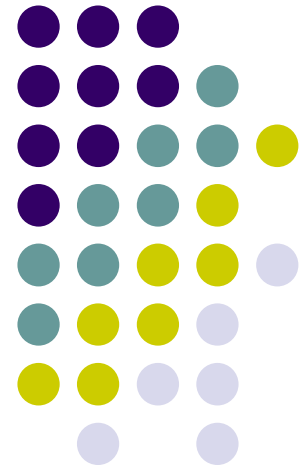


## 6. Применение языков моделирования при реализации имитационных моделей

---

Обзор инструментальных  
средств *GPSS World*





## Общие сведения о GPSS World

**GPSS World** (*General Purpose Simulation System*) – общецелевая система моделирования.

Разработана компанией *Minuteman* (США).

Является развитием системы **GPSS/PC** – пакета моделирования дискретных систем, но приобрела комбинированный характер (способна моделировать как дискретные, так и непрерывные процессы).

Учебная версия – на портале

***[www.minutemansoftware.com/download](http://www.minutemansoftware.com/download)***



## Объекты системы моделирования *GPSS World*

В системе моделирования ***GPSS World*** 4 вида объектов:

- модель (Model);
- процесс моделирования (Simulation);
- отчет (Report);
- текстовый объект (Text File).



Модель разрабатывается на языке **GPSS** и состоит из операторов.

Объект «Модель» создается при помощи встроенного текстового редактора.

Объект «Процесс моделирования» — результат трансляции модели.

Получается после выполнения команды меню *Create Simulation*.

Далее процесс моделирования запускается с помощью команд **GPSS**.



Объект «Отчет», как правило, автоматически создается по завершении моделирования.

Текстовый объект (текстовый файл ***GPSS World***) предназначен для разработки больших моделей и создания библиотеки исходных текстов.

Модель может быть разделена на наборы операторов (в отдельных текстовых файлах), а затем с помощью объекта «Процесс моделирования» собрана из них.



## Объекты языка моделирования *GPSS*

Модель представляется совокупностью элементов и логических правил их взаимодействия.

Набор логических правил ограничен и может быть описан небольшим числом стандартных операций.

Предполагается:

для моделируемых систем можно выделить набор абстрактных элементов, называемых *объектами*.



В составе системы **GPSS World** имеется программа-планировщик, которая выполняет функции:

- обеспечения продвижения по заданным разработчиком маршрутам динамических объектов, называемых *транзактами*;
- планирования событий, происходящих в модели, путем регистрации времени наступления каждого события и выполнения их в нарастающей временной последовательности;
- регистрации статистической информации о функционировании модели;
- продвижения модельного времени в процессе моделирования.



Объекты в моделируемой системе предназначены для различных целей.

Подразделяются на 7 категорий и 15 типов.

Не обязательно, чтобы в одной модели участвовали все типы объектов.

Необходимо только наличие *блоков* и *транзактов*.





## Объекты GPSS

Категории	Типы объектов
Динамическая	Транзакты
Операционная	Блоки
Аппаратная	Одноканальные устройства, накопители (многоканальные устройства), логические ключи
Вычислительная	Переменные, функции, генераторы случайных чисел
Статистическая	Очереди, таблицы
Запоминающая	Ячейки, матрицы ячеек
Группирующая	Числовые группы, группы транзактов, списки



*Транзакты* – динамические объекты, которые создаются в определенных точках модели, продвигаются планировщиком через блоки, а затем уничтожаются.

Являются аналогами единиц потоков в реальной системе.

Могут представлять различные элементы даже в одной системе.

Например: в модели функционирования ИВС транзакты могут представлять

- пакеты сообщений, подлежащие обработке,
- пакеты-подтверждения правильного приема,
- потоки отказов в аппаратных средствах.



С каждым транзактом могут связываться *параметры*.

Например: время обработки пакета ЦП.

Каждый транзакт может иметь любое число параметров.

Параметры нумеруются или им даются имена.

Транзактам может назначаться *приоритет*  
(определяет предпочтение, которое получает транзакт, когда он вместе с другими транзактами претендует на один ресурс).

Например: категория срочности при передаче сообщений.



Транзакты движутся от блока к блоку так, как движутся элементы, которые они представляют.

Каждое продвижение считается событием, которое должно происходить в конкретный момент времени (автоматически определяется планировщиком).



*Объекты аппаратной категории* – это абстрактные элементы, на которые может быть декомпозирована реальная система.

Воздействуя на эти объекты, транзакты могут изменять их состояние и влиять на движение других транзактов.



*Одноканальные устройства* (ОКУ) предназначены для имитации оборудования, которое в любой момент может быть занято только одним транзактом. Например: один канал передачи данных.

*Многоканальные устройства* (МКУ) предназначены для имитации оборудования, осуществляющего параллельную обработку. Они могут быть одновременно использованы несколькими транзактами.



Возможны ситуации:

ранее происходившие в системе события могут заблокировать или изменить движение транзактов и/или наступление следующих событий.

Например: один из каналов связи вышел из строя, и все заявки на передачу сообщений должны быть направлены на исправные каналы.

Для моделирования таких ситуаций используются *логические ключи*.

Транзакт может устанавливать эти ключи в положение «включено» или «выключено», после чего состояние ключей может проверяться другими транзактами для определения пути их следования.



*Операционные объекты (блоки)* задают логику функционирования модели и определяют пути движения транзактов между объектами аппаратной категории.

В блоках могут происходить события 4 категорий:

- создание или уничтожение транзактов;
- изменение числового атрибута транзакта;
- задержка транзакта на определенный период времени;
- изменение маршрута движения транзакта.





*Объекты запоминающей категории* обеспечивают обращения к сохраняемым значениям.

*Ячейки* и *матрицы ячеек* сохраняемых величин – для сохранения некоторой числовой информации.

Любой активный транзакт может произвести запись информации в эти объекты.

Впоследствии эту информацию может считать любой активный транзакт.

Матрицы могут иметь до 6 измерений.



К статическим объектам относятся *очереди* и *таблицы*.

Движение потока транзактов может быть задержано из-за недоступности устройств.

Например: ОКУ занято;

МКУ заполнено.

В этом случае транзакт ставится в *очередь*.

Учет очередей – одна из основных функций планировщика.



Планировщик автоматически накапливает стандартную статистику относительно устройств и очередей.

Кроме этого, пользователь может собирать дополнительную статистическую информацию, указав специальные точки в модели.

Для табулирования статистической информации – специальный объект – *таблица*.

Таблицы используются для получения выборочных распределений указанных случайных величин.



