Ввод-вывод данных в консоли

Основная функция для считывания вводимых пользователем данных:

var:read(expr1, . . ., exprn);

Выражения **expr1, . . ., exprn** при вводе интерпретируются и выводятся на консоль.

Аргументы функции **read** могут включать подсказку.

Вводимое значение помещается в **var**.

Пример:

```
(%i1)a:42$
(%i2)a:read("Значение a = ",a," введите новую
величину");
Значение a = 42 введите новую величину (p+q)^3;
```

$$(\%o2) \quad (q+p)^3$$

```
(%i3)    display(a);
```

$$a = (q+p)^3$$

$$(\%o3) \quad done$$

Функция readonly осуществляет только ввод данных

Пример:

сравнение использования функций **read** и **readonly**:

```
(%i1)    a:7$
(%i2)    readonly("Введите выражение:");

Введите выражение: 2^a;
(%o2)  2^a
(%i3)    read("Введите выражение:");
Введите выражение: 2^a;
(%o3)  128
```

Вывод на экран

Вывод на экран осуществляется функцией **display**.

Синтаксис её вызова:

display(expr1, . . ., exprn);.

Выражения из списка аргументов выводятся слева направо –

сначала само выражение, а затем после знака равенства — его значение.

Аналогичная функция **disp** - синтаксис вызова:

disp (expr1, . . ., exprn);

выводит на экран только значение выражения после его интерпретации.

Функция **grind** осуществляет вывод в консоль **Maxima** аналогично **disp**, но в форме, удобной для ввода с клавиатуры.

Вывод на экран - пример

```
(%i1)  a:1$ b:2$ c:3$
(%i4)  display(a,b,c);
```

$$a = 1$$

$$b = 2$$

$$c = 3$$

$$(\%o4)\ done$$

```
(%i5)  disp(a,b,c);
```

1

2

3

$$(\%o5)\ done$$

```
(%i6)  grind(a);
```

1

$$(\%o6)\ done$$

Установки *флагов – глобальных переменных*

Переменная linel определяет длину строки, в которую должна вписываться выдача. Изначально установлена величина ”79”.

Если желательно получить более узкую страницу (например, 60 позиций для двух колоночной печати), следует присвоить переменной **linel** значение ”60”:

linel:60;

Установки *флагов – глобальных переменных*

Переменная display2d включает или выключает ”двумерное” рисование дробей, степеней, и т.п. Изначально установлено значение ”**true**”.

При этом дроби рисуются красиво, но выдача не может быть использована для ввода в MAXIM’у.

Если установить значение ”**false**”, то вывод может быть впоследствии использован как ввод.

Пример - display2d

```
(x^2+a)/(y^2+b);
```

$$\frac{x^2+a}{y^2+b}$$

```
display2d:false$
(x^2+a)/(y^2+b);
```

```
(x^2+a)/(y^2+b)
```

Переменная (флаг) display_format_internal

Переменная (флаг) display_format_internal включает или выключает вывод с использованием «внутреннего» представления выражения или в виде «стандартного»алгебраического. Пример:

```
(%i13) a:(x+y+z)^5;
                                                    5
(%o18)                                       (x + y + z)
(%i14) display_format_internal:false;
(%o14)                                          false
(%i15) a/b;
                                                    5
                                             (z + y + x)
(%o15)                                       -----------
                                                  b
(%i16) display_format_internal:true;
(%o16)                                          true
(%i17) a/b;
                                           - 1                5
(%o17)                                    b     (x + y + z)
```

Переменные (флаги) ibase и obase

Переменные (флаги) ibase и **obase** (по умолчанию 10) задают значение системы счисления целых чисел при вводе/выводе. Пример:

```
(%i23) a:255;
(%o23)                                  0FF
(%i24) obase:10;
(%o36)                                  10
(%i37) a:255;
(%o37)                                  255
(%i38) obase:16;
(%o26)                                  10
(%i27) a;
(%o27)                                  0FF
(%i28) obase:2;
(%o101000)                              10
(%i101001) a;
(%o101001)                          11111111
```

Печать времени выполнения

Переменная `showtime` включает или выключает печать времени, затраченного на каждое действие.

Изначально установлено значение **`false`**.

Если установить значение **`true`**, то будет печататься:

- процессорное время, в течение которого выполнялась выкладка,
- физическое время, затраченное на выкладку.

```
showtime:true$
                    Evaluation took 0.00 seconds (0.00 elapsed)
fun1(a,b,c)$
                    Evaluation took 4.08 seconds (4.20 elapsed)
```

Функции dispterms и print

Вывод на экран длинных выражений по частям (одна часть над другой) осуществляется функцией **dispterms**. Синтаксис вызова: **dispterms(expr).**

Кроме того, для вывода результатов вычислений используется функция **print**. Синтаксис вызова:

print(expr1, . . ., exprn);.

Выражения **(expr1, . . ., exprn)** интерпретируются и выводятся последовательно в строчку (в отличие от вывода, генерируемого функцией display). Функция **print** возвращает значение последнего интерпретированного выражения.

Примеры функций print и display

```
(%i1)  a:1$ b:2$ c:(a^2+b^2)$
(%i4)  rez:print("Пример:",a,b,c);

Пример: 1 2 5
(%o4) 5
(%i5)  rez;
(%o5) 5
(%i6)  display("Пример:",a,b,c);

Пример:=Пример:
```

$$a = 1$$

$$b = 2$$

$$c = 5$$

$$(\%o6)\ done$$

Файловые операции ввода-вывода

Ввод-вывод текстовых данных

Сохранение текущего состояния рабочей области **Maxima** осуществляется при помощи функции **save**. Эта функция позволяет сохранить в файле отдельные объекты с указанными именами.

