Міністерство освіти і науки України

Національний технічний університет "ХПІ"

кафедра "Інформатика та інтелектуальна власність"

**ЗВІТ**

до лабораторної роботи № 11

Тема : «**Обробка списків у Пролозі** »

Варіант номер 18

з дисципліни "Основи штучного інтелекту"

Виконав:

студент групи КН-321В

Хома Д.М.

Перевірив:

Паржин Ю. В.

Харків 2023

**1 Завдання**

Розробити програму з використанням списків, за індивідуальним завданням додати до програми правила для вирішення задачі. Так як варіантів 16, а я 18. То я взяв 2 варіант.

Завдання:

У списку цілих чисел S1, S2, ..., Sn виключити всі послідовності зазначеного виду, наприклад [1, 11, 24, 5].

**2. Код програми**

Код програми наведено на рис 2.1.

% Власна реалізація конкатенації списків

my\_append([], L, L). % конкатенація порожнього списку з будь-яким списком дає той самий список

my\_append([H|T], L, [H|R]) :- my\_append(T, L, R). % Рекурсивно додає голову до результату конкатенації залишку списку та другого списку

% Власна реалізація порівняння двох елементів

my\_dif(X, Y) :- X \= Y. % Визначає, чи два елементи неоднакові

% Предикат для видалення послідовності зі списку

remove\_sequence([], \_, []). %видалення з порожнього списку дає порожній список

% Якщо поточна частина списку починається з заданої послідовності, пропустити її

remove\_sequence(List, Sequence, Result) :-

my\_append(Sequence, Rest, List), % Застосовує функцію конкатенації списків для отримання залишку

remove\_sequence(Rest, Sequence, Result). % Рекурсивно викликає саму себе для обробки залишку

% Якщо поточна частина списку не починається з заданої послідовності, додати голову до результату і продовжити рекурсію

remove\_sequence([Head|Tail], Sequence, [Head|Result]) :-

my\_dif(Head, Sequence), % Використовуємо функцію порівняння елементів для визначення, чи голова не співпадає з послідовністю

remove\_sequence(Tail, Sequence, Result). % Рекурсивний виклик для обробки залишку та формування результату

Рисунок 2.1 – Код програми

my\_append - Цей предикат визначає, як конкатенувати два списки. Базовий випадок визначає, що конкатенація порожнього списку з будь-яким іншим списком дає той самий список. Рекурсивне правило додає Head першого списку до результату конкатенації залишку першого списку та другого списку.

my\_dif - Цей предикат визначає, чи два елементи неоднакові, використовуючи операцію \=.

remove\_sequence - Цей предикат визначає, як видаляти послідовність зі списку. Базовий випадок визначає, що видалення з порожнього списку дає порожній список. Рекурсивне правило використовує my\_append для отримання залишку, якщо поточна частина списку починається з заданої послідовності. Якщо це не так, то додає Head до результату і продовжує рекурсію для обробки залишку. Видалена послідовність не входить до результату.

**3. Питання з відповідями**

Нижче на рис 3.1 наведене питання де перший параметр це список цілих чисел, а другий параметр – це послідовність, яку потрібно видалити. Повинно залишитись [9,9,9].

remove\_sequence([1, 11, 24, 5, 9, 1, 11, 24, 5, 9, 9], [1, 11, 24, 5], Result)

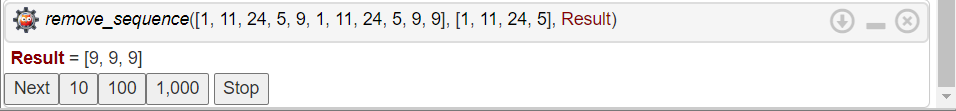


Рисунок 3.1 – Питання з відповіддю

Перевіримо на рис. 3.2 якщо список складається тільки із елементів послідовності, яку потрібно видалити

remove\_sequence([1, 11, 24, 5], [1, 11, 24, 5], Result)

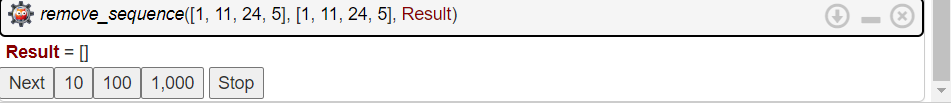


Рисунок 3.2 – Питання з відповіддю

На рис. 3.3 перевіримо якщо перший список пустий.

remove\_sequence([], [1, 11, 24, 5], Result)



Рисунок 3.3 – Питання з відповіддю

На рис 3.4 перевіримо на списку в якому немає елементів з послідовності.

remove\_sequence([1,2,3,4,5], [1, 11, 24, 5], Result)

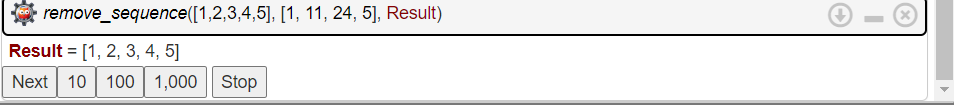


Рисунок 3.4 – Питання з відповіддю

**4. Висновки**

У даному звіті було розглянуто програму на мові Prolog, яка реалізує видалення зі списку усіх заданих послідовностей. Були створені власні реалізації операцій конкатенації та порівняння. При використанні цих предикатів слід враховувати можливість неефективності при обробці великих списків через рекурсію Prolog. Код з коментарями та результати програми наведені в звіті.