Каждая таблица включает частотные классы (диапазоны значений) и число попаданий конкретного числового атрибута в каждый класс.

Для каждой таблицы вычисляется математическое ожидание и среднеквадратическое отклонение.

В конце эксперимента с моделью результаты, содержащиеся в таблицах, выводятся в отчет.



*Списки* – объекты группирующей категории.

В списках хранятся транзакты в процессе моделирования.

В любой момент времени транзакт может находиться в одном из 5 видов списков:

- текущих событий;
- будущих событий;
- задержки ОКУ или МКУ;
- отложенных прерываний ОКУ;
- пользователя.



## Системные числовые атрибуты

Каждому объекту соответствуют атрибуты, описывающие его состояние в данный момент времени.

Они называются *системными числовыми атрибутами* (СЧА).

Доступны для использования в течение всего процесса моделирования.



Пример:

$PR$  – приоритет обрабатываемого в данный момент транзакта;

$P_i$  – значение  $i$ -го параметра активного транзакта.

Всего в **GPSS World** – более 50 СЧА.



## Блок-диаграммы

Модели систем в **GPSS** могут быть первично описаны в виде *блок-диаграмм*.

*Блок-диаграмма* представляет собой набор блоков с характерным очертанием, соединенных между собой линиями.

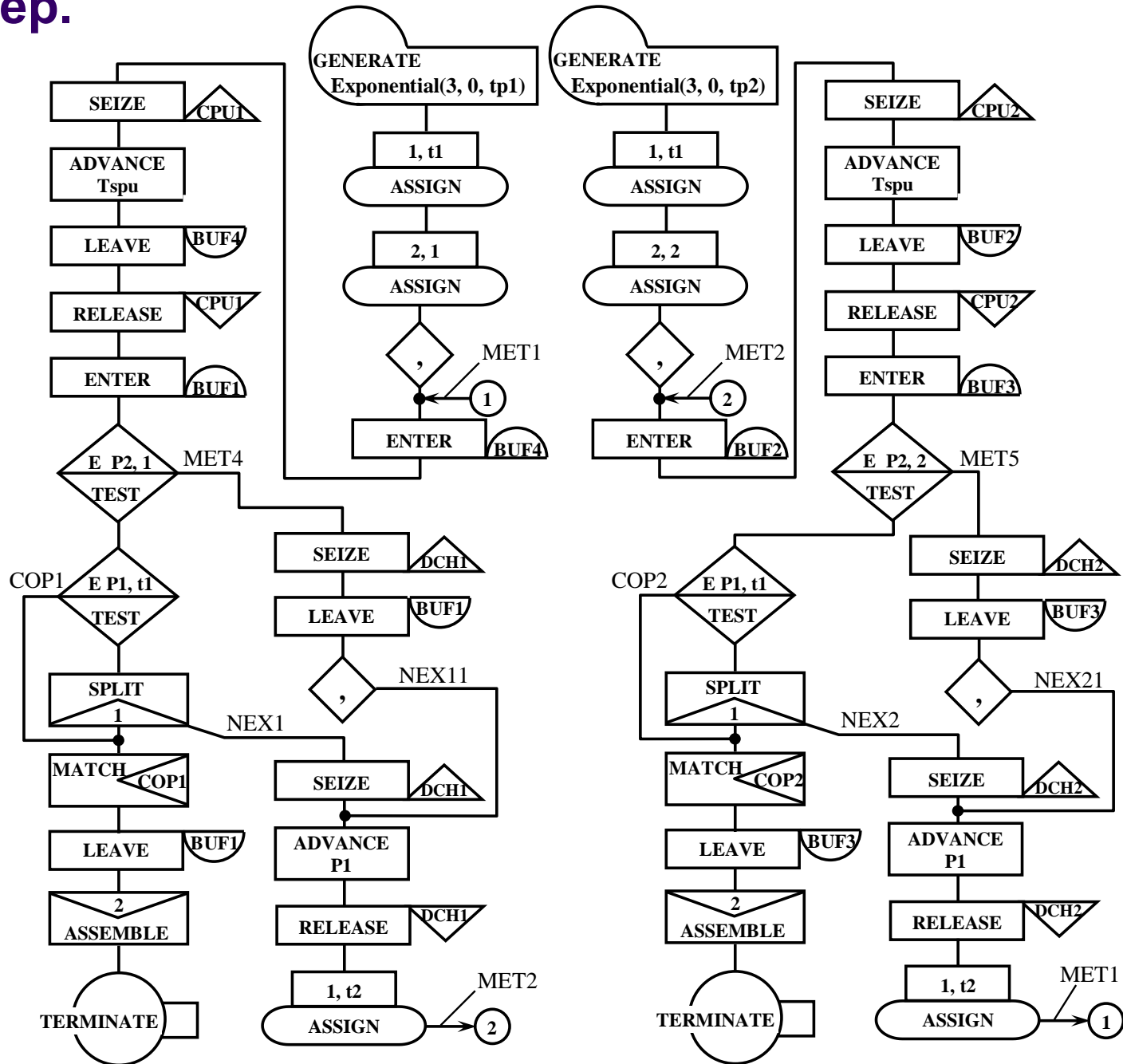
Вид каждого блока стандартен.

Конфигурация блок-диаграммы отражает направления, по которым происходит движение транзактов.

Работа модели состоит в перемещении транзактов от блока к блоку.



# Пример.





В начале моделирования в модели нет ни одного транзакта.

В процессе моделирования

- транзакты входят в модель в определенные моменты времени
- покидают модель в определенные моменты времени.

В общем случае в модели может существовать большое число транзактов, но в каждый момент времени движется только один транзакт.

Он перемещается от блока к блоку по пути, предписанному блок-диаграммой.



Каждый блок можно рассматривать как некоторую точку, в которой происходит обращение к подпрограмме.

При входе транзакта в блок вызывается соответствующая подпрограмма.



Продвижение транзакта происходит до тех пор, пока не произойдет одно из следующих событий:

- ❑ транзакт входит в блок, функцией которого является задержка транзакта на некоторое модельное время;
- ❑ транзакт входит в блок, функцией которого является удаление транзакта из модели;
- ❑ транзакт пытается войти в следующий блок (в соответствии с предписанной блок-диаграммой логикой), но блок отказывается принять этот транзакт  
транзакт остается в текущем блоке и повторяет попытки войти в следующий блок;  
когда условия в модели изменятся, одна из попыток может оказаться успешной, и транзакт продолжит свое движение.



Выполнение модели заключается в последовательном обращении к подпрограммам (следствие входа в определенные блоки двигающихся транзактов).