Варианты вызова **save**:

save(filename, name1, name2, . . .);

— сохраняет текущие значения переменных

name1, name2, . . . в файле **filename**.

Аргументы должны быть именами переменных, функций или других объектов. Если имя не ассоциируется с какой-либо величиной в памяти, оно игнорируется. Функция **save** возвращает имя файла, в который сохранены заданные объекты.

save(filename,values, functions, labels, . . .); — сохраняет все значения переменных, функций, меток и т.п.

save(filename, [m,n]); — сохраняет все значения меток ввода/вывода в промежутке от **m** до **n** (**m,n** — целые литералы).

save(filename, name1=expr1, . . .); — позволяет сохранить объекты **Maxima** с заменой имени **expr1** на имя **name1** и т.д..

save(filename, all); — сохраняет все объекты, имеющиеся в памяти.

Глобальный флаг file_output_append

Глобальный флаг file_output_append управляет режимом записи.

Если **file_output_append:true**, то результаты вывода **save** добавляются в конец файла результатов. Иначе файл результата переписывается.

Вне зависимости от **file_output_append**, если файл результатов не существует, то он создаётся.

Данные, сохранённые функцией **save**, могут быть снова загружены функцией **load**.

Варианты записи при помощи **save** могут совмещаться друг с другом, например -

save(filename, aa, bb, functions, . . .);.

Загрузка предварительно сохранённого функцией **save** файла осуществляется функцией **load(filename)**.

Функция stringout

Аналогичный синтаксис и у функции **stringout**, которая предназначена для вывода в файл выражений **Maxima** в формате, пригодном для последующего считывания **Maxima.**

Синтаксис вызова **stringout**:

stringout(filename, name1, name2, . . .);

stringout(filename, [m,n]);

stringout(filename, values);

stringout(filename, functions);

stringout(filename, name1=expr1, . . .);

Функция load(filename)

Функция **load(filename)** вычисляет выражения в файле **filename**, создавая, таким образом, переменные, функции, и другие объекты **Maxima.**

Если объект с некоторым именем уже присутствует в **Maxima**, при выполнении **load** он будет замещён считываемым.

Чтобы найти загружаемый файл, функция **load** использует переменные **file_search**, **file_search_maxima** и **file_search_lisp** как справочники поиска.

Если загружаемый файл не найден, печатается сообщение об ошибке.

Загрузка работает одинаково хорошо для кода на **Lisp** и кода на макроязыке **Maxima.**

Файлы, созданные функциями **save, translate_file, compile_file** содержат код на **Lisp**, а созданные при помощи функции **stringout** содержат код **Maxima**.

Все эти файлы могут с равным успехом быть обработаны функцией **load**.

load использует функцию **loadfile**, чтобы загрузить файлы **Lisp** и

batchload, чтобы загрузить файлы **Maxima.**

Load не распознаёт конструкции **:lisp** в файлах, содержащих код на **Maxima**, а также глобальные переменные **_, __, %**, и **%th**, пока не будут созданы соответствующие объекты в памяти.

Функция **loadfile(**filename**)** предназначена для загрузки файлов, содержащих код на **Lisp**, созданные функциями **save, translate_file, compile_file**.

Для задач конечного пользователя удобнее функция **load**.

Протокол сессии Maxima может записываться при помощи функции **writefile**

Он записывается в формате вывода на консоль.

Для тех же целей используется функция **appendfile** - запись в конец существующего файла.

Завершение записи и закрытие файла протокола осуществляется функцией **closefile**.

Синтаксис вызова:

writefile(filename**)**
closefile(filename**)**
appendfile(filename).

Ввод-вывод командных файлов

Основная функция, предназначенная для ввода и интерпретации командных файлов — функция **batch**(filename). Функция **batch** читает выражения **Maxima** из файла filename и выполняет их.

Функция **batch** отыскивает filename в списке **file_search_maxima**. В файл filename включают последовательность выражений **Maxima**, каждое из которых должно оканчиваться **;** или **$**.

Специальная переменная **%** и функция **%th** обращаются к предыдущим результатам в пределах файла. Файл может включать конструкции **:lisp**.

Пробелы, табуляции, символы конца строки в файле игнорируются.

Подходящий входной файл может быть создан редактором текста или функцией **stringout**.

Ввод-вывод командных файлов

Функция **batch** считывает каждое выражение из файла filename, показывает ввод в консоли, вычисляет соответствующие выражения и показывает вывод также в консоли.

Метки ввода назначаются входным выражениям, метки вывода — результатам вычислений, функция **batch** интерпретирует каждое входное выражение, пока не будет достигнут конец файла.

Если предполагается реакция пользователя (ввод с клавиатуры), выполнение **batch** приостанавливается до завершения ввода

.

Для остановки выполнения batch-файла используется Ctrl-C.

Ввод-вывод командных файлов

Функция **batchload**(filename) считывает и интерпретирует выражения из командного файла, но не выводит на консоль входных и выходных выражений. Метки ввода и вывода выражениям, встречающимся в командном файле, также не назначаются.

Специальная переменная **%** и функция **%th** обращаются к предыдущим диалоговым меткам, не имея результатов в пределах файла.

Кроме того, файл filename не может включать конструкции **:lisp**.

Этой функцией «удобно подгружать» наборы библиотечных функций или пользователя.

Ввод-вывод командных файлов

В случае необходимости «прервать пакетное выполнение» можно воспользоваться, например, таким кодом (в файле abort.max):

```
print("**** ошибка ...не корректое .... *****");
 /* lisp (setq *debugger-hook* nil), */
lisp (setq *debugger-hook* 'abort);
read("Нажмите ctrl-C");
print(" dont print it.....");
```

Вызов выполнения batchload('d:\\test\\maxima\\abort.max");