

Programowanie w Logice

Wejście i wyjście

Przemysław Kobylański
na podstawie [CM2003]

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Czytanie i pisanie termów

- ▶ Term czyta się ze standardowego wejścia predykatem `read/1`.
- ▶ Każdy wczytywany term powinien być zakończony kropką.
- ▶ Predykat jest uzgodniony tylko raz (kolejna próba zawodzi).

```
?- read(X).  
|: a1a.
```

```
X = a1a.
```

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Czytanie i pisanie termów

- ▶ Term pisze się na standardowe wyjście predykatem `write/1`.
- ▶ Predykat jest uzgodniony tylko raz (kolejna próba zawodzi).
- ▶ Aby przejść do nowej linii należy użyć predykatu `nl/0`.

```
?- write(f(a, X)), X = b, write(f(a, X)).  
f(a,_G949)f(a,b)  
X = b.
```

```
?- write(f(a, X)), nl, X = b, write(f(a, X)).  
f(a,_G949)  
f(a,b)  
X = b.
```

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Czytanie i pisanie termów

- ▶ Do wydrukowania odstępu długości `N` spacji można użyć predykatu `tab(N)`.

```
pp([H | T], I) :- !, J is I+3, pp(H, J), ppx(T, J).  
pp(X, I) :- tab(I), write(X), nl.
```

```
ppx([], _).  
ppx([H | T], I) :- pp(H, I), ppx(T, I).
```

```
?- pp([1, [2], 3, [4, 5]], 0).  
1  
2  
3  
4  
5  
true.
```

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Czytanie i pisanie termów

- ▶ Term w postaci prefiksowej można wydrukować predykatem `write_canonical/1` (w niektórych implementacjach `display/1`).

```
?- write_canonical(a+b*x+c*x*x).  
+(+(a,*(b,x)),*(*(c,x),x))  
true.
```



Wejście i wyjście

Czytanie i pisanie termów

- ▶ Drukując term predykatem `write/1` należy mieć świadomość, że jeśli będziemy go odczytywać kiedyś predykatem `read/1`, to możemy otrzymać zupełnie inny term.
- ▶ Aby na wydruku postać termu była odpowiednia dla jego ponownego odczytania należy użyć predykatu `writeq/1`.

```
?- write('Atom').  
Atom  
true.
```

```
?- writeq('Atom').  
'Atom'  
true.
```



Wejście i wyjście

Czytanie i pisanie termów

- ▶ Do wydruku sformatowanego należy użyć predykatu `format(FORMAT, LISTA_WARTOŚCI)`.
- ▶ W formacie można użyć specyfikacji formatu rozpoczynających się od znaku tyldy.
- ▶ Wybrane specyfikacje:
 - ▶ `~t` tabulacja
 - ▶ `~n` nowa linia
 - ▶ `~w` drukuje się predykatem `write/1`
 - ▶ `~k` drukuje się predykatem `write_canonical/1`
 - ▶ `~q` drukuje się predykatem `writeq/1`

```
?- format('write: ~w~ncanonical: ~k~nquoted: ~q~n',  
        ['Atom'+2, 'Atom'+2, 'Atom'+2]).  
write: Atom+2  
canonical: +('Atom',2)  
quoted: 'Atom'+2  
true.
```



Wejście i wyjście

Czytanie i pisanie termów

Problem

Napisz w Prologu cel, który wydrukuje swoją dokładną kopię.

```
?- write('write('...)
```



Wejście i wyjście

Czytanie i pisanie termów

Problem (Rozwiązanie)

W poniższym celu użyto predykatu `format/2`.

```
?- X='X=~q,format(X,X).',format(X,X).  
X='X=~q,format(X,X).',format(X,X).
```



Wejście i wyjście

Czytanie i pisanie znaków

- ▶ Do czytania pojedynczego znaku ze standardowego wejścia można użyć predykatu `get_char/1`.
- ▶ Do pisania znaku można użyć predykatu `put_char/1`.
- ▶ Jeśli podczas wykonywania `get_char(X)` plik zakończył się, to pod zmienną `X` zostanie podstawiona stała `end_of_file`.

Poniższy program czyta znaki i zamienia literki a na b:

```
zamieniaj :- get_char(C), dalej(C).
```

```
dalej(end_of_file) :- !.  
dalej(a          ) :- !, put_char(b), zamieniaj.  
dalej(X          ) :- put_char(X), zamieniaj.
```



Wejście i wyjście

Czytanie i pisanie znaków

Example (Zamiana czytanego tekstu na listę słów)

Zaprezentujemy program, który czyta znaki ze standardowego wejścia i wydziela z nich słowa złożone z niebiałych znaków.

