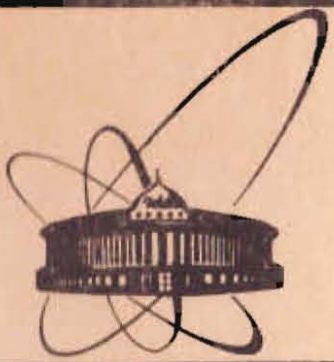

сообщения
объединенного
института
ядерных
исследований
дубна

11-85-846

А.М.Хасанов

LISS – СИСТЕМА СОПРОВОЖДЕНИЯ БИБЛИОТЕК ПРОГРАММ

Команды системы

1985

I. Общие сведения по работе с системой

Работа с системой происходит в интерактивном режиме с помощью нескольких команд. Основной единицей работы системы является порция.

Порция - это часть банка данных, удовлетворяющая заданным условиям. Например, сначала мы работаем с порцией, равной всему банку данных. Обозначим ее через $A0$, затем из нее получаем порцию $A1$ и т.д.: $A0 \supseteq A1 \supseteq A2 \supseteq \ldots$. В каждый момент времени система работает с определенной порцией, которая называется текущей. Информацию о ней можно вывести на экран и/или печать, ее можно отсортировать и т.п. Отсортированную порцию обозначим SI. В любой момент времени мы можем вернуться к началу работы, т.е. к порции $A0$.

Работу с системой можно представить в виде следующей блок-схемы:

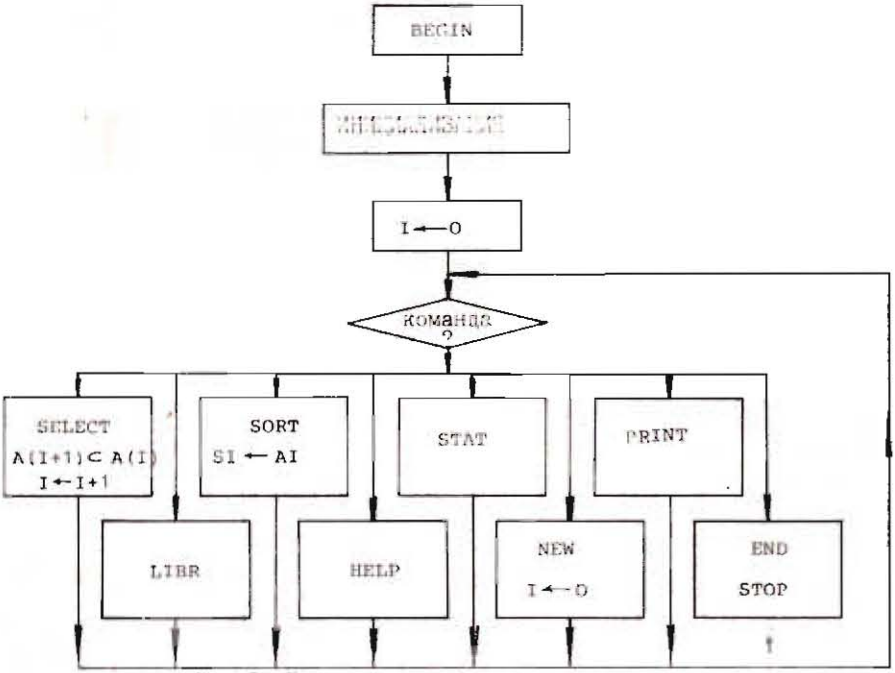

Рис.I. Блок-схема работы системы.

В настоящей версии системы существует восемь команд: SELECT , SORT , STAT , PRINT , LIBR , HELP , NEW и END . С каждой командой можно работать в двух режимах: в режиме набора всей команды в один прием и в режиме набора команды по частям. Ввод любой информации в машину завершается нажатием клавиши "ВВОД". Будем обозначать это символом " □ ".

Например, если набрано только название команды: SELECT□ , то система переходит во второй режим. Если же после названия команды есть еще символы, то система готовится принять всю команду сразу: SELECT,P(SET(SUPPRT & FORTRN))□

В существующей версии системы первый режим не реализован. Работа во втором режиме происходит в виде диалога с системой, где инициатива принадлежит системе. Она задает вопросы, пользователь отвечает.

На любой вопрос системы можно ответить "нет" нажатием клавиши "ввод" без набора символов. Обычно "нет" означает либо подтверждение режима "по умолчанию", либо отказ от ввода каких-либо данных, либо отказ от каких-то действий. Вопросы построены таким образом, что отвечать "нет" приходится чаще, что уменьшает объем вводимой пользователем информации и ускоряет работу. Если на все вопросы отвечать "нет", то,во-первых, вопросов будет мало, во-вторых, это будет означать пустую команду.

Вопросы системы могут быть трех видов. Вопросы,на которые надо отвечать "нет" или "да" (вводом символа 'Y'). Например:

"Будете вводить условия на тип USERN ?"

Вопросы, на которые надо отвечать вводом одной или нескольких цифр из числа предлагаемых системой. Например:

"Вы имеете порцию А3. Ваши дальнейшие действия ?
1 - новая порция;
2 - сортировка;
3 - статистика;
4 - печать;
5 - библиотека;
6 - помощь системы;
7 - начать работу заново;
8 - конец работы.
Введите соответствующую цифру ".