Блоки имеют счетчики, содержимое которых доступно разработчику через СЧА блоков:

**$W$**  – текущее число транзактов, находящихся в блоке;

**$N$**  – общее количество входов в блок.



Каждое продвижение транзакта в модели является *событием*.

Для поддержания правильной временной последовательности событий планировщик имеет *таймер* модельного времени, который автоматически корректируется в соответствии с логикой, предписанной моделью.



## Особенности таймера **GPSS**:

- регистрируются только положительные вещественные значения времени;
- единица модельного времени определяется разработчиком модели (все временные интервалы задаются в одних и тех же единицах);
- планировщик не анализирует состояние модели в каждый момент модельного времени, а продвигает таймер к моменту времени, когда должно произойти ближайшее следующее событие.



Значения таймера доступны разработчику через СЧА:

- **C1** – относительное модельное время;
- **AC1** – абсолютное модельное время.





Содержательное значение транзактов определяет разработчик модели.

Он устанавливает аналогию между транзактами и реальными объектами моделируемой системы.

Эта аналогия не указывается планировщиком **GPSS** (существует только в памяти разработчика модели).



## Структура модели на языке **GPSS**

Информация об объектах **GPSS** записывается в виде последовательности операторов.

Операторы модели в **GPSS World**:

- операторы **GPSS**;
- **PLUS**-операторы.

Операторы **GPSS**:

- операторы блоков (создают блоки);
- команды, которые не создают блоков.



Операторы **GPSS** должны записываться одной текстовой строкой длиной не более 250 символов.

Операторы состоят из *полей*.

*Поле* – это набор символов, отделенный пробелами или ограничителем.

В общем случае оператор состоит из следующих полей:

- номер строки (необязательно);
- метка (необязательно);
- имя блока или команды (обязательно);
- операнды (в зависимости от блока или команды);
- комментарий (необязательно).



Номера строк оставлены для совместимости с **GPSS/PC** (система моделирования под управлением **MS DOS**).

Игнорируются **GPSS World**.

Для различения объектов **GPSS** им могут присваиваться имена.

Имена должны начинаться с буквы, могут содержать до 200 букв и цифр и символы подчеркивания.



Для реализации переходов от одного объекта модели к другому объекты могут идентифицироваться с помощью меток (в поле метки).

Метки формируются так же, как имена объектов.



Если рабочая строка заканчивается символом «;», то далее в этой строке можно поместить комментарий. Комментарий также может помещаться в строке, в первой позиции которой находится символ «\*».

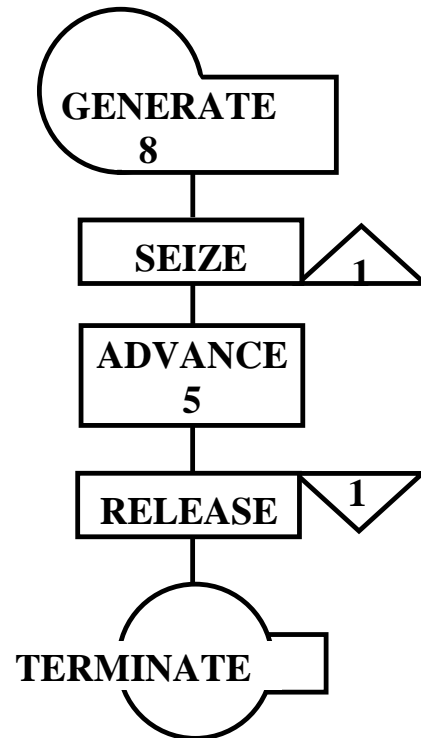


## **Пример** (создание простейшей программы).

Заявки поступают в одноканальную СМО через фиксированное время 8 мин. Обработка каждой заявки занимает фиксированное время 5 мин. После обработки заявки покидают систему. Выполнить обработку 100 заявок.



Блок-диаграмма имитационной модели имеет вид:

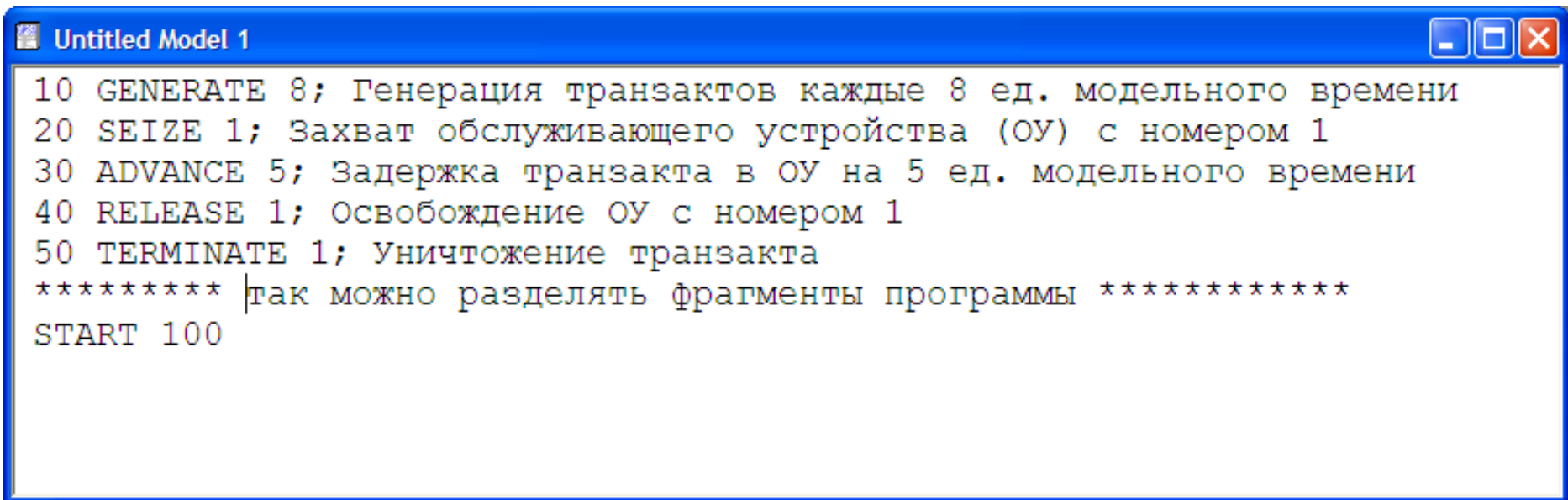


Для создания новой программы – команда меню  
**File | New | Model**





В окне редактора вводится текст программы:

