Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 4

з дисципліни

«Мультипарадигменне програмування»

ВИКОНАВ:
студент III курсу ФІОТ
групи ІО-23
Бичкар Н. В.
Залікова № 2302

ПЕРЕВІРИВ:
ас. Очеретяний О. К.

Київ – 2025

Лістинг:

```
% === Факти ===
% Вхідний числовий ряд (можна змінювати)
series([9.84, 1.71, 6.07, 3.43, 8.62, 2.17, 0.96, 4.11, 0.25, 6.93,
 7.74, 2.45, 1.36, 8.11, 5.29, 7.05, 0.83, 4.57, 3.34, 1.19,
 9.12, 4.89, 6.78, 2.03, 7.43, 9.36, 3.86, 1.90, 0.41, 5.67,
 2.95, 6.34, 9.70, 1.08, 8.54, 7.30, 0.66, 5.04, 3.69, 2.69,
 4.21, 6.49, 7.98, 0.03, 1.56, 9.95, 2.82, 5.45, 3.25, 8.29,
 4.63, 1.28, 6.17, 2.58, 7.89, 5.91, 0.13, 4.76, 9.60, 3.50,
 8.81, 2.40, 1.65, 6.02, 5.35, 0.38, 3.73, 7.18, 4.97, 9.03,
 6.87, 1.49, 8.95, 2.21, 7.12, 5.58, 3.00, 0.79, 4.34, 6.22,
 9.49, 1.99, 8.33, 3.94, 2.64, 0.54, 5.76, 7.38, 6.73, 1.14,
 4.42, 9.26, 0.10, 3.17, 8.68, 5.10, 2.33, 6.60, 7.60, 0.47]).

% Заданий алфавіт (можна змінювати за змістом і кількістю)
alphabet([a, b, c, d]).

% Параметр лямбда для пуассонівського розподілу
lambda(2).

% === Правила ===
% Знаходження мінімального елемента списку
my_min([X], X).
my_min([H|T], Min) :- my_min(T, Temp), Min is min(H, Temp).

% Знаходження максимального елемента списку
my_max([X], X).
my_max([H|T], Max) :- my_max(T, Temp), Max is max(H, Temp).

% Факторіал для обчислення пуассонівської функції розподілу
factorial(0, 1) :- !.
factorial(N, Result) :-
    N > 0,
    N1 is N - 1,
    factorial(N1, Result1),
    Result is Result1 * N.

% Обчислення ймовірності за пуассонівським розподілом P(X = k) = (lambda^k *
e^(-lambda)) / k!
poisson_pmf(K, Lambda, Prob) :-
    Exp is exp(-Lambda),
    LambdaPowK is Lambda ** K,
    factorial(K, Fact),
    Prob is (LambdaPowK * Exp) / Fact.

% Обчислення функції розподілу Пуассона F(x) = Σ(i=0 до x) P(X = i)
poisson_cdf(X, Lambda, CDF) :-
    X < 0, CDF is 0, !.
```

```
poisson_cdf(X, Lambda, CDF) :-
    X >= 0,
    Floor is floor(X),
    poisson_cdf_sum(0, Floor, Lambda, CDF).

poisson_cdf_sum(K, Max, _, 0) :- K > Max, !.
poisson_cdf_sum(K, Max, Lambda, Sum) :-
    K =< Max,
    poisson_pmf(K, Lambda, Prob),
    K1 is K + 1,
    poisson_cdf_sum(K1, Max, Lambda, RestSum),
    Sum is Prob + RestSum.

% Довжина списку (замість вбудованої length/2)
my_length([], 0).
my_length([_|T], L) :-
    my_length(T, L1),
    L is L1 + 1.

% between/3 (заміна для вбудованого предиката в SWI-Prolog)
my_between(Low, High, Low) :- Low =< High.
my_between(Low, High, Value) :-
    Low < High,
    NextLow is Low + 1,
    my_between(NextLow, High, Value).

% Генерація списку ймовірностей для Пуассонівського розподілу
generate_poisson_probs(MaxK, Lambda, Probs) :-
    generate_poisson_probs_helper(0, MaxK, Lambda, [], Probs).

generate_poisson_probs_helper(K, MaxK, _, Acc, Probs) :-
    K > MaxK,
    my_reverse(Acc, Probs), !.
generate_poisson_probs_helper(K, MaxK, Lambda, Acc, Probs) :-
    K =< MaxK,
    poisson_pmf(K, Lambda, Prob),
    K1 is K + 1,
    generate_poisson_probs_helper(K1, MaxK, Lambda, [Prob|Acc], Probs).

% Власна реалізація реверсу списку (щоб не конфліктувати з вбудованим reverse/2)
my_reverse(List, Reversed) :-
    my_reverse_helper(List, [], Reversed).

my_reverse_helper([], Acc, Acc).
my_reverse_helper([H|T], Acc, Reversed) :-
    my_reverse_helper(T, [H|Acc], Reversed).

% Сума перших N елементів списку (Використовуємо my_sum_list замість sum_list)
```