Вопросы, в ответ на которые надо ввести требуемую информацию. Например:

"Введите условия на атрибуты типа INT ".

Существуют различия в диалоге с системой опытного пользователя, который знает ее, и новичка. В начале своей работы система делает запрос:

"Система может работать в трех режимах:
1 - обучение и помощь в работе;
2 - только помощь;
3 - самостоятельная работа (по умолчанию).
Введите соответствующую цифру".

Пользователь может выбрать себе подходящий режим. В дальнейшем он может быть изменен с помощью команды HELP .

В режиме 1 система выдает на экран терминала свое описание и затем действует в режиме 2. Он отличается от режима 3 полнотой вопросов. Вопросы в режиме 3 звучат просто как напоминание о чем говорится в данный момент:

"Введите имена атрибутов" .

Вопросы в режиме 2 дают исчерпывающую информацию:

"Стандартная выдача для записей типа USERN состоит из значений следующих атрибутов:
Тип NAME:NAME
Тип INT:
Тип SET:COMPUT,LANG ,STATUS
Наберите 'STAND' , если вы хотите стандартную выдачу, наберите 'ALL' если вы хотите выдать все атрибуты, иначе введите имена требуемых атрибутов через запятую, в любом порядке и на любом количестве строк. Имена можно сокращать. 'END' означает конец ввода".

Если вы некорректно отвечаете на вопрос, система указывает, в чем состоит ваша ошибка, и просит повторить ответ. Если ошибка допускается пять раз подряд, система выдает сообщение:

"Система поможет вам, если вы наберете "HELP" .

Если вы никак не можете выйти из тупика, попробуйте набрать 'END' .

Опишем подробнее команды системы.

2. HELP

Эта команда предназначена для помощи пользователю при работе с системой. Например, она может выдать на экран терминала описание системы LISS . Заметим, что,если система применяется для другого приложения, то описание должно быть соответствующим образом изменено. Из него пользователь может узнать, какие типы записей существуют в банке данных, структуру этих записей, получить справку о командах системы и т.д. Словом,почти всю информацию, данную в этой статье, он может найти, используя команду HELP .

Это же описание системы может быть выдано на печать в форматированном виде для использования как инструкции. Как уже говорилось, с помощью команды HELP можно изменять режим работы с системой.

3. SELECT

Эта команда по заданным условиям строит из текущей порции следующую:
SELECT(AI) —→ A(I+1)

Как известно, банк данных состоит из наборов. В очередную порцию входят те наборы из текущей порции, которые удовлетворяют поставленным условиям. Так как мы можем применять команду SELECT последовательно, итеративно уточняя наши условия, то в порции могут оказаться наборы, выбранные по довольно сложным запросам.

Что значит – набор удовлетворяет определенным условиям ? Будем последовательно уточнять, как задаются условия. В соответствии со структурой набора условия на набор состоят из условий на записи, которые, в свою очередь, состоят из условий на атрибуты и т.д.

Набор содержит одну запись главного типа и некоторое количество записей каждого из подчиненных типов. На записи каждого типа накладываются отдельные условия. Можно задать условия на все существующие типы записей, а можно только на некоторые.

Когда все условия заданы, система начинает проверять каждый набор. Сначала проверяется, удовлетворяет ли запись главного типа соответствующим условиям. Если нет, то дальнейшей проверки не производится и набор считается "невыбранным". Если же да или условий на главный тип не наложено, то происходит проверка записей тех подчиненных типов, на которые есть условия.

В наборе может быть несколько экземпляров записей одного подчиненного типа. Одни из них могут удовлетворять условиям на этот тип, другие нет. Введем понятие "выбор типа".

В системе (существует) возможны три способа проверки записей подчиненных типов:

ALL : тип считается "выбранным", если все экземпляры данного типа удовлетворяют заданным условиям.

ONE : тип считается "выбранным", если условиям удовлетворяет хотя бы один экземпляр записи данного типа, то есть проверка происходит до первого удовлетворяющего.

ONLY : тип считается "выбранным", если удовлетворяет хотя бы один экземпляр, но, в отличие от случая ONE , проверяются все экземпляры. При этом неудовлетворяющие помечаются "невыбран". Можно сказать, что выбираются только релевантные записи.

Такая проверка происходит по каждому типу записей. Если на некоторый тип условий нет, то он автоматически считается "выбранным". Так как в наборе есть только одна запись главного типа, то "выбор главного типа" совпадает с "выбором" этой записи.

Результаты проверки типов могут быть связаны между собой логическими операциями OR или AND . При этом если "тип выбран", считается, что результат проверки равен " true ". Главный тип всегда связан с подчиненными операцией AND , так как набор не может существовать без записи главного типа. Для подчиненных типов пользователь может задать операцию AND или OR . В случае AND набор считается "выбранным" в очередную порцию, если "выбраны" все типы записей. В случае OR набор "выбран", если "выбран" главный тип и хотя бы один из тех подчиненных, на которые заданы условия.