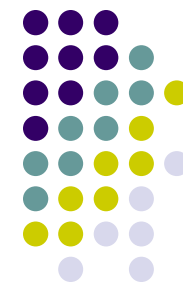


```
10 GENERATE 8; Генерация транзактов каждые 8 ед. модельного времени
20 SEIZE 1; Захват обслуживающего устройства (ОУ) с номером 1
30 ADVANCE 5; Задержка транзакта в ОУ на 5 ед. модельного времени
40 RELEASE 1; Освобождение ОУ с номером 1
50 TERMINATE 1; Уничтожение транзакта
***** так можно разделять фрагменты программы *****
START 100
```

Запуск имитации – команда меню

**Command | Create Simulation**

По окончании имитации **GPSS World** создает стандартный отчет:



Untitled Model 1.3.1 - REPORT

GPSS World Simulation Report - Untitled Model 1

Monday, March 24, 2014 11:21:28

START TIME	END TIME	BLOCKS	FACILITIES	STORAGES
0.000	805.000	5	1	0

BLOCK	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY
1	GENERATE	100	0	0
2	SEIZE	100	0	0
3	ADVANCE	100	0	0
4	RELEASE	100	0	0
5	TERMINATE	100	0	0

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
1	100	0.621	5.000	1	0	0	0	0	0

FEC	XN	PRI	ST	ASSE	CURRENT	NEXT	PARAMETER	VALUE
101	0		08.000	103	0	1		

Количество используемых в программе блоков, установленных ОКУ и МКУ

Количество транзактов, задержанных в блоке на момент окончания моделирования

Количество транзактов, ожидающих условий для прохождения через блок

Количество транзактов, проходящих через блок

Характеристика работы ОКУ с номером 1

Коэффициент использования

Среднее время обработки одного транзакта

Готовность устройства на момент окончания моделирования



## Объекты вычислительной категории

К ним относятся арифметические и булевы переменные и функции.

*Переменная* – это обобщенное понятие, которое определяется *выражением*.

Выражение может включать элементы:

- константы;
- СЧА;
- арифметические и логические операции;
- библиотечные функции;
- переменные пользователя.



- Константы представляются одним из трех типов данных:
  - целочисленным  
32-разрядные числа; при переполнении целое число преобразуется в вещественное;
  - вещественным  
64-разрядные числа с плавающей точкой двойной точности;  
мантисса может изменяться от  $-308$  до  $+308$ ;  
точность ограничена 15 разрядами;
  - строковым  
массив символов ASCII любой длины (длина ограничена значением ***Max Memory Request*** – настройка в меню **Edit | Settings**, вкладка **Simulation**).

- Арифметические, условные и логические операторы.



Оператор	Операция	Выражение	Значение
$\wedge$	Возведение в степень	$A \wedge B$	A в степени B
# (или *)	Умножение	$A \# B$	Произведение A на B
/	Деление	$A / B$	Частное от деления A на B
\	Целочисленное деление	$A \setminus B$	Частное от целочисленного деления A на B
@	Целый остаток	$A @ B$	Целый остаток от деления A на B
-	Вычитание	$A - B$	Разность A и B
+	Сложение	$A + B$	Сумма A и B
$\geq$	Больше или равно	$A \geq B$	1, если $A \geq B$ , 0 в противном случае
$\leq$	Меньше или равно	$A \leq B$	1, если $A \leq B$ , 0 в противном случае
$>$	Больше	$A > B$	1, если $A > B$ , 0 в противном случае
$<$	Меньше	$A < B$	1, если $A < B$ , 0 в противном случае
=	Равно	$A = B$	1, если $A = B$ , 0 в противном случае
$\neq$	Не равно	$A \neq B$	1, если $A \neq B$ , 0 в противном случае
&	Логическое И	$A \& B$	1, если $A \neq 0$ и $B \neq 0$ , 0 в противном случае
	Логическое ИЛИ	$A   B$	1, если $A \neq 0$ или $B \neq 0$ , 0 в противном случае



Если оператор требует определенного типа данных, то данные преобразуются автоматически.

Например: если над строкой производится числовая операция, то будет использован числовой эквивалент этой строки.

Оператор «#» используется для умножения потому что «\*» используется для указания косвенной адресации.

Изменение значений операторов «#» и «\*» — меню **Edit | Settings**, вкладка **Simulation**.



Приоритет операций (в порядке убывания):

1)  $\wedge$

2)  $\# (*) / \backslash$

3)  $@$

4)  $- +$

5)  $>= <= > <$

6)  $= !=$

7)  $\&$

8)  $|$



- Библиотечные математические функции (библиотечные процедуры).

Функция	Значение
<b>ABS(A)</b>	Абсолютное значение A
<b>ATN(A)</b>	Арктангенс A
<b>COS(A)</b>	Косинус A
<b>EXP(A)</b>	Экспонента ( $e^A$ )
<b>INT(A)</b>	Целая часть A
<b>LOG(A)</b>	Натуральный логарифм A
<b>SIN(A)</b>	Синус A
<b>SQR(A)</b>	Квадратный корень из A
<b>TAN(A)</b>	Тангенс A





Аргумент библиотечной функции может представлять собой выражение.

Аргументы во всех случаях преобразуются в числовые значения.

Углы задаются в радианах.

Значения математических функций имеют вещественный тип.



- Библиотечные генераторы случайных чисел.

В ***GPSS World*** имеется более 20 библиотечных генераторов случайных чисел, реализующих различные законы распределения.

Все аргументы генераторов могут представлять собой выражения.

Далее рассматриваются наиболее широко используемые генераторы.



## ***Биномиальное распределение.***

Синтаксис:

***BINOMIAL(Stream, TrialCount, Probability)***

Аргументы:

*Stream* — номер генератора случайных чисел  
(число, большее или равное 1);

*TrialCount* — количество независимых испытаний  
(положительное число);

*Probability* — вероятность успеха отдельном испытании  
(число, принадлежащее интервалу (0, 1) ).

Возвращаемое значение:

целое число — одно значение СВ, имеющей  
биномиальный закон распределения.



## ***Распределение Пуассона.***

Синтаксис:

***POISSON(Stream, Mean)***

Аргументы:

*Stream* – номер генератора случайных чисел  
(число, большее или равное 1);

*Mean* – среднее число событий на  
рассматриваемом промежутке, среднее  
значение СВ (положительное число);

Возвращаемое значение:

целое число – одно значение СВ, имеющей  
распределение Пуассона.

# ***Показательное распределение.***



## Синтаксис:

***EXPONENTIAL(Stream, Locate, Scale)***

## Аргументы:

*Stream* – номер генератора случайных чисел  
(число, большее или равное 1);

*Locate* – величина сдвига;

*Scale* – параметр, характеризующий растяжение /  
сжатие (положительное число);

при  $Locate = 0$  *Scale* – среднее значение;  
в общем случае  $Locate + Scale$  – среднее значение.