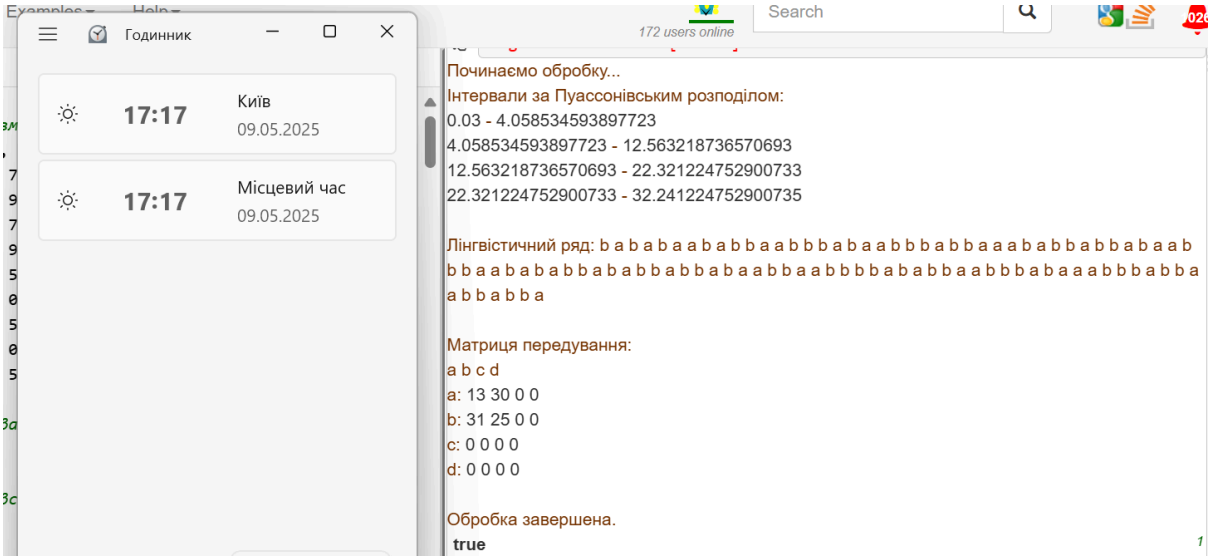
```
sum_first_n(List, N, Sum) :-
    take_first_n(List, N, FirstN),
    my_sum_list(FirstN, Sum).

% Власна реалізація суми списку
my_sum_list([], 0).
my_sum_list([H|T], Sum) :-
    my_sum_list(T, Rest),
    Sum is H + Rest.

% Взяти перші N елементів списку
take_first_n(List, N, Result) :-
    take_first_n_helper(List, N, [], RevResult),
    my_reverse(RevResult, Result).

take_first_n_helper(_, 0, Acc, Acc) :- !.
take_first_n_helper([], _, Acc, Acc) :- !.
take_first_n_helper([H|T], N, Acc, Result) :-
    N > 0,
    N1 is N - 1,
    take_first_n_helper(T, N1, [H|Acc], Result).

% Побудова інтервалів за пуассонівським розподілом
build_poisson_intervals(Min, Max, N, Lambda, Intervals) :-
    MaxK is N * 2, % Збільшуємо верхню межу для отримання більшої точності
    generate_poisson_probs(MaxK, Lambda, Probs),
    my_sum_list(Probs, TotalProb),

    % Обчислюємо межі інтервалів залежно від ймовірностей
    Range is Max - Min,
    build_poisson_intervals_helper(0, Min, Range, Probs, TotalProb, N, [], IntervalsRev),
    my_reverse(IntervalsRev, Intervals).

build_poisson_intervals_helper(N, _, _, _, _, N, Acc, Acc) :- !.
build_poisson_intervals_helper(I, CurrentMin, Range, Probs, TotalProb, N, Acc, Result) :-
    I < N,
    sum_probs_till_index(Probs, I, N, TotalProb, CumulativeProb),
    NextMin is CurrentMin + (Range * CumulativeProb),
    I1 is I + 1,
    build_poisson_intervals_helper(I1, NextMin, Range, Probs, TotalProb, N, [[CurrentMin,
NextMin]|Acc], Result).

% Підсумовування ймовірностей до певного індексу відносно загальної кількості
інтервалів
sum_probs_till_index(Probs, I, N, TotalProb, RelativeProb) :-
    my_length(Probs, L),
    Fraction is (I+1) / N,
    TargetIdx is floor(Fraction * L),
```