Таким образом, при AND проверка ведется до первого "невыбранного" типа, если таковой окажется. При OR проверку можно было бы вести до первого "выбранного" типа, считая и все остальные "выбранными". Но так как в каждом типе для проверки записей можно задать режим ONLY , который выбирает только удовлетворяющие экземпляры записей этого типа (релевантные), то проверка ведется по всем типам записей, на которые есть условия.

Теперь уточним, что значит, задать условия на тип записи. Каждый тип (или экземпляр) записи состоит из атрибутов (или их значений) трех типов SET , NAME и INT . Задать условия на запись значит указать, какие значения могут принимать атрибуты этих записей. Условия на атрибуты каждого типа задаются отдельно и связываются между собой логической операцией AND . Но внутри одного типа атрибутов условия на них могут быть, по желанию, связаны как операцией AND так и OR .

Условия на атрибуты типа INT и NAME записываются в виде триады:

< имя атрибута > < знак отношения > < значение атрибута >

Знаками отношения могут быть: = , > , < . Значением атрибута для типа INT должно быть целое число, для типа NAME – строка символов. На каждый атрибут типа INT или NAME могут быть наложены условия и со знаком > и со знаком < , тем самым задавая диапазон значений атрибута. Если заданы неоднозначные условия, то: если заданы условия с одинаковыми знаками, то во внимание принимается последнее; если заданы условия со знаком = и со знаком > или < , то берется условие со знаком = .

Условия вводятся в ответ на запрос системы вида:

"Введите условия на атрибуты типа INT"

Условия задаются через запятую на одной или нескольких строках. Каждая строка вводится отдельно нажатием клавиши "Ввод". Последняя строка оканчивается словом 'END' :

```
PERFO > 100, PERF < 1000□
USNUM=3,END□
```

Здесь записаны условия на атрибуты типа INT для записей типа PROGRM . Имена атрибутов можно сокращать.

Проверка на больше/меньше для строк символов производится обычным образом: 'POLY' > 'POL' , 'POL' > 'POCKET' . Проверку на равенство можно делать двояко : считать, что строки равны, только если они полностью совпадают ('AB = 'AB' , но 'ABC' ≠ 'AB') - сравнение на точное равенство; строки совпадают, если одна строка является частью другой ('ABC'='AB') - сравнение на включение. В системе по умолчанию принят первый тип сравнения. По желанию для любого типа записи можно задать и второй.

Примеры условий на атрибуты типа NAME для записей типа USERN :

NAME > MINUIT, NAME < MODIF,END□

NAME=FUMILI,END□

Условия на атрибуты типа SET задаются по-другому. Так как значениями атрибутов здесь являются множества, то и условия накладываются перечислением элементов множества. Например, для записей типа INTERN эти условия можно записать в виде:

EC,CDC,FORTR,SUPPRT□

ENTRY,FU,SUB,END□

Имена значений можно задавать не полностью, но так, чтобы не было неоднозначности. Например, FUNCT можно сократить только до FU . Если задать F , система возьмет FORTRN . Естественно, не обязательно задавать условия для каждого атрибута. Порядок перечисления элементов не существенен, можно записать:

EC,SUBR,FORT□

Система сама найдет, к какому атрибуту относятся значения. Например, здесь для атрибута COMPUT задано значение в виде множества {EC1060, CDC} , для LANG - {FORTRN} , для STATUS -{SUPPRT} для ENTRY - {ENTRY} , для TYPROC - {FUNCT, SUBR} . Назовем эти множества контрольными.

Условия на атрибуты типа SET заданы. Как их сравнить с реальными значениями атрибутов, заложенными в банке данных ? Сначала система выделяет контрольные множества, затем последовательно производит их сравнение с фактическими значениями тех атрибутов, на которые есть условия. Для нашего примера сначала сравнивается множество {EC1060, CDC} с множеством, являющимся значением атрибута COMPUT , затем множество {FORTRN} со значением атрибута LANG и т.д. Как уже говорилось, результаты сравнения могут быть связаны логическими операциями AND или OR. По умолчанию действует AND .

Таким образом, здесь каждый раз сравнивается два множества. Множества можно сравнивать по-разному. Обозначим контрольное множество

за SETC , а множество, являющееся значением некоторого атрибута, за SETA . Будем приводить примеры для атрибута COMPUT . Его значениями являются множества из элементов EC1060, CDC, BESM6 .

В системе заложено шесть типов сравнения множеств:

1)."Принадлежит, т.е. SETA⊆SETC .

Какие условия можно поставить с помощью этого типа сравнения?

Пусть SETC = {CDC} , тогда вопрос на обычном языке, соответствующий такому условию, выглядит так: выбрать программы, которые поставлены только на CDC-6500 . Только для тех записей, у которых атрибут COMPUT имеет значение SETA = {CDC} , результатом сравнения будет " true". SETC = {CDC,EC1060} означает: выбрать программы, которые поставлены хотя бы на одной из ЭВМ:CDC-6500 или ЕС-1060. Результатом сравнения будет " true " для SETA = {CDC} , SETA = {CDC, EC1060} и SETA = {EC1060} . Для всех других, например, SETA = {CDC, BESM6} , результат " false".

