узловыми точками, аргумент – параметр 4 транзакта, вызвавшего обращение к функции BBB.

Здесь аргумент задан с использованием косвенной адресации, признаком которой является символ «*», т. е. запись *4 означает, что аргументом является величина, указанная в параметре 4 транзакта, вызвавшего функцию (в данном примере можно было бы использовать равноценную запись *p4). В общем случае косвенная адресация выполняется путем записи операнда в виде СЧА*СЧА, например S*X2, что означает емкость памяти, занятая в накопителе, именем которого является значение переменной X2, или Q*p5 – длина очереди с именем, записанным в параметре 5 транзакта.

Практические задания

Задание 1. *Выбор направления движения.*

Базовые операторы: *select, косвенная адресация(*), matrix, msavevalue.*

В цехе имеется *a* станков. Заготовки поступают через определенное количество минут в соответствии с законом распределения – *b* минут. Каждая следующая заготовка направляется на станок, имеющий минимальную очередь, причем время движения заготовки от места поступления до станка равно *5n* (*n* – номер станка). Реализовать время перехода через обращение к матрице. Время обработки на станке зависит от длины очереди и определяется функцией *обработка=c–d*очередь* (при этом максимальное повышение производительности равно 10*d*). Промоделировать работу участка в течение месяца. Предложить варианты повышения эффективности. Задания выполняются согласно индивидуальным вариантам (таблица 6.1).

Таблица 6.1

Вариант	*a*	*b*	*c*	*d*
1	3	Normal(12,1)	40–46	1
2	3	Exponential(7)	22–26	0,5
3	3	Poisson(10)	41–43	1,2
4	4	Normal(8,1)	40–44	1,1
5	4	Exponential(10)	44–48	1
6	4	Poisson(11)	52–56	1,4
7	5	Normal(10,1)	59–61	1,2
8	5	Exponential(8)	42–50	1
9	5	Poisson(12)	62–66	0,8
10	6	Normal(12,1)	80–86	2
11	6	Exponential(9)	60–64	1,1
12	6	Poisson(10)	70–74	1,4
13	6	Normal(10,1)	62–64	0,4
14	5	Exponential(11)	62–64	0,8
15	4	Poisson(9)	42–44	1

Задание 2. *Подсчет устройств, удовлетворяющих критерию*.

Базовые операторы: *select, count, matrix, msavevalue.*

На заводе проходит испытание новое устройство покраски деталей, состоящее из трех красящих и одного вспомогательного манипуляторов. В цехе установлены четыре устройства. Процесс организован следующим образом: вспомогательный манипулятор устанавливает деталь в одну из трех рабочих позиций (*a* минут), затем красящий манипулятор наносит краску (*b* минут), вспомогательный убирает готовую деталь (*c* минут). Детали поступают каждые *d* минут и попадают на устройство с наименьшим числом обрабатываемых в данный момент деталей. Кроме того, каждый час происходит контроль одного из параметров (*e*). Результаты проверок заносятся в таблицу. Промоделировать работу цеха в течение дня (2 смены). Задания выполняются согласно индивидуальным вариантам (таблица 6.2).

Таблица 6.2

Вариант	*a*	*b*	*c*	*d*	*e*
1	2	5	3	6–8	Число свободных красящих манипуляторов
2	2	1	3	7–9	Число занятых красящих манипуляторов
3	1	4	3	5–6	Число красящих манипуляторов с загрузкой меньше 60 %
4	5	5	2	5–7	Минимальная загрузка красящих манипуляторов
5	3	2	1	6–10	Число свободных красящих манипуляторов
6	3	2	3	7–10	Число занятых красящих манипуляторов
7	1	1	3	8–10	Число красящих манипуляторов с загрузкой меньше 60 %
8	5	3	5	8–9	Минимальная загрузка красящих манипуляторов
9	3	4	1	8–11	Число свободных красящих манипуляторов
10	5	1	3	10–12	Число занятых красящих манипуляторов
11	2	3	2	4–6	Число красящих манипуляторов с загрузкой меньше 60 %
12	1	4	3	4–8	Минимальная загрузка красящих манипуляторов
13	2	2	1	5–9	Число свободных красящих манипуляторов
14	4	2	3	6–9	Число занятых красящих манипуляторов
15	1	3	2	6–8	Число красящих манипуляторов с загрузкой меньше 60 %