## Возвращаемое значение:

вещественное число – одно значение СВ, имеющей  
показательный закон распределения.



## ***Нормальное распределение.***

Синтаксис:

***NORMAL(Stream, Mean, StdDev)***

Аргументы:

*Stream* – номер генератора случайных чисел  
(число, большее или равное 1);

*Mean* – среднее значение;

*StdDev* – среднее квадратическое отклонение  
(положительное число).

Возвращаемое значение:

вещественное число – одно значение СВ, имеющей нормальный закон распределения.



## ***Равномерное распределение.***

Синтаксис:

***UNIFORM(Stream, Min, Max)***

Аргументы:

*Stream* – номер генератора случайных чисел  
(число, большее или равное 1);

*Min* – наименьшее возможное значение СВ;

*Max* – наибольшее возможное значение СВ;  
 $Min < Max$ .

Возвращаемое значение:

вещественное число – одно значение СВ, имеющей  
равномерное на  $[Min, Max]$  распределение.

# Организация поступления транзактов в модель и удаления их из модели



- *Поступление транзактов в модель.*

Введение транзакта в модель – блок **GENERATE**.

Формат блока:

**GENERATE** [A], [B], [C], [D], [E]

[ ] – необязательный  
операнд

Операнды A и B могут быть

- неотрицательным числом,
- именем,
- выражением в скобках,
- СЧА.

Значение B не должно превышать значения A.

Если операнды не заданы, по умолчанию подразумеваются нулевые значения.





A – средний интервал времени между последовательными поступлениями транзактов в модель.

B – модификатор, изменяющий значение интервала генерации транзактов по сравнению с указанным операндом A.

Имеется два типа модификаторов:

- модификатор-интервал;
- модификатор- функция.



С помощью *модификатора-интервала* задается равномерный закон распределения времени между поступлениями транзактов:

$A - B$  задает нижнюю границу интервала;

$A + B$  задает верхнюю границу интервала;

случайное число из полученного интервала – время, через которое следующий транзакт выйдет из блока ***GENERATE***.

В частности,  $B = 0$  определяет детерминированное значение интервала времени между приходами транзактов, равное  $A$ .



Интервалы времени, имеющие закон распределения, отличный от равномерного, могут быть заданы с помощью *модификатора-функции*.

При этом значение  $A$  умножается на значение функции, заданной операндом  $B$ .



Операнды C, D и E задают следующие параметры.

C – смещение (момент времени, в который в блоке ***GENERATE*** должен появиться первый транзакт).

Может задаваться теми же способами, что операнды A и B.

D – граничное значение числа транзактов, которые могут войти в модель через данный блок ***GENERATE*** в течение времени моделирования.

Если операнд D не задан, по умолчанию это ограничение отсутствует.



Е – класс приоритета каждого из транзактов, входящих в модель через данный блок ***GENERATE***.

Рекомендуется использовать последовательность целых чисел 0, 1, 2, 3, ...

Чем больше число, тем выше приоритет.

Если операнд Е не задан, по умолчанию задается приоритет, равный 0.

Операнды D и Е могут задаваться так же, как А, В и С, но принимать значения только целых положительных и целых чисел соответственно.



В любом блоке ***GENERATE*** должен быть задан либо операнд A, либо операнд D.

Нельзя использовать в качестве операндов параметры транзактов.

Транзакт не должен входить в блок ***GENERATE***.  
Такая попытка вызовет ошибку выполнения.



## Примеры.

**GENERATE** (exponential(1, 0, 0.2))

интервал времени между появлениями транзактов имеет показательное распределение со средним значением, равным 0,2 ( $\lambda = 5$ );  
параметры B, C, D и E не заданы.

**GENERATE** 7, 2

интервал времени между появлениями транзактов имеет равномерное на [5, 9] распределение;  
параметры C, D и E не заданы.



***GENERATE*** 7,2,,100

генерация 100 транзактов; интервал времени между их появлениями имеет равномерное на [5, 9] распределение;  
параметры C и E не заданы.

***GENERATE*** 13.3, 2.8,,,1

интервал времени между появлениями транзактов имеет равномерное на [10.5, 16.1] распределение;  
приоритет транзактов равен 1;  
параметры C и D не заданы.





Все СЧА, используемые в блоке **GENERATE**, должны быть уже определены предшествующими командами.

Пример.

SrIntPost EQU 47.2

StandOtkl EQU 28.6

INITIAL X\$KolTrans, 43

GENERATE SrIntPost,StandOtkl,,X\$KolTrans

Присвоение именам  
числовых значений

Присвоение начального  
значения сохраняемой  
ячейке с именем KolTrans



- **Удаление транзактов из модели и завершение моделирования.**

Удаление транзакта из модели – блок ***TERMINATE***.

Формат блока:

***TERMINATE*** [A]

A – число единиц, на которое блок ***TERMINATE*** уменьшает содержимое *счетчика завершения*, определяющего момент окончания моделирования.

Операнд A может быть

- неотрицательным целым числом,
- именем,
- выражением в скобках,
- СЧА.



По умолчанию значение  $A$  равно 0 (транзакт уничтожается, а содержимое счетчика не меняется).

*Счетчик завершения* – ячейка памяти, которая хранит неотрицательное целое число, записанное в начале моделирования командой ***START***.

При попадании транзактов в блок ***TERMINATE*** значение счетчика уменьшается на  $A$ .

По достижении нуля моделирование прекращается.

В модели может быть множество блоков ***TERMINATE***, но счетчик завершения – только один.



Для запуска процесса моделирования используется команда ***START***.

Формат команды:

***START*** A, [B], C, [D]

A – начальное значение содержимого счетчика завершения.

Может быть целым положительным числом.

B – операнд вывода статистики.

Может быть NP («нет вывода данных» – стандартный отчет не выводится) либо опущен.

По умолчанию выводится стандартный отчет.



Операнд С не используется (сохранен для совместимости с ранними версиями **GPSS**).

Операнд D определяет необходимость вывода содержимого списков событий.

Если задать в качестве D любое положительное целое число, то списки текущих и будущих событий включаются в стандартный отчет.

Если операнд D не задан, по умолчанию эти списки не выводятся.



Команду ***START*** можно

- указывать в конце программы  
тогда после трансляции (т. е. создания объекта ***Simulation***) начинается моделирование;
- вводить в интерактивном режиме.



Блоки ***GENERATE*** и ***TERMINATE*** и команда ***START*** могут использоваться для управления временем моделирования.

### Пример.

Пусть единица времени составляет 1 мин.