Zacznijmy od zdefiniowania jakie znaki uznawane są za białe:

```
biały(' '). % spacja  
biały('\t'). % tabulacja  
biały('\n'). % nowa linia
```



Wejście i wyjście

Czytanie i pisanie znaków

Example (cd.)

Predykat `czytaj(X)` czyta standardowe wejście i zwraca listę słów `X`, będących ciągami niebiałych znaków:

```
czytaj(X) :-  
    get_char(C),  
    czytaj_dalej(C, X).
```

Kluczową rolę odgrywa predykat `czytaj_dalej(C, X)`, którego działanie zależy od ostatnio przeczytanego znaku `C`.



Wejście i wyjście

Czytanie i pisanie znaków

Example (cd.)

Jeśli czytając standardowe wejście osiągnięto koniec strumienia znaków, to zwracana jest lista pusta. W przeciwnym przypadku albo pomija się biały znak albo przystępuje do czytania słowa złożonego z niebiałych znaków:

```
czytaj_dalej(end_of_file, []) :-
    !.
czytaj_dalej(C1, X) :-
    biały(C1),
    !,
    get_char(C2),
    czytaj_dalej(C2, X).
czytaj_dalej(C1, [H | T]) :-
    czytaj_słowo(C1, C2, '', H),
    czytaj_dalej(C2, T).
```

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Czytanie i pisanie znaków

Example (cd.)

Predykat czytaj_słowo(C1, C3, N1, N) czyta słowo N, które składa się prefiksu niebiałych znaków N1, po którym następuje albo niebiały znak C1, albo czytanie słowa kończy się.

Znak C3 jest pierwszym niebiałym znakiem (lub końcem strumienia) po przeczytanim słowie.

```
czytaj_słowo(end_of_file, end_of_file, N, N) :-
    !.
czytaj_słowo(C, C, N, N) :-
    biały(C),
    !.
czytaj_słowo(C1, C3, N1, N) :-
    atom_concat(N1, C1, N2),
    get_char(C2),
    czytaj_słowo(C2, C3, N2, N).
```

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Czytanie i pisanie znaków

Example (cd.)

Przykładowy dialog:

```
?- czytaj(X).
|: Ala ma Asa, As to Ali pies.
|: ^DX = ['Ala', ma, 'Asa,', 'As', to, 'Ali', 'pies.'].

```

Proszę zwrócić uwagę na apostrofy przy słowach zaczynających się wielką literą (to są atomy a nie zmienne). Znaki przestankowe zostały dolepione na końcu słów (bo nie są białymi znakami).

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Zmiana strumienia wyjściowego

- ▶ Bieżący strumień wyjściowy można zmienić na PLIK wywołując predykat `tell(PLIK)`.
- ▶ Aby powrócić do poprzedniego strumienia wyjściowego należy wywołać predykat `told`.

```
?- tell(ala), write(1), tell(ola), write(2), told,
   write(3), told, write(4).
4
true.
```

W pliku `ala` znajdzie się 13 a w pliku `ola` znajdzie się 2.

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Wejście i wyjście

Zmiana strumienia wejściowego

- ▶ Bieżący strumień wejściowy można zmienić na PLIK wywołując predykat `see(PLIK)`.
- ▶ Aby powrócić do poprzedniego strumienia wejściowego należy wywołać predykat `seen`.

```
?- see(ala), get_char(X), see(ola), get_char(Y), seen,  
   get_char(Z), seen.  
X = '1',  
Y = '2',  
Z = '3'.
```



Wejście i wyjście

Otwieranie i zamykanie strumieni

- ▶ Strumień otwiera się predykatem `open(NAZWA_PLIKU, TRYB, ZMIENNA)`.
- ▶ Otwarty strumień zostanie zunifikowany ze zmienną `ZMIENNA`.
- ▶ Możliwe tryby to `read`, `write`, `append` lub `update`.
- ▶ Jeśli `STRUMIEŃ` jest otwartym strumieniem, to zamyka się go predykatem `close(STRUMIEŃ)`.
- ▶ Jednoargumentowe predykaty czytające i piszące mają swoje wersje dwuargumentowe (dodatkowym pierwszym argumentem jest strumień).

```
?- open(ala, read, X), get_char(X, Y), get_char(X, Z),  
   close(X).  
X = <stream>(0x7fbdca477880),  
Y = '1',  
Z = '3'.
```