Контрольные вопросы

1. Какие типы функций используются в GPSS?
2. Какие действия необходимо сделать для определения GPSS-функции?
3. В чем заключаются особенности вычисления дискретных и непрерывных GPSS-функций?
4. Какие операторы в GPSS поддерживают работу с матрицами?
5. В каких режимах может применяться блок MSAVEVALUE?
6. В каких режимах может применяться блок SELECT?
7. Какой смысл имеет внутренний операнд <X> в блоке COUNT?
8. Что понимается под косвенной адресацией в GPSS?
9. Какой формат имеет блок SELECT в режиме MIN/MAX?
10. Как получить длину очереди с именем, записанным в параметре 1 транзакта?

ЛАБОРАТОРНАЯ РАБОТА №7

УМЕНЬШЕНИЕ ЧИСЛА ОБЪЕКТОВ В МОДЕЛИ МЕТОДОМ КОСВЕННОЙ АДРЕСАЦИИ. ОБРАБОТКА ОДНОВРЕМЕННЫХ СООБЩЕНИЙ

Цель работы – *использование косвенной адресации, организация обработки временных узлов.*

Теоретические сведения

Арифметические переменные. Оператор VARIABLE

Арифметические переменные аналогичны арифметическим выражениям в алгоритмических языках. Арифметическая переменная с фиксированной точкой задается оператором VARIABLE, называемым оператором описания переменной, который содержит арифметическое выражение. Формат оператора VARIABLE:

<имя> VARIABLE <A>

В поле операнда <A> записывается выражение, которое используется для вычисления значения переменной. При обращении к переменной используется СЧА V<номер переменной> или V$<имя переменной>.

Пример:

RSL VARIABLE QT$WAITL+3-FN$DSTRB#P7

Оператор описания VARIABLE определяет арифметическую переменную RSL.

При любом обращении к переменной RSL (т. е. к СЧА V$RSL) ее значение вычисляется как текущая длина очереди WAlTL (QT$WAITL – СЧА регистратора очереди) плюс константа 3 и минус произведение значения функции DSTRB (FN$DSTRB – СЧА функции) на значение параметра седьмого транзакта, обрабатываемого в данный момент (Pj – СЧА транзакта).

Перед выполнением любой арифметической операции определяется значение каждого операнда и выделяется его целая часть. Константы без знака считаются положительными числами. Пробелы между символами в выражениях не допускаются.

В выражении, заданном с помощью арифметической переменной, могут быть использованы любые СЧА, функции и другие арифметические переменные. Запрещается использование самой вычисляемой переменной, а также переменных со знаком, т. к. знаки в данном случае рассматриваются как арифметические операции.

Система моделирования GPSS допускает использование скобок в выражениях, задающих арифметические переменные (для группировки членов или для обозначения операции умножения).

В GPSS World выражения, записанные в круглых скобках, обрабатываются вычислительной процедурой встроенного алгоритмического языка PLUS. Поэтому их можно использовать в качестве операндов блоков и операторов языка GPSS. Например, выражение, описанное в примере, может быть использовано таким образом: ADVANCE (QT$WAITL+3-FN$DSTRB#P7).

Стандартный числовой атрибут V$<имя переменной> используется для обращения к значениям как арифметических переменных, так и переменных с плавающей точкой. Способ вычисления переменной определяется оператором описания этой переменной.

