**Лабораторна робота № 3**

**Перевірка типів термів. Створення та декомпозиція термів засобами мови Prolog.**

**Варіант № 1**

1. Написати предикат, який видаляє зі списку усі атоми, крім останнього.
2. Написати предикат contains\_only\_vars(Term) таким чином, щоб він приймав істинне зна­чен­ня, якщо серед аргументів параметра Term (та його підтермів) не зустрічаються константи (атоми або числа).

**Варіант № 2**

1. Написати предикат для обчислення кількості елементів списку, які є структурами і ма­ють не менше трьох компонент.
2. Написати предикат is\_concrete(Term) таким чином, щоб він приймав істинне зна­чен­ня, якщо Term (та його підтерми) не містять неконкретизованих змінних.

**Варіант № 3**

1. Написати предикат, який переставляє усі вільні змінні, які зустрічаються у списку, на його початок.
2. Написати предикат sum\_of\_numbers(Term, Sum) таким чином, щоб він давав змогу зна­хо­дити суму усіх числових компонентів терма-структури (з урахуванням його підтермів).

**Варіант № 4**

1. Написати предикат, який виконує заміну усіх числових елементів списку на першу по по­рядку вільну змінну серед елементів списку.
2. Написати предикат occurrence\_count(Term, Atom, Count) таким чином, щоб він давав змогу зна­хо­дити кількість входжень заданого атому у термі (з урахуванням входжень у підтерми).

**Варіант № 5**

1. Написати предикат, який видаляє зі списку усі вільні змінні та атоми, які входять у цей список тільки один раз.
2. Написати предикат number\_count(Term, Count) таким чином, щоб він давав змогу зна­хо­дити кількість різних чисел у термі (з урахуванням входжень у якості аргументів терма та його підтермів).

**Варіант № 6**

1. Написати предикат, який заміняє у списку усі елементи-структури, які мають рівно одну ком­поненту, на цю компоненту.
2. Написати предикат get\_depth(Term, Depth) таким чином, щоб він дозволяв знайти глибину терму (Вважати, що глибина константи рівна 1, глибина структури рівна 1 + максимальна глибина компонент, глибина змінної невизначена).

**Варіант № 7**

1. Написати предикат, який виконує перевірку того, чи є його аргумент структурою, перша та останні компоненти якої — числа або атоми.
2. Написати предикат replace(Value1, Term1, Value2, Term2), який конструює терм Term2, замінюючи у термі Term1 (та усіх його підтермах) усі входження пара­метра Value1 на Value2.

**Варіант № 8**

1. Написати предикат для знаходження найменшої кількості аргументів, які мають ті еле­менти списку, які є структурами.
2. Написати предикат get\_frequentest(Term, Functors) таким чином, щоб він до­зволяв знаходити список функторів, які найчастіше зустрічаються у складі заданого тер­му (серед аргументів терму та аргументів його підтермів).

**Варіант № 9**

1. Написати предикат, який переставляє усі вільні змінні, які зустрічаються у списку, у його кінець.
2. Написати предикат contains(Term, Subterm) таким чином, щоб він дозволяв пе­ревіряти, чи входить підтерм (2-й аргумент) до складу заданого терму.

**Варіант № 10**

1. Написати предикат для знаходження типу даних, який має найбільша кількість елементів списку (атом, число, структура чи вільна змінна).
2. Написати предикат simple\_terms(Term, Subterms) таким чином, щоб він дозво­ляв знаходити список *простих підтермів* (не структурованих термів, які не мають влас­них підтермів, тобто констант або змінних) заданого терму (серед аргументів терму та ар­гу­­ментів його підтермів).

**Варіант № 11**

1. Написати предикат для перевірки того, чи є аргументи заданого терму «центрально си­мет­рич­ними» (перший ідентичний останньому, другий — передостанньому і т. д.)
2. Написати предикат deepest(TermList, ResultList) для знаходження списку термів найбільшої глибини серед термів, які входять до складу списку TermList (Вважати, що глибина константи та змінної рівна 1, глибина структури рівна 1 + мак­си­мум із глибин її компонент).

**Варіант № 12**

1. Написати предикат для знаходження підсписку назв (функторів) усіх складених термів за­даної арності, які є елементами заданого списку.
2. Написати предикат list\_free(Term) таким чином, щоб він приймав істинне зна­чен­ня, якщо Term (та його підтерми) не містять аргументів, які є списками.

**Варіант № 13**

1. Написати предикат для знаходження підсписку, який містить ті елементи списку, які є струк­турами та мають максимальну кількість аргументів.
2. Написати предикат remove\_vars(Term, Result) таким чином, щоб він дозволяв видаляти усі вільні змінні, які входять до складу терма-структури (з урахуванням входжень змінних у його підтерми).

**Варіант № 14**

1. Написати предикат, який виконує перевірку того, чи є його аргумент структурою, ос­тан­ні дві компоненти якої — атоми.
2. Написати предикат count\_of\_variables(Term, Count) таким чином, щоб він давав змогу зна­хо­дити кількість *різних* змінних, які входять до складу терма-структури (з урахуванням входжень змінних у його підтерми).

**Варіант № 15**

1. Написати предикат для знаходження тієї кількості аргументів, яку має найбільше число елементів списку (вважати, що для констант та змінних кількість аргументів рівна 0). Наприклад, для списку [f(1), X, g(2), 3, a] три елементи мають 0 аргументів, а два — 1 аргумент. Тому результатом є число 3.
2. Написати предикат get\_atoms(Term, Result) для знаходження списку усіх атомів, які входять до складу заданого терму (елементи у списку Result не мають повто­рю­ватися).