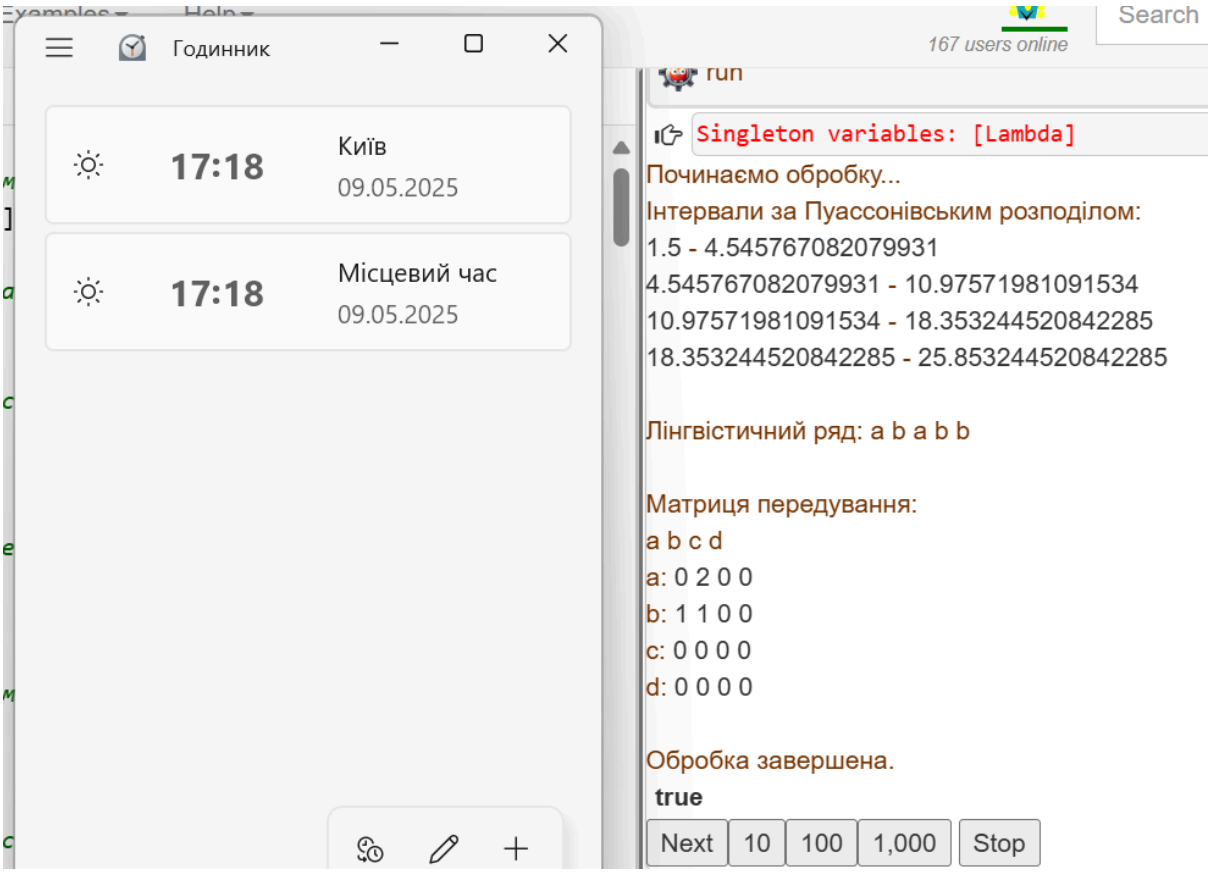
```
    sum_first_n(Probs, TargetIdx, Sum),
    RelativeProb is Sum / TotalProb.

% Пошук інтервалу, до якого належить значення
value_interval(Value, [[A,B]|_], 0) :- Value >= A, Value < B, !.
value_interval(Value, [_|T], Index) :-
    value_interval(Value, T, Temp),
    Index is Temp + 1.
value_interval(Value, [[A,_]], 0) :- Value >= A, !. % Для крайнього правого значення

% Отримання N-того елемента списку (замість nth0/3)
my_nth0(0, [X|_], X) :- !.
my_nth0(N, [_|T], X) :-
    N > 0,
    N1 is N - 1,
    my_nth0(N1, T, X).

% Відображення значення у символ алфавіту
value_to_symbol(Value, Intervals, Alphabet, Symbol) :-
    value_interval(Value, Intervals, Index),
    my_length(Alphabet, L),
    (Index >= L -> LastIndex is L - 1 ; LastIndex is Index),
    my_nth0(LastIndex, Alphabet, Symbol).

% Перетворення всього числового ряду в лінгвістичний
map_series([], _, _, []).
map_series([H|T], Intervals, Alphabet, [S|Rest]) :-
    value_to_symbol(H, Intervals, Alphabet, S),
    map_series(T, Intervals, Alphabet, Rest).

% === Побудова матриці передування ===
% Формує список переходів (a->b, b->c, ...)
transitions([], []).
transitions([_], []).
transitions([A,B|T], [(A,B)|Rest]) :-
    transitions([B|T], Rest).

% Підрахунок кількості конкретного переходу в списку
count_transitions([], _, 0).
count_transitions([(A,B)|T], (A,B), N) :-
    count_transitions(T, (A,B), N1),
    N is N1 + 1.
count_transitions([(X,Y)|T], (A,B), N) :-
    (X \= A ; Y \= B),
    count_transitions(T, (A,B), N).

% Побудова одного рядка матриці передування
build_matrix_row(_, [], _, []).
```

```
build_matrix_row(From, [To|T], Transitions, [Count|Rest]) :-
    count_transitions(Transitions, (From, To), Count),
    build_matrix_row(From, T, Transitions, Rest).

% Побудова повної матриці передування
build_transition_matrix(_, [], _, []).
build_transition_matrix(Alphabet, [From|RestFrom], Transitions, [[From|Row]|MatrixRest]) :-
    build_matrix_row(From, Alphabet, Transitions, Row),
    build_transition_matrix(Alphabet, RestFrom, Transitions, MatrixRest).