Булевы переменные

Булевы переменные позволяют принимать решения в зависимости от значений СЧА и состояния объектов GPSS, при этом используется только одно выражение. Булевы переменные – это логические выражения, состоящие из различных СЧА и (или) других булевых переменных. В булевой переменной проверяется одно или несколько логических условий. Результатом проверки является единица (истина), если условия выполняются, и ноль (ложь) – в противном случае.

Стандартный числовой атрибут BV$<имя переменной> применяется для обращения к значениям булевых переменных.

При описании булевых переменных используются три типа операторов: логические, булевы и операторы отношения (условные операторы).

Логические операторы связаны с такими ресурсами, как устройства, накопители и логические ключи. Они применяются для определения состояния данных объектов. Логические операторы, используемые в GPSS, представлены в таблице 7.1.

Таблица 7.1

Логические операторы	Значение оператора, отражающее состояние ресурса
FUj или Fj	Равно 1, если устройство j занято или обслуживает прерывание, иначе – 0
FNUj	Равно 1, если устройство j не занято и не обслуживает прерывание, иначе – 0
Ij	Равно 1, если устройство j обслуживает прерывание, иначе – 0
NIj	Равно 1, если устройство j не обслуживает прерывание, иначе – 0
NUj	Равно 1, если устройство j не используется, в противном случае – 0
UJ	Равно 1, если устройство j используется, иначе – 0
SFJ	Равно 1, если накопитель j заполнен, иначе – 0
SNFj	Равно 1, если накопитель j не заполнен, иначе – 0
SEj	Равно 1, если накопитель j пуст, иначе – 0
SNEj	Равно 1, если накопитель j не пуст, иначе – 0
SVj	Равно 1, если накопитель j находится в состоянии использования, иначе – 0
SNVJ	Равно 1, если накопитель j не используется, иначе – 0
LRj	Равно 1, если логический ключ j выключен, иначе – 0
LSj	Равно 1, если логический ключ j включен, иначе – 0

Пример:

VAR1 BVARIABLE FNU$CHAN

Переменная VAR1 принимает значение «истина», если устройство с именем CHAN не занято и не обслуживает прерывание.

Списки пользователя. Блоки LINK и UNLINK

Блок LINK (внести в список)

Собирает транзакты из списка текущих событий и помещает их в список пользователя. Таким образом, интерпретатор их не просматривает и не перемещает по блокам модели до тех пор, пока пользователь не возвратит их в модель.

Формат блока LINK:

LINK <A>,<B>,[<C>]

Поля операндов имеют следующий смысл:

<A> – задает номер или имя списка, в который будет помещен транзакт. Операнд <A> может быть положительным целым числом, именем, СЧА;

<B> – задает алгоритм упорядочивания списка, может быть LIFO, FIFO, целым числом, СЧА. Допустимые значения операнда <B>:

- FIFO – вошедший транзакт помещается в конец списка;
- LIFO – вошедший транзакт помещается в начало списка;
- номер параметра – входящие в список транзакты располагаются в соответствии со значением указанного параметра;
- PR – приоритет транзакта (транзакт помещается в список в соответствии с приоритетом);
- M1 – время нахождения транзакта в модели (транзакт помещается в список в соответствии с временем нахождения транзакта в модели);

– <C> – параметр, в котором запоминаются номера копий транзактов. Операнд <C> указывает альтернативный выход, который используется при описании разных ситуаций, возникающих в очередях, может быть именем, положительным целым числом, СЧА.

Если операнд <C> не задан, индикатор, связанный с заданным списком, устанавливается в положение «1». Это приводит к тому, что все транзакты, безусловно входящие в блок, заносятся в список, определенный операндом <A>, в порядке, который задан операндом <B>.

Если операнд <C> задан, проверяется индикатор списка. Если индикатор списка установлен в положение «1», вошедший транзакт заносится в список в порядке, заданном операндом <B>. Если же индикатор списка установлен в положение «0», он переводится в положение «1», и вошедший транзакт перемещается к блоку, заданному в операнде <C>.