Заметим, что,если SETA=∅, пустое множество, то результат сравнения всегда " false ". Поэтому SETC = {CDC,EC1060, BESM6} имеет смысл задавать для исключения тех записей, у которых атрибут COMPUT не имеет значения (например, оно неизвестно).

2)."Включает", т.е. SETA⊇SETC

Какие условия можно задать с помощью этого типа ?

Пусть SETC = {CDC} : выбрать программы, которые поставлены на CDC (могут быть и на других ЭВМ).

Пусть SETC = {CDC, EC1060} : выбрать программы, которые поставлены и на CDC и на ЕС-1060 (могут быть и на БЭСМ-6).

SETC = {CDC, EC1060, BESM6} : выбрать программы, поставленные и на CDC , и на EC1060 и на БЭСМ-6.

3)."Равно", т.е. SETA≡SETC .

Пусть SETC = {CDC} :выбрать программы, которые поставлены только на CDC .

SETC = {CDC, EC1060} : программы и для CDC и для ЕС-1060 (которых нет на БЭСМ-6).

SETC = {CDC, EC1060, BESM6} : как в типе 2.

4)."Пересечение не пусто", т.е. SETA∩SETC≠∅

SETC = {CDC} : программы для CDC (как в типе 2).

SETC = {CDC, EC1060} : программы, которые поставлены либо на CDC , либо на ЕС-1060.

SETC = {CDC, EC1060, BESM6},как в типе 1.

5)."Пересечение пусто", т.е. SETA∩SETC≠∅

SETC = {CDC} : программы, которых нет на CDC .

SETC = {CDC,EC1060} : программы только для БЭСМ-6, как в типе 3.

SETC = {CDC, EC1060, BESM6} : программы, у которых атрибут COMPUT не имеет значения (см. тип I или 4).

6). "Не равно", т.е. SETA ≠ SETC.

SETC = {CDC} : все программы, кроме тех, что поставлены только на CDC .

SETC = {CDC, EC1060} : все программы, кроме тех, что поставлены и на CDC и на ЕС-1060.

SETC = {CDC, EC1060, BESM6} : все программы, кроме тех, что поставлены на трех машинах.

Для любого типа записи пользователь должен указать, по какому типу надо сравнивать множества. По умолчанию система использует тип 2.

Итак, все условия заданы. Система начинает выборку из текущей порции наборов, удовлетворяющих этим условиям. Мы получаем новую текущую порцию. На экран выдается краткая информация об этой порции: ее мощность, т.е. количество наборов, которые она содержит, а также количество записей каждого типа. Может оказаться, что новая порция пуста: нет ни одного релевантного набора. Тогда мы должны применить команду NEW , и текущей порцией станет весь банк данных.

4. SORT.

Как уже говорилось, можно ввести сразу всю команду, например:

SORT,KEY=NAME,REC=USN,INT␣

или по частям, т.е. SORT␣ и т.д. В данной версии режим набора в один прием не реализован.

В любой момент система работает с текущей порцией банка данных AI . Команда SORT предназначена для сортировки текущей порции: SORT(AI)→SI .

Если сортировка данной порции AI не производилась, то SI не определено.

В системе возможны два вида сортировки: по атрибутам типа NAME , в алфавитном порядке их значений, по атрибутам типа INT , в порядке возрастания их значений. Атрибут, по которому производится сортировка, назовем ключом сортировки.

Сортировка может проводиться по наборам и по записям. При сортировке по наборам сортируются только записи главного типа, набор не расчленяется, записи подчиненных типов остаются в наборах. В результате получается та же порция наборов AI , но расположенных в ином порядке.

Во втором случае сортировка производится по записям подчиненных типов или одного типа. Записи главного типа не участвуют в сортировке. Набор теперь уже не рассматривается как одно целое. Каждый экземпляр записей подчиненных типов участвует в сортировке как равноправный.

Если сортировка производится по записям нескольких подчиненных типов, то в каждом из них должен содержаться атрибут-ключ сортировки.

В результате сортировки получаем список, в котором могут быть перемешаны записи разных подчиненных типов (в простом случае одного типа) в некотором порядке, определяемом ключом сортировки. Отметим, что хотя набор и оказывается разорванным, но в списке для каждого экземпляра записи остается ссылка: к какому набору принадлежит данная запись.

В начале работы с командой SORT надо задать вид сортировки: по наборам или по записям. Затем система просит ввести имя атрибута-ключа сортировки.

При сортировке по наборам система проверяет , есть ли такой атрибут в записях главного типа. Если да, то определяется его тип: INT или NAME и производится сортировка. Если такого атрибута нет или он имеет тип SET , то система выдает сообщение об этом и просит повторить ввод ключа. Например, по ключам SPACE и COMMON можно вести сортировку наборов, а по ключам SPAC и COMPUT нельзя. Атрибута SPAC не существует, а COMPUT принадлежит к типу SET .