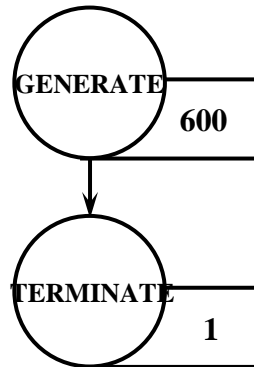
Требуется промоделировать работу системы в течение 10 часов (после этого процесс моделирования должен быть закончен).

Если за единицу модельного времени взять 1 мин., то время моделирования составит 600 единиц.



Для управления временем моделирования следует

- 1) включить в модель сегмент из блоков **GENERATE** и **TERMINATE**;



- 2) во всех прочих блоках **TERMINATE** модели использовать операнд A по умолчанию;
- 3) в команде **START** задать A=1.





Блоки ***GENERATE*** и ***TERMINATE*** и команда ***START*** могут использоваться для управления количеством обработанных транзактов.

### Пример.

Пусть требуется завершить моделирование после обработки 100 деталей.

В блоках ***TERMINATE***, которые выводят из модели транзакты, имеющие смысл изготовленных деталей, в качестве операнда A указать 1.

У остальных блоков ***TERMINATE*** операнд A должен быть опущен.

В команде ***START*** в качестве операнда A задать требуемое количество обработанных деталей.



.....  
TERMINATE 1

Выводит 1 транзакт, имеющий  
смысл изготовленной детали

.....  
TERMINATE 1

Выводит 1 транзакт, имеющий  
смысл изготовленной детали

.....  
TERMINATE

Выводит из модели транзакт,  
имеющий другой смысл

.....  
START 100



- *Изменение значений параметров транзактов.*

Каждый транзакт может иметь любое число параметров.

Их интерпретация – произвольная.

В момент генерации транзакта все его параметры нулевые.

Основное средство задания значений параметров транзактов – блок **ASSIGN**.

Формат блока:

**ASSIGN** A, B, [C]



А – номер параметра, которому присваивается значение.

Операнд А может быть

- положительным целым числом,
- именем,
- выражением в скобках,
- СЧА

и следующими за ними знаками

+ (если нужно увеличить) или  
– (если нужно уменьшить).



В – значение, которое следует добавить, вычесть или которым заменить значение в параметре, заданном операндом А.

Если такой параметр не существует, он будет создан со значением, равным 0.

Операнд В может задаваться теми же способами, что операнд А; кроме того, быть числом или строкой.

С – номер или имя модификатора-функции.

Значение В умножается на значение модификатора, и полученное значение произведения заменяет значение параметра, заданного операндом А.

## Примеры.



### **ASSIGN** 1,517.3

параметру 1 активного транзакта  
присваивается значение 517,3;  
параметр C не задан.

### **ASSIGN** 4+,Q5

к значению параметра 4 активного транзакта  
прибавляется значение текущей длины очереди с  
номером 5;  
параметр C не задан.

### **ASSIGN** Tr-, (Normal(32, Sr, StOtkl)), Expdis

вычисляются значения выражения в скобках и  
функции с именем Expdis, перемножаются, и  
полученное произведение вычитается из значения  
параметра с именем Tr активного транзакта.



Блок ***MARK*** позволяет сопоставить активному транзакту или его параметру значение абсолютного модельного времени (СЧА АС1).

Формат блока:

***MARK*** [A]

A – номер параметра, в который записывается значение абсолютного модельного времени.



## Примеры.

### ***MARK***

по умолчанию вошедшему в этот блок транзакту устанавливается время входа в систему, равное абсолютному модельному времени.

### ***MARK Vход***

значение абсолютного модельного времени заносится в параметр с именем *Vход* вошедшего в блок транзакта.

Если этого параметра нет, то он создается.





Параметр **MARK** можно использовать, например, если необходимо определить время обработки транзакта, которое складывается из времен обработки на нескольких этапах.

С помощью **MARK** нужно записать абсолютное модельное время начала и конца обработки, и найти разность этих значений.



## Моделирование функционирования ОКУ

ОКУ – объект аппаратной категории.

Характеризуется свойствами:

- ❑ в любой момент времени в ОКУ может обслуживаться только один транзакт;  
если в процессе обслуживания появляется новый транзакт, то он должен
  - либо ждать своей очереди на обслуживание,
  - либо перейти на другое устройство,
  - либо (в случае более высокого приоритета) прервать текущее обслуживание;
- ❑ когда на ОКУ поступает транзакт, в модели следует учесть время, необходимое для его обслуживания.



В модели может быть множество ОКУ



им даются символические имена  
(требования к именам описаны выше).

Возможны следующие режимы функционирования ОКУ:

- занятие ОКУ и его освобождение;
- прерывание обслуживания ОКУ;
- недоступность ОКУ и восстановление доступности.

Далее рассматривается моделирование первого из этих режимов.



- ***Занятие ОКУ и его освобождение.***

Попытка занятия транзактом ОКУ моделируется попыткой входа в блок, описывающий это ОКУ.

Такой блок должен обладать свойствами:

- если ОКУ уже используется, то транзакт не может войти в блок и должен ждать очереди;
- если ОКУ не используется, то транзакт может войти в блок; при этом статус ОКУ изменяется на «занято».

Указанными свойствами обладает блок ***SEIZE***.

Этот блок моделирует занятие ОКУ.



После окончания обслуживания должен быть использован блок, моделирующий освобождение устройства.

Его основное назначение – изменение состояния ОКУ с «занято» на «свободно».

Таким блоком является блок ***RELEASE***.



Форматы блоков:

***SEIZE***      A

***RELEASE*** A

В обоих случаях операнд A – имя или номер занимаемого (освобождаемого) ОКУ.

Может быть

- положительным целым числом,
- именем,
- выражением в скобках,
- СЧА.

Прежде чем освободить ОКУ, транзакт может пройти через неограниченное число блоков.



- *Имитация обслуживания.*

В течение времени, необходимого для обслуживания транзакта в ОКУ, этот транзакт должен приостановить свое движение по модели (только по истечении этого времени он может попасть в блок **RELEASE**).

Для задержки транзакта в течение некоторого промежутка модельного времени используется блок **ADVANCE**.



Формат блока:

***ADVANCE*** A, [B]

A – среднее время обслуживания.

B – модификатор, изменяющий значение операнда A.

Как и в блоке ***GENERATE***, модификаторы могут быть двух видов; их применение аналогично.

Операнды A и B могут быть

- числами,
- именами,
- выражениями в скобках,
- СЧА.





