Функциональное и логическое программирование

Лекция 8

2.13 Строки

Строка – последовательность символов, заключенная в двойные кавычки.

Предикаты для работы со строками:

1. string_length(S, L)

Определяет длину строки S.

Пример 1:

?- string_length("asds f",L).
L = 6.

?- string_length('asds f',L).
L = 6.

?- string_length(asd,L).
L = 3.

3. sub_string(S,K,N,R,S1)

Выделяет в строке S подстроку S1, которая начинается с K-го элемента и содержит N символов. R – количество символов, стоящих в S после подстроки S1. Нумерация элементов строки начинается с 0.

Пример 3:

?- sub_string("asdf gh h",3,4,_,S).

S = "f gh".

?- sub_string('asdf gh h',3,4,_,S).

S = "f gh".

4. string_chars(S,L)

Преобразует строку S в список символов и наоборот.

Пример 4:

?- string_chars("asd",L).

L = [a, s, d].

?- string_chars(S,[a, s, d]).

S = "asd".

?- string_chars(asd,L).

L = [a, s, d].

5. string_to_list(S,L)

Преобразует строку S в список кодов символов и наоборот.

Пример 5:

?- string_to_list(asdf,L).

L = [97, 115, 100, 102].

?- string_to_list(S,[97, 115, 100, 102]).

S = "asdf".

6. char_code(C,K)

Преобразует символ C в его код K и наоборот.

Пример 6:

?- char_code(d,K).

K = 100.

?- char_code(X,100).

X = d.

7. split_string(S,R,D,L)

Преобразует строку S в список подстрок L, используя R как разделитель, удаляя из начала и конца подстрок символы строки D.

Пример 7:

```
?- split_string("dfgh 123    l: fg; .fgg,"," ",":;.,",L).
L = ["dfgh", "123", "", "", "", "l", "fg", "fgg"].
```

Если хотим удалить пустые строки, то необходимо добавить пробел в строку для удаления символов:

```
?- split_string("dfgh 123    l: fg; .fgg,"," "," :;.,",L).
L = ["dfgh", "123", "l", "fg", "fgg"].
```

8. atomic_list_concat(L,R,S)

Преобразует список L в строку S, используя R как разделитель и наоборот.

Пример 8:

?- atomic_list_concat([a,s,d,f],"**",S).

S = 'a**s**d**f'.

?- atomic_list_concat(X," ","qw dfg 123").

X = [qw, dfg, '123'].

9. number_string(N,S)

Преобразует число N в строку S и наоборот.

Пример 9:

?- number_string(10,S).

S = "10".

?- number_string(X,"1000").

X = 1000.

10. atom_string(A,S)

Преобразует атом A в строку S и наоборот.

Пример 10:

?- atom_string(asd,S).

S = "asd".

Пример 11:

Предикат, который преобразует строку S в строку S1, удаляя все пробелы.

```
del_probel(S,S1):-string_chars(S,L),delete(L,' ',L1),
                  string_chars(S1,L1).
```

Пример 12:

Предикат, который считает кол-во вхождений символа C в строку S.

```
goal:-read(S),read(C),
        string_chars(S,L),
        char_count(L,C,N),
        writeln(N).
char_count([],_,0):-!.
char_count([C|T],C,N):-char_count(T,C,N1),N is N1+1,!.
char_count([_|T],C,N):-char_count(T,C,N).
```

2.14 Предикаты для работы с файлами

1. exists_file(<‘имя файла’>)

Завершается успешно, если файл с указанным именем существует (обратные слэши дублируются).

2. open(<'имя файла’>, <режим>,F)

Открытие файла для чтения, записи или добавления.

Режимы:

- read (для чтения);
- write (для записи);
- append (для добавления).

F – файловая переменная

Предикаты для работы с файлами являются внелогическими. Чтение и обработку данных следует выполнять отдельно!

3. set_input(F)

   set_output(F)

Перенаправление ввода из файла или вывода в файл.

4. close(F)

Закрытие файла.

5. see(<'имя файла‘>)

   tell(<'имя файла‘>)

Открытие и перенаправления ввода из файла или вывода в файл вместо 2 и 3.

При перенаправлении ввода на клавиатуру, а вывода на экран в качестве имени файла используют имя user.

6. seen

   told

Закрытие файлов, открытых с помощью see и tell.

7. seeing(F)

telling(F)

Связывает F с именем файла, являющегося текущим входным или выходным потоком.

8. at_end_of_stream

Успешно завершается, если найден конец файла.

9. read_line_to_codes(F,L)

Читает строку из входного потока F и преобразует ее в список кодов символов этой строки (без кода перевода строки).

10. read_stream_to_codes(F,L)

Читает содержимое из входного потока F (до конца файла) и преобразует его в список кодов символов (включая коды перевода строки 10).

Пример 1: Считаем файл in.txt в список кодов.

```
goal1:-see('in.txt'),seeing(F),read_stream_to_codes(F,L),writeln(L).
```

Пример 2:

Написать предикат, который выводит на экран строки файла, начиная с некоторого номера.

Имя файла и номер строки вводятся с клавиатуры.

```
goal:-writeln('Введите имя файла'),read(Filename),
        check_exist(Filename),
        writeln('Введите номер строки'), read(N),
        open(Filename,read,F), set_input(F),
        read_file(F,N),
        format('Содержимое файла, начиная с ~w строки\n',N),
        write_screen(F), close(F).
check_exist(Filename):-exists_file(Filename),!.
check_exist(_):-writeln(‘Такого файла нет‘),fail.
read_file(_,_):-at_end_of_stream, !,
                    format('В файле меньше, чем ~w строк\n',N),
                     fail.
read_file(_,1):-!.
read_file(F,N):-read_line_to_codes(F,_), N1 is N-1, read_file(F,N1).
write_screen(_):-at_end_of_stream,!.
write_screen(F):- read_line_to_codes(F,L), string_to_list(S,L),
                        writeln(S),
                        write_screen(F).
```

Пример 3:

Написать предикат, который записывает вводимые с клавиатуры строки в файл out.txt. Окончание ввода – строка ‘#’.

```
goal3:-writeln('Введите строку для записи в файл'), read(S),
        tell('out.txt'),
        write_to_file(S),
        told,
        writeln('Строки записаны в файл').
write_to_file('#'):-!.
write_to_file(S):-writeln(S),
                  read(S1),
                  write_to_file(S1).
```