При сортировке по записям система последовательно проверяет, в каких записях подчиненных типов есть атрибут-ключ сортировки. Тип атрибута во всех типах записей должен быть одинаковым, либо INT либо NAME . Если в каком-то типе записей этот атрибут имеет тип NAME , то в последующих типах записей он ищется только среди атрибутов типа NAME . Если поиск прошел успешно, система выдает имена типов записей, по которым возможна сортировка с этим ключом. Например, для системы сопровождения библиотек атрибут NAME присутствует во всех записях подчиненных типов: USERN , INTERN и EXTERN . Пользователь затем указывает один или несколько типов, по которым затем производится сортировка.

Если ключ задан неправильно, система просит повторить ввод.

Если в ответ на запрос системы вместо ключа сортировки ввести пробел, то команда SORT заканчивает работу и сортировка не производится.

Команду SORT можно применять к одной порции AI любое количество раз, но в любой момент времени существовать может только одна отсортированная порция - последняя, т.е. все предыдущие теряются. Так же, если мы к порции AI применим команду SELECT и получим A(I+1) , то отсортированная порция SI становится недоступной. Таким образом, порция SI всегда соответствует текущей порции либо не определена.

5. PRINT

Эта команда выдает на АЦПУ или экран терминала информацию о текущей порции AI или SI . В режиме ввода в один прием команда может выглядеть так:

```
PRINT,FORMAT,BIL,REC=PROG,ATR=AU,CO,REC=US,ATR=NAME␣
```

Этот режим не реализован, и всю информацию о работе команды система получает от пользователя во время диалога.

Каждая выдача команды PRINT состоит из двух частей: "шапки" и распечатки текущей порции. В "шапке" содержится информация о том, как данная порция была получена и, тем самым, какими свойствами она обладает. Заметим, что без такой шапки любая выдача была бы бесполезна, поэтому она выдается автоматически.

Команда SELECT выбирает из банка данных порцию наборов, удовлетворяющих некоторым условиям. Эти условия в определенном виде и записываются в "шапку". Но команды SELECT и PRINT могут набираться независимо друг от друга и необязательно друг за другом. Например, мы можем обратиться к команде SELECT несколько раз подряд. Тогда в результате получается порция, удовлетворяющая сумме отдельных условий. Что записывается в "шапку" в этом случае ?

После каждой работы команды SELECT условия выбора каждой порции запоминаются в некотором буфере. Когда мы в конце концов наберем PRINT , все эти условия из буфера и будут выданы в "шапке", причем условия, относящиеся к разным порциям, будут разделены. Но если теперь набрать несколько раз команду PRINT , то "шапки" в выдаче будут отсутствовать (вернее, в ней останется только первая строка). Они полностью совпадают с первой "шапкой", так как соответствуют одной текущей порции, и поэтому не выдаются.

Если мы хотим распечатать отсортированную порцию, то в "шапке" добавятся сведения о том, какого вида была сортировка, имена типов записей и ключ сортировки. Заметим, что, если производилось несколько сортировок одной порции, то информация выдается только о последней.

Таким образом, по выдаче на АЦПУ мы всегда можем проследить историю работы с банком данных. Буфер периодически заполняется условиями из команды SELECT и очищается при выдаче команды PRINT . Однако, если при этом мы подавляем печать на АЦПУ (оставляя только выход на экран терминала), то информация в буфере сохраняется. Его очищение происходит только при получении "твердой копии" буфера на листинге (или каком-то файле). На экран терминала всегда выдается такой объем информации, какой должен идти на АЦПУ.

Вывод команды PRINT делится на страницы. В начале каждой страницы печатается системная константа, в случае системы сопровождения это CVARSYST='LISS/1.8 ДУБНА' , а также текущая дата в виде 23/04/83 и номер текущей страницы.

Существует два вида печати: форматная и простая. При форматной печати каждая логическая страница выдается с новой физической страницы. При простой печать идет в сжатом виде, без пропусков. Также можно задать распечатку в виде билистинга.

Пользователь может регулировать размеры страницы. По умолчанию распечатка производится на страницы размером 128x70 или 160x30. Минимальный размер строки и страницы, который можно задать, равен 10. Небольшую длину строки можно использовать, например, для распечатки столбиком.

Вторая часть выдачи команды PRINT содержит информацию о текущей порции. Рассмотрим распечатку несортированной текущей порции, т.е. AI . Порция состоит из некоторого количества наборов, каждый из которых содержит несколько экземпляров записей разных типов.

Пользователь может задать распечатку любого объема из этой порции. Во-первых, он может указать, записи каких типов ему нужны, во-вторых, значения каких атрибутов из этих записей надо распечатать. При этом можно явно перечислить имена этих атрибутов или воспользоваться стандартными списками имен, заложенными в системе.

Если в ответ на запрос системы -ввести имена атрибутов - ответить 'ALL' , это означает, что для данного типа записей распечатке подлежат все атрибуты. Если ответить 'STAND' , то будет распечатываться стандартный список атрибутов, например, для записей типа USERN это следующие атрибуты : NAME,COMPUT,LANG,TYPE.

Из команды HELP можно узнать, какой список атрибутов является стандартным. В процессе работы с системой их можно изменять.

