Лабораторна робота № 7
Списки в Prolog

Мета роботи: вивчення прийомів обробки списків; отримання і закріплення практичних навичок складання програм мовою Prolog.

Порядок виконання роботи

1. Вивчіть теоретичні відомості.

2. Складіть правила для визначення приналежності елемента:
   а) до голови списку: is_head(integer, integerlist);
   б) до хвоста списку: is_tail(integer, integerlist);
   в) до списку в цілому: is_member(integer, integerlist).

3. Перевірте, який результат роботи програми визначення приналежності елементу до списку (*пункт 2,в*) буде отримано, якщо задати мету у вигляді:
а) is_member(X, [1, 2, 5]) ?
б) is_member(2, [1, 3, N]) ?

4. Складіть правила для друку списку: print(integerlist).

5. Переставте два останні предикати (write(H) і print(T)) у хвості рекурсивного правила для print(integerlist). Дослідіть хід виконання. Що змінилося в роботі програми? Який з варіантів, на Вашу думку, краще?

6. Складіть і відладьте програму **обчислення суми елементів списку** за двома методами:
а) загальної (неоптимальної) рекурсії,
б) хвостової (оптимальної) рекурсії.
Порівняйте механізми роботи програм.

7. Складіть і відладьте програму для видалення елементу із списку:

delete (X, List, Reslist)

де X – *елемент, який підлягає видаленню*;

List – *початковий_список*;

Reslist – *отриманий_список* .

8. Перевірте роботу Вашої програми видалення елемента з цілями наступних типів:

а) delete (4, [2, 4, 6], R)
б) delete (4, [2, 4, 6], [2, 6])
в) delete (9, [2, 4, 6], [2, 6])
г) delete (Z, [2, 4, 6], [2, 6])
д) delete (4, L, [2, 6])

Який сенс мають ці цілі?

9. Складіть і відладьте програми злиття і множинного об'єднання списків. Розберіться в їх роботі.

10. Самостійно придумайте **власне завдання** на обробку списку і розв’яжіть його.

11. Готуємося до МК2. Використовуючи алгоритми, раніше розроблені у 1-й модульній контрольній для Вашої предметної області за індивідуальним варіантом, реалізуйте відповідну програму із зберіганням даних у вигляді списків.

12. Підготуйте та захистіть звіт про виконану роботу.

Приклади контрольних питань для самоперевірки

1. Як інтерпретатор Прологу розпізнає списки у програмі?
2. Яким знаком позначається операція виділення голови списку, хвоста списку?
3. Як позначається порожній список?
4. Що є головою і що є хвостом списку, який містить один елемент – [x]?
5. З яких частин складається будь-яке рекурсивне правило?
6. З яких частин може складатися рекурсивне правило для обробки списків?
7. Коли закінчується послідовність рекурсивних викликів правил в програмах обробки списків?

Теоретичні відомості

Список в Пролозі – це впорядкована множина елементів одного типу (одного домена). Списки прийнято записувати у вигляді послідовності елементів, розділених комами і узятих в квадратні дужки:

[1,3,5], [a, b, e, f], [jam, apple, bread].

Список може містити довільну кількість елементів.

Окремими випадками списків є список, що містить один елемент, – наприклад, [x], і порожній список, в якому не міститься жодного елементу – [].

В ході роботи програми елементи можуть бути додані до списку або видалені з нього. Тобто, кількість елементів списку може змінюватися в процесі роботи програми. Тому списки відносять до динамічних структур даних.

Для списків базовими операціями є:
- виділення голови списку
- виділення хвоста списку
- додавання елементу в голову списку.

Голова списку – це один елемент, що стоїть в списку першим. Хвіст списку – це все інше. Інакше кажучи, хвостом списку називається список, який залишиться, якщо з початкового списку вилучити голову.

При написанні програм необхідно пам'ятати, що голова і хвіст – це об'єкти різних типів: голова списку – це *елемент* (наприклад, integer або symbol), а хвіст списку – це *список* (тобто, динамічна структура даних, утворена з цих елементів).

У мові Prolog опис списку здійснюється в секції domains. Ознакою опису списку є наявність символу * (зірочка) після опису типу елементу.

Приклади опису списків:

```
domains
  list=integer*   % список цілих чисел, наприклад:  [3,5]

  l_char=char*  % список символів, наприклад:  ['a', 'f', 'h', 'd']

  l_col=color*   % список кольорів, наприклад:  [red, green, yellow]
  color=symbol
```

(*якщо останнє не спрацює, то поставте спочатку color=symbol, а потім l_col=color**)

Для запису базових операцій із списками в мові Prolog використовується один символ – вертикальна риска (|):
List=[Head | Tail].

(Англійською мовою Head і Tail – це голова і хвіст.
У програмах частіше пишуть просто L, H, T)

Залежно від ситуації, в якій зустрілася вертикальна риска, процедури зіставлення і уніфікації, що є компонентами інтерпретатора Прологу, самостійно виконують операції розділення списку на голову Head і хвіст Tail або додавання елементу Head до списку Tail .

Оскільки списки є рекурсивними структурами, то для їх обробки зручно використовувати рекурсивні алгоритми.
Рекурсивні алгоритми, як правило, складаються з двох гілок:

1. що робити, якщо даний крок алгоритму останній?
2. що робити, якщо даний крок алгоритму не останній?

У застосуванні до списків ці дві гілки можуть мати, наприклад, такий сенс:

1. що робити з порожнім списком?
2. що робити із списком, у якого є голова і хвіст?

або:

1. що робити, якщо потрібний елемент знайдено в голові списку?

2. що робити, якщо в голові списку потрібний елемент не знайдено?

3. що робити, якщо список вже порожній, а ми все ще шукаємо?

і т.п.