Пример:

LINK LIST,FIFO.

Транзакт, вошедший в блок, помещается в конец списка с именем LIST.

Блок UNLINK (вывести из списка)

Блок UNLINK удаляет транзакты из списка. После этого интерпретатор GPSS возобновляет их движение по модели.

Формат блока UNLINK:

UNLINK [<X>] <A>,<B>[,<C>][,<D>][,<E>][,<F>]

Операнд <A> задает список пользователя (СП), из которого удаляются один или несколько транзактов. Операнд <A> может быть именем, положительным целым числом, СЧА.

В операнде <B> указывается номер блока, к которому переходят удаляемые из списка транзакты. Операнд <B> может быть именем, положительным целым числом, СЧА.

Операнд <C> задает число транзактов, удаляемых из списка (счетчик удалений). Операнд <C> может быть именем, положительным целым числом, СЧА или «ALL» (означает удаление всех транзактов).

Операнд <D> может быть именем, целым числом, СЧА или «BACK». Действия, выполняемые при вхождении транзакта в блок UNLINK, зависят от того, на что ссылается операнд <D>. В операнде <D> могут быть указаны номер параметра, булева переменная или слово «BACK».

Номер параметра. Если операнд <E> пропущен, значение заданного параметра вошедшего транзакта сравнивается со значением этого же параметра транзактов списка. Если <E> не пропущен, значение заданного параметра транзактов списка сравнивается со значением СЧА из операнда <E>. В обоих случаях транзакты, удовлетворяющие заданному отношению, будут удалены из списка и направлены в блок, указанный в операнде <B>.

Булева переменная BVj вычисляется отдельно для каждого транзакта из списка. Если для транзакта значение BVj=l, он удаляется из списка (количество удаляемых транзактов не может превышать значения операнда <C>). Если BVj=0 для всех транзактов списка, вошедший транзакт пытается переместиться в блок, заданный в операнде <F>. Если операнд <F> пропущен, транзакт пытается перейти в следующий по номеру блок. Если в операнде <D> задана булева переменная, операнд <E> должен быть пустым. Если булева переменная BVj имеет ссылку на какой-либо параметр, то эта ссылка относится к параметрам транзактов из списка, а не к входящему в блок UNLINK транзакту.

Слово «BACK». Из указанного списка, начиная с его конца, будет исключено столько транзактов, сколько задано операндом <C>. Операнд <E> в этом случае должен быть пустым.

Операнд <E> содержит СЧА, значение которого сравнивается со значением параметра транзактов списка (номер параметра указан в операнде <D>). Операнд <E> может быть именем, целым числом, СЧА.

Операнд <F> задает номер следующего блока для того транзакта, который входит в блок UNLINK в случаях, когда соответствующий список пустой или не выполнено заданное отношение, или же указанная в операнде <D> булева переменная равна нулю для всех транзактов списка (т. е. в случае, когда из

списка нельзя ничего удалить). Операнд <F> может быть именем, положительным целым числом, СЧА.

Операторы отношения, которые записываются во вспомогательном операнде <X>, определяют, какое условие (отношение) будет рассматриваться. Если этот оператор не задан, предполагается отношение равенства E. Операторы отношения задаются аналогично оператором отношения в блоке TEST.

Пример:

UNLINK LIST,MET,1

Первый транзакт из списка с именем LIST помещается в блок с меткой MET. Он заносится в список текущих событий после транзактов с таким же приоритетом. Транзакт, вошедший в блок UNLINK, переходит в следующий блок.