Когда система знает все, что ей нужно, начинается процесс печати. Вначале печатается заголовок, в котором содержатся названия всех атрибутов, подлежащих распечатке. Значения печатаются точно под именем соответствующего атрибута.

Если возможно, система пытается разместить информацию на странице в наиболее компактном и удобочитаемом виде. Если позволяет размер строки, заданный пользователем, все записи одного набора выдаются в одну строку. Вернее, на одну строку помещается по одному экземпляру от каждого типа записи. Далее выводится по второму экземпляру каждой записи, причем записи одного типа печатаются точно друг под другом, и так, пока не будет распечатан весь набор. Если для какого-то типа записи исчерпаются раньше, то соответствующие места остаются пустыми. Система пытается также разместить несколько наборов в одной строке.

Это наиболее удобочитаемые виды выдачи. Записи одного типа находятся в одной колонке и четко отделяются от записей других типов. Значения одинаковых атрибутов выдаются в одном столбце, возглавляемом именем атрибута.

Если же набор не помещается на строке, система сначала печатает запись главного типа на одной или нескольких строках, а затем пытается разместить записи подчиненных типов на одной строке. И только если это не получается, каждая запись начинает распечатываться с отдельной строки. Если какая-либо запись не помещается на одной строке, то она продолжается на любое нужное количество строк. Отметим, что расположение имен в заголовке всегда соответствует расположению атрибутов в записях.

Как выводится каждая запись ? Сначала выдаются значения атрибутов типа NAME , затем SET и, наконец, INT . Если атрибут имеет несколько значений, то все они печатаются друг под другом в одной колонке. Так же выводятся элементы множества, являющегося значением атрибутов типа SET . Таким образом, одна запись может расположиться на нескольких строках либо вследствие того, что одной строки не хватает для всех атрибутов записи, либо из-за множественности значений некоторых атрибутов.

Если значение некоторого атрибута не определено, например, неизвестно, вместо него выдаются звездочки: ****** . Когда выбор подчиненных типов производится с помощью ONLY , некоторые экземпляры могут оказаться с пометкой "не выбран". Обычно они на печать не выдаются. Но можно заказать их выдачу, тогда они помечаются звездочкой * .

Распечатка текущей порции, отсортированной по наборам, происходит точно так же, как и несортированной, только в шапке указывается, по какому атрибуту шла сортировка.

Выдача порции SI , отсортированной по записям, происходит несколько иначе. Здесь пользователь тоже может указать, какие атрибуты он бы хотел распечатать. Но, так как в порции могут находиться записи разных типов, то все эти атрибуты обязательно должны принадлежать каждому из этих типов. Например, мы произвели сортировку по атрибуту NAME записей трех типов: USERN, INTERN и EXTERN . Тогда на печать мы можем выдать самое большее два атрибута: NAME и TYPE , только они присутствуют во всех типах. Если же сортировка проводилась по типам USERN и INTERN , то здесь ограничений на выдаваемые атрибуты нет, так как записи этих типов совпадают по своей структуре.

Заметим, что атрибут - ключ сортировки выдается всегда, даже если пользователь его не закажет, и стоит на первом месте.


```
LISS/1.8 ДУБНА                                  19/09/85     PAGE  1

-------УСЛОВИЯ ВЫБОРА ОБЪЕКТОВ ИЗ БАНКА ДАННЫХ:ПОРЦИЯ A1------
                    *** ТИП PROGRM ***

....УСЛОВИЯ НА АТРИБУТЫ ТИПА SET (SETA ⊃ SETC)
*COMPUT=EC1060 CDC     AND STATUS=SUPPRT AND REFILE=INPUT

....УСЛОВИЯ НА АТРИБУТЫ ТИПА NAME('ABC'='AB')
*D100<=INDEX<=D500

....УСЛОВИЯ НА АТРИБУТЫ ТИПА INT
*USNUM =1

==========ВСЕ СЛЕДУЮЩИЕ ТИПЫ СВЯЗАНЫ ОПЕРАЦИЕЙ AND==========
                    *** ТИП EXTERN *** ONE  ***

....УСЛОВИЯ НА АТРИБУТЫ ТИПА NAME('ABC'≠'AB')
*INDEX =USER
                    *** ТИП USERN  *** ONLY ***

....УСЛОВИЯ НА АТРИБУТЫ ТИПА SET (SETA ∩ SETC = ∅)
*ENTRYP=ENTRY

--------УСЛОВИЯ ВЫБОРА ОБЪЕКТОВ ИЗ БАНКА ДАННЫХ:ПОРЦИЯ A2------
                    *** ТИП INTERN *** ALL  ***

....УСЛОВИЯ НА АТРИБУТЫ ТИПА SET (SETA = SETC)
*LANG  =FORTRN  OR TYPROC=SUBR

INDEX   COMPUT REFILE NAME   TYPE   NAME   TYPE   NAME   TYPE
D115    EC1060 INPUT  CHEBQU USERN  D115BD INTERN F      EXTERN
        CDC
        BESM6
D300    EC1060 INPUT  EPDE1  USERN  EPDCJ  INTERN GETCO  EXTERN
        CDC    OUTPUT                EPDCHK INTERN USER1  EXTERN
        BESM6  TAPE                  EPDBPS INTERN USER2  EXTERN

 LISS/1.8 ДУБНА                                 19/09/85     PAGE  2

INDEX   COMMON NAME   TYPROC NAME   LANG   TYPROC NAME   INDEX
D115    DPCHEB CHEBQU FUNCT  D115BD FORTRN SUBR   F      USER
        CHEINT
D300    BLANK  EPDE1  SUBR   EPDCJ  FORTRN SUBR   GETCO  USER
                             EPDCHK FORTRN SUBR   USER1  USER
                             EPDBPS FORTRN SUBR   USER2  USER

 LISS/1.8 ДУБНА                                 19/09/85     PAGE  3

СОРТИРОВКА ПО 'NAME  ' В ТИПАХ 'USERN ' 'INTERN'
NAME    TYPE   INDEX
CHEBQU  USERN  D115
D115BD  INTERN D115
EPDBPS  INTERN D300
EPDCHK  INTERN D300
EPDCJ   INTERN D300
EPDE1   USERN  D300
```