% Вивід елемента
write_element(X) :- write(X).

% Вивід списку
write_list([]).
write_list([H|T]) :-
    write_element(H),
    write(' '),
    write_list(T).

% Вивід інтервалів для перевірки
print_intervals([]).
print_intervals([[A, B]|Rest]) :-
    write(A), write(' - '), write(B), nl,
    print_intervals(Rest).

% Вивід матриці
print_matrix([]).
print_matrix([[Row|Rest1]|Rest2]) :-
    write(Row), write(': '),
    write_list(Rest1), nl,
    print_matrix(Rest2).

% Друк заголовку матриці
print_matrix_header(Alphabet) :-
    write('  '),
    write_list(Alphabet),
    nl.

% === Головна функція запуску ===
run :-
    write('Починаємо обробку...'), nl,
    series(Series),
    alphabet(Alphabet),
    lambda(Lambda),

    % Знаходимо мінімум та максимум ряду
    my_min(Series, Min),
```

```
my_max(Series, Max),

% Кількість інтервалів дорівнює потужності алфавіту
my_length(Alphabet, N),

% Побудова інтервалів за Пуассонівським розподілом
build_poisson_intervals(Min, Max, N, Lambda, Intervals),

% Виведення інтервалів для перевірки
write('Інтервали за Пуассонівським розподілом:'), nl,
print_intervals(Intervals), nl,

% Перетворення чисельного ряду на лінгвістичний
map_series(Series, Intervals, Alphabet, Linguistic),
write('Лінгвістичний ряд: '),
write_list(Linguistic), nl,

% Побудова та виведення матриці передування
transitions(Linguistic, Transitions),
build_transition_matrix(Alphabet, Alphabet, Transitions, Matrix),
nl, write('Матриця передування:'), nl,
print_matrix_header(Alphabet),
print_matrix(Matrix),

% Завершення
nl, write('Обробка завершена.'), nl.
```

Результати:




Examples
Help
Search
167 users online
Годинник
17:18
Київ
09.05.2025
17:18
Місцевий час
09.05.2025
run
Singleton variables: [Lambda]
Починаємо обробку...
Інтервали за Пуассонівським розподілом:
1.5 - 4.545767082079931
4.545767082079931 - 10.97571981091534
10.97571981091534 - 18.353244520842285
18.353244520842285 - 25.853244520842285
Лінгвістичний ряд: a b a b b
Матриця передування:
a b c d
a: 0 2 0 0
b: 1 1 0 0
c: 0 0 0 0
d: 0 0 0 0
Обробка завершена.
true
Next
10
100
1,000
Stop