Практические задания

Задание 1. *Уменьшение числа объектов в модели методом косвенной адресации.*

Базовые операторы: *function, variable, table, qtable, priority.*

На вход многоканальной системы с тремя каналами обслуживания поступает экспоненциальный поток заявок со средним интервалом поступления *a* единиц модельного времени. Каждая заявка с равной вероятностью 0,2 относится к одному из пяти видов: 1, 2, 3, 4 или 5. Среднее время обслуживания заявок каждого типа составляет соответственно *b, c, d, e, f* единиц модельного времени. Чем меньше среднее время обслуживания заявки, тем выше ее приоритет. Необходимо построить модель, позволяющую оценить средние значения времени ожидания заявок каждого вида, а также распределения общего времени ожидания в очереди и общего времени пребывания в системе. Задания выполняются согласно индивидуальным вариантам (таблица 7.2).

Таблица 7.2

Вариант	*a*	*b*	*c*	*d*	*e*	*f*
1	140	90	100	110	120	130
2	130	120	110	100	90	90
3	150	140	100	90	110	120
4	145	120	100	110	90	130
5	180	130	140	100	80	90
6	165	100	90	140	100	110
7	120	90	110	110	110	120
8	130	80	90	100	110	120
9	160	130	120	100	110	90
10	155	150	100	130	90	120
11	140	140	130	90	110	120
12	190	150	90	120	130	130
13	170	100	110	150	90	100
14	140	120	110	100	90	80
15	135	90	110	100	120	130

Задание 2. *Обработка временных узлов для моделей со списками пользователя.*

Базовые операторы: *variable, transfer, assign, priority, preempt, gate, test, link, unlink.*

На вычислительный комплекс коммутации сообщений поступают сообщения от трех абонентов и далее передаются по двум каналам передачи данных со скоростью 1кбит/с. Длительности интервалов между сообщениями от каждого абонента распределены по экспоненциальному закону с интенсивностью λ 1/с. Сообщения равновероятно могут принадлежать одной из двух категорий: команды или иная информация. Команды обладают абсолютным приоритетом. Длины сообщений-команд равномерно распределены в интервале 1400–6000 байт. Длины остальных сообщений (иная информация) распределены по нормальному закону с параметрами *m* и *n* байт. Для хранения сообщений, ожидающих обработки в комплексе, предусмотрен накопитель емкостью 1 Мбайт.

Разработать имитационную модель с целью исследования в течение 1 ч функционирования вычислительного комплекса зависимости емкости накопителя от интенсивности поступления сообщений. Данная модель должна обеспечивать вероятность передачи сообщений-команд не менее *a*, а иной информации – не менее *b*. Задания выполняются согласно индивидуальным вариантам (таблица 7.3).

Таблица 7.3

Вариант	*a*	*b*	*m*(*10^3)	*n*(*10^2)
1	0,9	0,7	2	3
2	0,8	0,6	3	4
3	0,7	0,6	1	2
4	0,6	0,5	1	2
5	0,9	0,6	2	3
6	0,8	0,7	3	4
7	0,7	0,5	3	4
8	0,6	0,4	2	3
9	0,9	0,5	3	4
10	0,8	0,6	2	3
11	0,7	0,4	2	3
12	0,6	0,3	2	3
13	0,9	0,6	2	3
14	0,8	0,5	1	2
15	0,7	0,4	1	2

Контрольные вопросы

1. В каких случаях целесообразно использовать косвенную адресацию при имитационном моделировании систем?

2. Что такое временной узел?

3. Для чего служит оператор PREEMPT?

4. Когда применяются операторы LINK, UNLINK?

5. Когда может возникнуть параллельная адресация?

6. Какие дисциплины обслуживания могут быть применены в операторе LINK?

7. Какой смысл имеет операнд <B> в блоке LINK?

8. Куда помещаются транзакты, удаленные из списка пользователя?

9. Какую статистику можно получить по списку пользователя?

10. Как исключить из списка пользователя необходимое количество транзактов?

ЛАБОРАТОРНАЯ РАБОТА №8

МОДЕЛИРОВАНИЕ ГИБКИХ УЧАСТКОВ ШТАМПОВКИ

Цель работы – *исследование гибких участков штамповки, разработка алгоритмов функционирования участков и оптимизация их работы посредством моделирования и анализа.*

Практическое задание

Разработать алгоритм имитационного моделирования и реализовать имитационную модель участков штамповки деталей из штучных заготовок, компоновочные схемы которых представлены на рисунке 8.1.