РИС.2. ПРИМЕР РАБОТЫ КОМАНДЫ PRINT.

Как уже говорилось, хотя при такой сортировке наборы и распадаются, но для каждой записи сохраняется ссылка - к какому набору она принадлежит. Поэтому, по желанию пользователя, команда PRINT может также вместе с каждой записью выдать и ключ этого набора, для системы сопровождения LISS это индекс программы INDEX .

На рис.2 приведен пример выдачи команды PRINT . Из "шапки" видно, что сначала при помощи команды SELECT из банка данных была выбрана порция A1 . Это программы, которые поддерживаются и на CDC-6500 и на ЕС-1060, читают данные из файла INPUT , имеют индекс между D100 и D500 , ровно один вход для пользователя и хотя бы одно обращение к подпрограмме, составленной пользователем. Из подпрограмм типа USERN выбраны только те, которые не имеют ENTRY входов.

Порция A1 не распечатывалась. Затем из A1 были выбраны только те программы, у которых либо ни одна внутренняя подпрограмма не имеет тип FUNCTION , либо все они написаны на языке FORTRAN .

Здесь показаны также три выдачи из порции A2 и S2 , которые различаются набором выдаваемых атрибутов.

Приведенный пример работы команды PRINT , конечно, специально построен так, чтобы показать, насколько сложные условия можно задавать в команде SELECT и как их распечатывает команда PRINT . На практике обычно чаще используются режимы по умолчанию и запросы проще.

Если заказана выдача на АЦПУ, то нам приходится ждать, пока система выведет всю информацию в буфер. Если объем большой, то время ожидания может быть больше минуты.

При выдаче на экран терминала после каждых десяти строк система ждет подтверждения - продолжать ли вывод. Мы можем отказаться от выдачи вводом любого символа, кроме пробела, или подтвердить ее нажатием клавиши "Ввод". Чтобы не выводить ненужную информацию на АЦПУ, можно сначала посмотреть на экране несколько первых десятков строк и лишь затем отправить ее на печать.

6. LIBR

Система LISS имеет одну, так сказать, библиотечно-зависимую функцию. Ее реализует команда LIBR . Она не будет использоваться при других приложениях системы.

Данное приложение, система сопровождения библиотек, обслуживает вполне конкретную библиотеку. Библиотека, чтобы ею можно было пользоваться, должна быть оттранслирована в библиотеку загрузочных модулей. Эта библиотека имеет реальное существование, она состоит из определенных подпрограмм, с определенными именами, дополнительными точками входов (алиасами). Хотелось бы, чтобы это реальное содержание библиотеки соответствовало той информации о ней, которая заложена в

банке данных. Команда LIBR проводит проверку этого соответствия по именам подпрограмм и ENTRY -входам.

Реальная библиотека задается в DD-карте (ОС ЕС):

```
//LIBRARY DD DSN=SYS1.DUBNA, DISP=SHR,DCB=(RECFM=F, BLKSIZE=256)
```

По оглавлению этой библиотеки система строит таблицу, в которой отмечены все имена, а также точки входа, являющиеся алиасами. Затем эта таблица сравнивается с текущей порцией SI , представляющей собой список записей одного или нескольких из типов USERN, INTERN и EXTERN , отсортированных по атрибуту NAME . Для сравнения из каждой записи используется только два атрибута NAME и ENTRY . Те подпрограммы, у которых атрибут ENTRY имеет значение ENTRY , считаются алиасами, а остальные основными точками входов.

В результате работы команда LIBR выдает на экран и/или печать список из трех столбцов. В первом находятся имена, которые имеются и в таблице и в порции SI , т.е. совпадающие имена. Во втором столбце - имена, которые есть только в таблице, и в третьем - имена, которые есть только в порции SI .

Имена считаются совпадающими, если совпадает и имя подпрограммы, и признак, показывающий, является ли оно алиасом или нет. Если имена одинаковы, а признаки нет, то печатается два имени в разных столбцах: одно с признаком алиаса, другое без него.