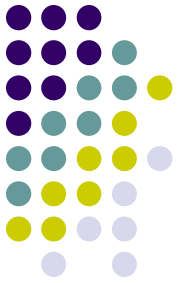
## Примеры.

**ADVANCE** (Normal(32,X\$Sr,X\$StOtkl))

время задержки распределено по нормальному закону со средним значением и средним квадратическим отклонением, предварительно записанными командой **INITIAL** в сохраняемые ячейки с именами Sr и StOtkl соответственно.

**ADVANCE** 56.7,23.2

время задержки имеет равномерное на [33.5, 79.9] распределение.

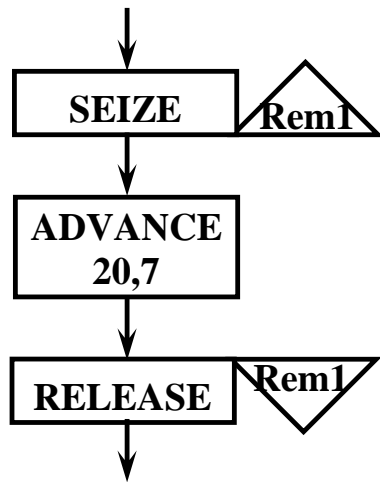


Блок ***ADVANCE*** никогда не препятствует входу транзакта.  
Любое число транзактов может находиться в этом блоке одновременно.



## Пример

использования блоков ***SEIZE***, ***RELEASE*** и ***ADVANCE***:



***SEIZE***      **Rem1**  
***ADVANCE***    **20,7**  
***RELEASE***    **Rem1**

Транзакт занимает ОКУ с именем **Rem1** и задерживается в нем на время, равномерно распределенное в промежутке **[13, 27]**.

Затем транзакт покидает ОКУ, и следующий транзакт попытается занять это устройство.



- ***Проверка состояния устройства.***

Пусть в СМО имеется 2 ОКУ, одно из которых занято.

Если при этом второе ОКУ свободно, то надо перенаправить активный транзакт на свободное ОКУ.

Вопрос: как узнать состояние ОКУ?



- ❑ Определение состояния ОКУ без изменения состояния выполняет блок ***GATE***.
- ❑ Блок ***TEST*** описывает условие, проверяемое при входе в этот блок транзакта, и определяет для этого транзакта номер следующего блока в зависимости от выполнения или невыполнения условия.

Форматы и способы применения блоков рассмотреть самостоятельно.



## Изменение маршрутов движения транзактов в модели

- Блок ***TRANSFER*** предназначен для передачи входящего в него транзакта в другой блок модели.

Формат блока:

***TRANSFER*** [A], [B], [C], [D]

Операнд А задает режим выбора (способ выбора блока, к которому должен быть направлен очередной транзакт).

Всего имеется 9 режимов работы блока.

Далее рассматриваются 5 из этих режимов.



Операнды В и С задают возможные значения номеров следующих блоков или их положение.

Если операнд В опущен, то используется номер блока, следующего за блоком ***TRANSFER***.



- *Режим безусловной передачи.*

В этом режиме операнд А не используется, а операнд В указывает метку блока, в который транзакт должен попытаться войти.

Пример.

***TRANSFER*** ,Oper

После входа в блок ***TRANSFER*** транзакт сразу же попытается войти в блок с меткой Oper.

Если этот блок отказывает во входе, то транзакт остается в блоке ***TRANSFER***.





- *Режим статистической передачи.*

Если операнд А используется и не является зарезервированным словом, блок **TRANSFER** работает в режиме передачи транзакта в один из двух блоков случайным образом.

Значение операнда А, записываемое после точки, рассматривается как трехзначное число, показывающее, какая часть (в тысячных долях) транзактов, входящих в блок, должна быть направлена в блок, определяемый операндом С.

Остальные транзакты отправляются в блок, определяемый операндом В, или в следующий по порядку блок, если операнд В опущен.



### Пример.

При моделировании работы цеха по производству деталей известно, что 7,5% изготовленных деталей бракуется.

***TRANSFER*** .075,Oper1,Oper2

Транзакты, имитирующие изготовленные детали в 7,5% случаев будут направлены в блок с меткой Oper2, и в 92,5% случаев – в блок с меткой Oper1.



- *Режим BOTH.*

Если в качестве операнда А используется ключевое слово *BOTH*, то в этом режиме каждый транзакт, вошедший в блок ***TRANSFER***, проверяет два пути.

Сначала проверяется возможность войти в блок, определяемый операндом В.

Если транзакт не может войти в этот блок, он пытается войти в блок, определяемый операндом С. Если транзакт не может войти и в этот блок, он задерживается в блоке ***TRANSFER***.

## Пример.



```
***** генерация заявок *****
      GENERATE (exponential(3,0,11.7))
      TRANSFER BOTH,,met
***** имитация ОКУ1 *****
      SEIZE Rem1
      ADVANCE 13.8,4.7
      RELEASE Rem1
      TERMINATE
***** имитация ОКУ2 *****
met SEIZE Rem2
      ADVANCE (exponential(4,0,15.7))
      RELEASE Rem2
      TERMINATE
*** определение времени моделирования ***
      GENERATE 3600
      TERMINATE 1
```



- *Режим ALL.*

Если в качестве операнда A используется ключевое слово *ALL*, то в этом режиме каждый транзакт, вошедший в блок ***TRANSFER***, проверяет возможность войти в любой блок, начиная с блока, определяемого операндом B, и заканчивая блоком, определяемым операндом C.

Операнд D определяет шаг изменения номера проверяемого блока.

В качестве операндов B и C можно указывать номера или метки блоков.



Пусть  $v$  – номер блока, определяемого операндом  $B$ ;  
 $z$  – номер блока, определяемого операндом  $C$ ;  
 $d$  – шаг, заданный операндом  $D$ .

Транзакт, вошедший в блок ***TRANSFER***, пытается войти в блок с номером  $v$ .

Если этот блок занят, транзакт пытается войти в блоки с номерами  $v+d$ ,  $v+2d$ ,  $v+3d$ , ...  $z$ .

Если операнд  $C$  опущен, то проверяется только один блок.

Пример.

Заявки распределяются по трем ОКУ. Очередная заявка обслуживается первым освободившимся ОКУ.