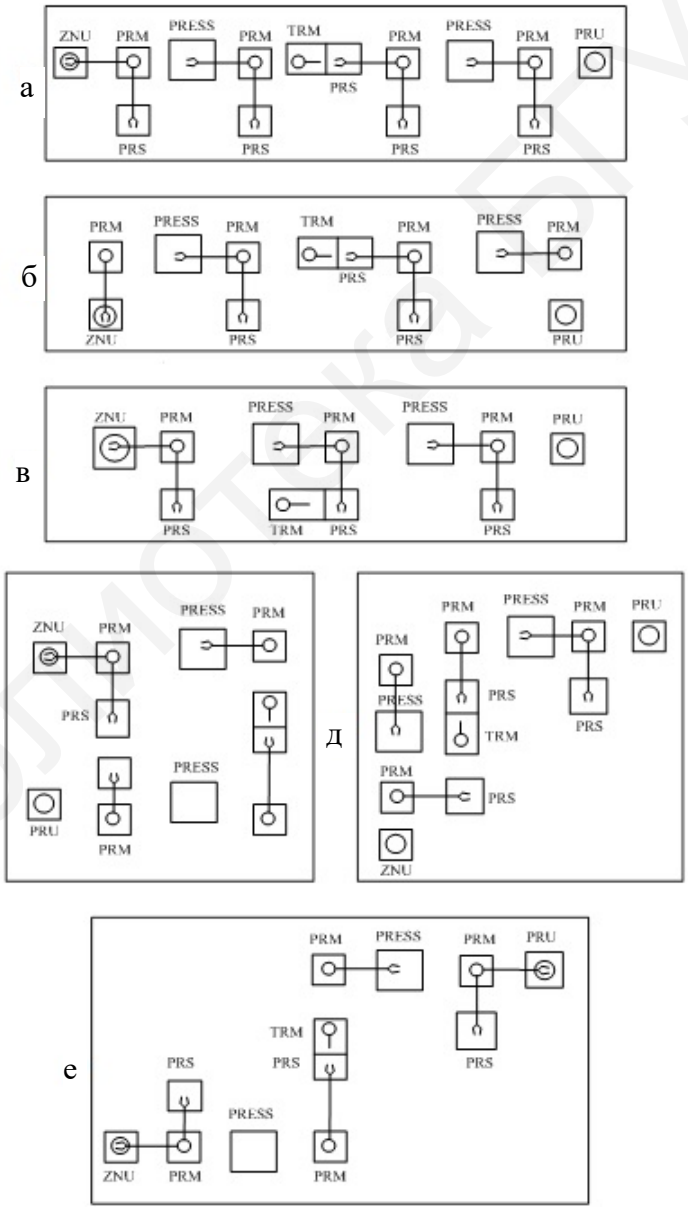

Рисунок 8.1 – Компоновочные схемы двухпрессовых участков

Двухпрессовый участок содержит два пресса PRESS, четырехпозиционное поворотное загрузочное устройство ZNU, приемное устройство PRU, промежуточные приемные столы PRS, манипуляторы PRM, на некоторых участках между прессами находится транспортный манипулятор TRM.

Движение заготовок, полуфабрикатов и деталей осуществляется слева направо. Продолжительность цикла работы манипулятора (опустить руку, взять заготовку, поднять руку, повернуться на 90°, опустить руку, положить заготовку, поднять руку, возвратиться в исходное положение) составляет k секунд. Продолжительность поворота загрузочного устройства на 90° – m секунд, перегрузка приемного устройства после его заполнения – n секунд, рабочего цикла прессования детали – q секунд.