Фрагмент выдачи команды LIBR:

СОВПАДАЮТ	НЕТ В SI	НЕТ В Б-КЕ
FINTED		
FINTER		
		FIO99X
	*FIO99X	
*FIO999		
	FIXPAR	
FLOARG		
		FLUCT

Здесь звездочка означает, что имя является алиасом (или ENTRY -входом).

Из такой таблицы хорошо видно, на какие подпрограммы следует обратить внимание. Например, о подпрограмме FIO99X , по-видимому, в банке данных представлена неправильная информация. Имя FIXPAR либо забыли внести в банк данных, либо это устаревшая подпрограмма, которую забыли исключить из библиотеки загрузочных модулей и т.п.

Так как библиотеки на каждой машине имеют свою специфику, то перед командой LIBR надо подготовить порцию SI , отвечающую конкретной ЭВМ, на которой реализована система. В нашем случае необходимо с помощью команды SELECT выбрать программы со следующими характеристиками: STATUS=SUPPRT, COMPUT=EC1060 . Затем отсортировать в полученной порции записи типов USERN и INTERN по атрибуту NAME . После этого можно применять команду LIBR .

Но эту команду можно использовать и для таких, например, целей: проверить, какие из программ на языке FORTRAN есть в библиотеке, или какие из программ, поставленных на CDC-6500 и БЭСМ-6, находятся в библиотеке на ЕС-1060 и каких нет и т.п. То есть любую порцию, отсортированную по атрибуту NAME , можно подать на вход команды LIBR .

Отметим, что, если последняя сортировка производилась по наборам, а не по записям, или по атрибуту типа INT , то команда LIBR выдаст сообщение и закончит работу.

Так же, как в команде PRINT , выдача идет постранично. Здесь можно также задать форматную (с разделением на страницы) печать.

По-видимому, описанную функцию системы можно расширить так, чтобы проверялось и соответствие реальных библиотек (или просто файлов) исходных модулей, тестов и т.д., например, на существование, язык программирования, тип подпрограммы, тип точки входа, количество перфокарт, common -блоки и т.п. То есть почти по всем атрибутам, заложенным в банке данных. Можно было бы поставить и обратную задачу: из существующих реально файлов и библиотек извлекать информацию для банка данных, что позволит автоматизировать его заполнение.

7. STAT.

Эта команда в настоящей версии системы не реализована. Предполагается с ее помощью выдавать различную статистическую информацию по текущим порциям AI и SI .

8. NEW.

Эта команда, в некотором смысле, обратная команде SELECT . Она возвращает нас (систему) в исходное состояние, т.е. для любого I :

NEW(AI) ⟶ AO

Текущей порцией становится опять весь банк данных, и все, что делалось до этого, забывается (становится недоступным).

Банк данных является объединением некоторого количества областей, каждая из которых имеет свое имя. С помощью команды NEW мы можем уменьшить число областей, входящих в порцию AO. Для этого надо задать системе список нужных областей в виде (для системы сопровождения библиотек):

A,C-E,G,K-Z.

9. END.

По этой команде система завершает свою работу.

10. Реализация системы.

Первый вариант системы реализован средствами языка Паскаль-6000 на ЕС-1060 в ОС мут и ЕС-1061 в ОС SVS . Запуск системы осуществляется через систему TERM /7/.

Объем программы составляет 3000 строк на паскале и одна небольшая программа на ассемблере.

В следующих вариантах система будет расширена, в ней появятся средства по архивной работе с текстами программ и описаний, средства управления версиями и другие.

Литература

1. Гласс Р., Нуазо Р. Сопровождение программного обеспечения. М., Мир, 1983.
2. Федорова Р.Н., Хасанов А.М. ОИЯИ, П-82-674, Дубна, 1982.
3. Лукстиня Л. и др. ОИЯИ, Р11-85-170, Дубна, 1985.
4. Мартин Дж. Организация баз данных в вычислительных системах. М., Мир, 1978.
5. Current trends in programming methodology, Prentice-holl, Inc., Englewood cliffs, New Jersey, 1977.
6. CERN Computer Centre, Program Library - I,II.
7. Кореньков В.В. ОИЯИ, П-84-316, Дубна, 1984.

Рукопись поступила в издательский отдел
25 ноября 1985 года.

Хасанов А.М. 11-85-846

LISS – Система сопровождения библиотек программ. Команды системы

В работе содержится описание функциональных возможностей системы сопровождения библиотек программ /Library Support System/. Система позволяет сопровождающему программисту:
- содержать банк данных системы в соответствии с историей развития библиотеки,
- оперативно получать разнообразную информацию по современному состоянию библиотеки и по ее развитию;
- выводить на терминал или печать различные документы и справки по библиотеке в форматированном виде.

Система предоставляет также помощь в обучении работе с ней.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна 1985

Перевод О.С.Виноградовой

Khasanov A.M. 11-85-846

LISS - a Library Support System. System Commands

The library support system (LISS) functional facilities is described. The system enables the programmer: to maintain the system data base in accordance with library history; to get quickly the different information about the present state of the library and its development; to output on terminal and printer different documents about the library in formative mode The system guide is included in the system.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Communication of the Joint Institute for Nuclear Research. Dubna 1985