\*\*\*\*\* генерация заявок \*\*\*\*\*

GENERATE 5,2

TRANSFER ALL,CHAN1,CHAN3,4

\*\*\*\*\* имитация ОКУ1 \*\*\*\*\*

CHAN1 SEIZE 1

ADVANCE 13,4

RELEASE 1

TRANSFER ,Exit1

\*\*\*\*\* имитация ОКУ2 \*\*\*\*\*

CHAN2 SEIZE 2

ADVANCE 11,3

RELEASE 2

TRANSFER ,Exit1

\*\*\*\*\* имитация ОКУ3 \*\*\*\*\*

CHAN3 SEIZE 3

ADVANCE 9,2

RELEASE 3

Exit1 TERMINATE

\*\*\* определение времени моделирования \*\*\*

GENERATE 3600

TERMINATE 1





## Замечание.

В режимах *BOTH* и *ALL* в тех случаях, когда возможен переход более чем к одному блоку, преимущество имеют:

блок В в режиме *BOTH*;

блок с меньшим номером в режиме *ALL* .

## Пример.

Фрагмент отчета из предыдущего примера:

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PE
1	239	0.867	13.059	1	0	
2	251	0.767	10.994	1	720	
3	229	0.569	8.944	1	719	

Устройства 1 и 2 обслужили большее число заявок, чем устройство 3, хотя среднее время обслуживания в них больше, чем в устройстве 3





- Блок ***LOOP*** предназначен для организации циклов.
- Блок ***DISPLACE*** предназначен для нахождения любого транзакта и перемещения его к новому блоку.



## Моделирование функционирования МКУ

МКУ может быть использовано несколькими транзактами одновременно.

Ограничений на число МКУ в модели нет.

Для различения МКУ им дают имена.

Для определения МКУ – команда ***STORAGE***.

Формат команды:

Name ***STORAGE*** A



Name – имя МКУ;

Символическому имени может быть поставлен в соответствие номер командой ***EQU*** (может быть необходимо при обращении к нескольким МКУ в блоках ***SELECT*** и ***COUNT***).

Операнд А определяет емкость МКУ.

Должен быть положительным целым числом.



В модели МКУ может функционировать в двух режимах:

- занятие и освобождение МКУ;
- недоступность МКУ.



- **Занятие МКУ и его освобождение.**

Занятие и освобождение МКУ имитируется блоками ***ENTER*** и ***LEAVE***.

Форматы блоков:

***ENTER*** A, [B]

***LEAVE*** A, [B]

Операнд A – для указания имени МКУ;

операнд B задает число устройств (элементов памяти), которое должно быть занято в блоке ***ENTER*** или освобождено в блоке ***LEAVE***.

По умолчанию B = 1.



Транзакт может войти в блок ***ENTER***, если

- существует МКУ с указанным именем,
- МКУ находится в доступном состоянии,
- достаточно емкости для выполнения запроса.

В противном случае транзакт помещается в список задержки устройства в соответствии с приоритетом.



При входе транзакта в блок **ENTER** планировщик

- увеличивает на значение операнда В текущее содержимое МКУ;
- уменьшает на значение операнда В доступную емкость МКУ.

МКУ можно переопределить (изменить емкость) другой командой **STORAGE** с тем же именем.



### Пример.

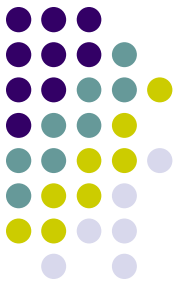
Поток заявок поступает в накопитель с допустимой емкостью, равной 3 единицам; время поступления подчинено равномерному на  $(4, 10)$  закону распределения.

Если заявки застают первый обслуживающий канал занятым, то они поступают на обработку во второй канал. Время обработки в первом и втором канале имеет равномерное распределение на  $(10, 16)$  и  $(7, 11)$  соответственно.





```
***** определение накопителя *****
BUF    STORAGE 3
** генерация заявок и вход в накопитель **
    GENERATE 7,3
    ENTER BUF
    TRANSFER BOTH,,CHAN2
***** имитация ОКУ1 *****
    SEIZE 1
    LEAVE BUF
    ADVANCE 13,3
    RELEASE 1
    TRANSFER ,Exit1
***** имитация ОКУ2 *****
CHAN2 SEIZE 2
    LEAVE BUF
    ADVANCE 9,2
    RELEASE 2
Exit1 TERMINATE
*** определение времени моделирования ***
    GENERATE 3600
    TERMINATE 1
```



## Организация независимых прогонов модели

При проведении эксперимента необходимо организовать несколько последовательных прогонов модели, возможно, изменяя при этом некоторые параметры модели.

Это может быть достигнуто за счет использования команды ***CLEAR***.

Формат команды:

***CLEAR*** [A]



Операнд *A* может иметь значение *ON* или *OFF*.

По умолчанию – значение *ON*.

Команда ***CLEAR*** возвращает процесс моделирования в исходное состояние:

сбрасывает всю накопленную статистическую информацию, удаляет все транзакты из процесса моделирования и заполняет все блоки

***GENERATE*** первым транзактом; ОКУ и МКУ становятся доступными, устанавливаются в незанятое состояние.

Генераторы случайных чисел **не** сбрасываются.



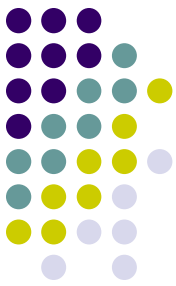
Если операнд  $A$  имеет значение  $OFF$ , то сохраняемые ячейки, логические ключи и элементы матриц остаются без изменений.

### Пример.

Следующая программа позволяет организовать 2 независимых прогона модели из предыдущего примера.



```
BUF      STORAGE 3
        GENERATE 7,3
        ENTER BUF
        TRANSFER BOTH,,CHAN2
        SEIZE 1
        LEAVE BUF
        ADVANCE 13,3
        RELEASE 1
        TRANSFER ,Exit1
CHAN2    SEIZE 2
        LEAVE BUF
        ADVANCE 9,2
        RELEASE 2
Exit1    TERMINATE
        GENERATE 3600
        TERMINATE 1
        START 1 ; запуск первого прогона
        CLEAR   ; сброс статистики
        START 1 ; запуск второго прогона
```



В результате создается два отчета  
(для первого и второго прогона).

Фрагменты этих отчетов:

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
1	236	0.848	12.932	1	511	0	0	0	0
2	274	0.679	8.916	1	0	0	0	0	0

STORAGE	CAP.	REM.	MIN.	MAX.	ENTRIES	AVL.	AVE.C.	UTIL.	RETRY	DELAY
BUF	3	3	0	2	510	1	0.042	0.014	0	0

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
1	240	0.868	13.015	1	514	0	0	1	0
2	274	0.681	8.951	1	515	0	0	1	0

STORAGE	CAP.	REM.	MIN.	MAX.	ENTRIES	AVL.	AVE.C.	UTIL.	RETRY	DELAY
BUF	3	2	0	1	515	1	0.055	0.018	0	0