Составить алгоритм моделирования работы участка согласно варианту (таблица 8.1), по нему реализовать программу имитационной модели, учитывая, что работа участка составила p смен при коэффициенте использования рабочего времени, равном 0,9. Оценить производительность участков и загрузку оборудования, а именно: количество отштампованных деталей, среднее время изготовления одной детали, коэффициенты загрузки основного и вспомогательного оборудования. Предложить варианты повышения производительности участков. Задания выполняются согласно индивидуальным вариантам (см. таблицу 8.1).

Таблица 8.1

Вариант	k	l	m	n	p	q	Рисунок
1	3	300	10	150	1	1	8.1, а
2	3,5	400	15	160	2	1,5	8.1, б
3	4	450	12	140	3	1,2	8.1, в
4	3	300	10	150	1	1	8.1, г
5	3,5	400	15	160	2	1,5	8.1, д
6	4	450	12	140	3	1,2	8.1, е
7	3	300	10	150	1	1	8.1, б
8	3,5	400	15	160	2	1,5	8.1, в
9	4	450	12	140	3	1,2	8.1, г
10	3	300	10	150	1	1	8.1, д
11	3,5	400	15	160	2	1,5	8.1, е
12	4	450	12	140	3	1,2	8.1, а
13	3	300	10	150	1	1	8.1, в
14	3,5	400	15	160	2	1,5	8.1, г
15	4	450	12	140	3	1,2	8.1, д

Контрольные вопросы

1. Как организовать моделирование счетчиков при заполнении тары?
2. Как уточнить коэффициент использования рабочего времени, чтобы не было незавершенного производства и максимально использовалось рабочее время?

3. Каким образом повысить коэффициенты использования менее загруженного оборудования?

4. Как зависит производительность участков от номенклатуры изделий?

5. Как изменить имитационную модель при учете задержки на замену штампов в случаях перехода к штамповке новой партии (типа) изделий?

6. Что такое циклограмма? Каковы правила ее составления?

7. Что такое система массового обслуживания? Назовите ее характеристики.

8. В чем состоят особенности разомкнутой системы массового обслуживания?

9. В чем состоят особенности стохастических сетей?

10. Назовите характеристики замкнутой системы массового обслуживания.

ЛИТЕРАТУРА

1. Лукьянец, С. В. Электронный учебно-методический комплекс по дисциплине «Моделирование в проектировании сложных систем» для студ. спец. 1-53 01 07 «Информационные технологии и управление в технических системах» / С. В. Лукьянец, С. В. Снисаренко. – Минск : БГУИР, 2017.

2. Боев, В. Д. Моделирование систем. Инструментальные средства GPSS World : учеб. пособие / В. Д. Боев. – СПб. : БХВ-Петербург, 2004. – 368 с.

3. Кудрявцев, Е. М. GPSS World. Основы имитационного моделирования различных систем / Е. М. Кудрявцев. – М. : ДМК Пресс, 2004. – 320 с.

4. Томашевский, В. И. Имитационное моделирование в среде GPSS / В. И. Томашевский, Е. Г. Жданова. – М. : Бестселлер, 2003. – 416 с.

Св. план 2020, поз. 20

Учебное издание

Захарьев Вадим Анатольевич
Снисаренко Светлана Валерьевна

МОДЕЛИРОВАНИЕ В ПРОЕКТИРОВАНИИ СЛОЖНЫХ СИСТЕМ. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Редактор *М. А. Зайцева*
Корректор *Е. Н. Батурчик*
Компьютерная правка, оригинал-макет *М. В. Касабуцкий*

Подписано в печать 07.08.2020. Формат 60×84 1/16. Бумага офсетная. Гарнитура «Таймс».
Отпечатано на ризографе. Усл. печ. л. 3,95. Уч.-изд. л. 4,0. Тираж 50 экз. Заказ 101.

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий №1/238 от 24.03.2014,
№2/113 от 07.04.2014, №3/615 от 07.04.2014.
Ул. П. Бровки, 6, 220013, г. Минск