Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

**Лабораторна робота №4**

з дисципліни:

«Мультипарадигменне програмування»

Виконав:

студент групи ІС-23

Шимків Мирослав

Київ 2025

**ЛАБОРАТОРНАЯ РОБОТА №4**

**Завдання**: за допомогою логічного програмування реалізувати перетворення чисельного ряду до лінгвістичного ланцюжка за певним розподілом ймовірностей потрапляння значень до інтервалів.

**Вхідні данні**: чисельний ряд, вид розподілу ймовірностей, потужність алфавіту.

**Вихідні дані**: лінгвістичний ряд.

**Мова програмування**: Prolog і його клони.

**Варіант 24**

(завдання 4 відповідно)

Мій варіант 4, отже, я обрав Хі-розподіл і мову програмування Prolog

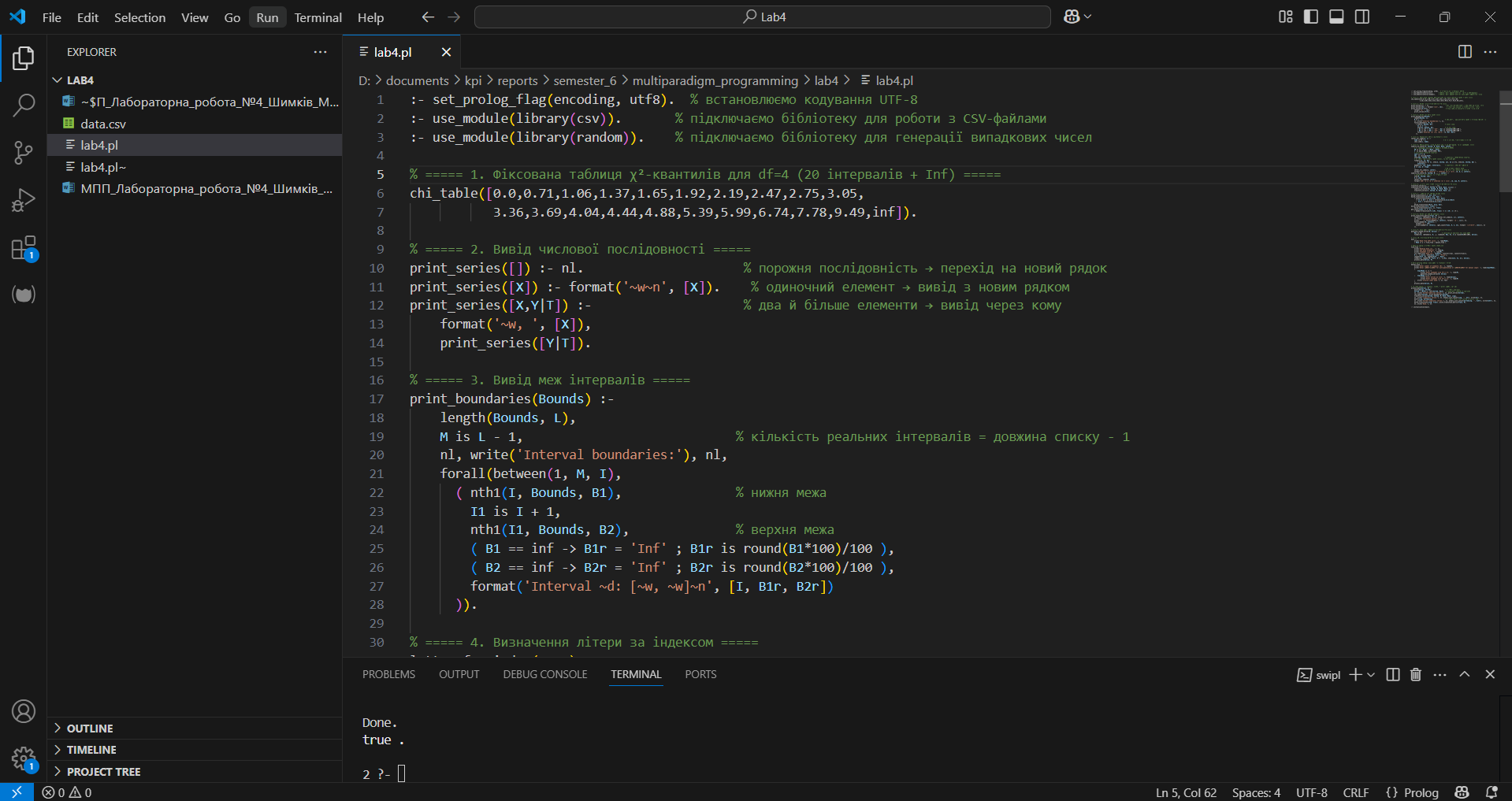
**Виконання роботи**

**Нижче наведено логічні блоки програми:**

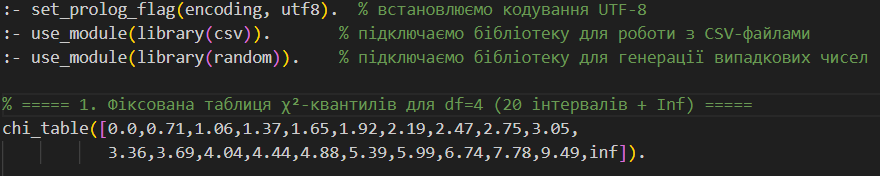
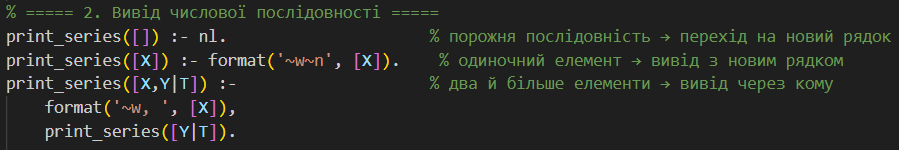
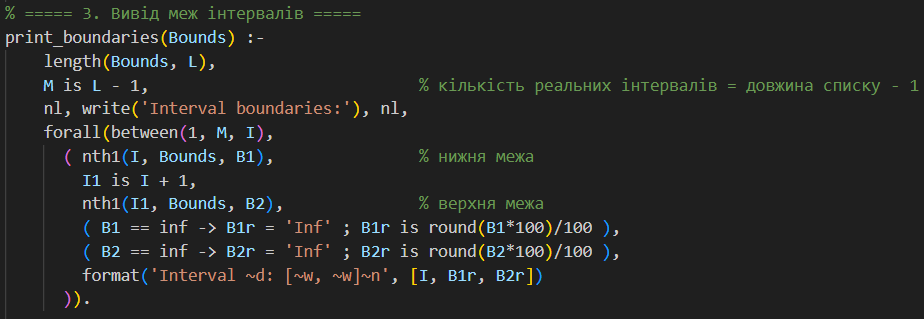
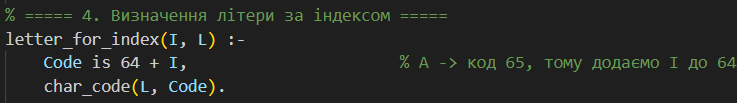
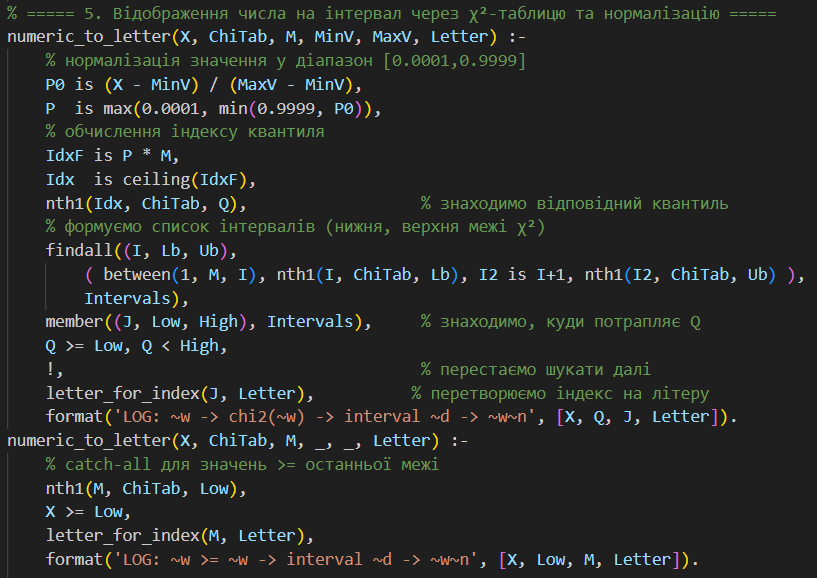
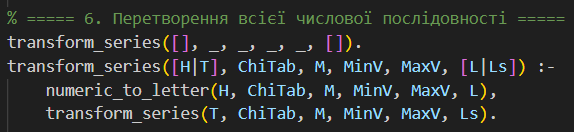
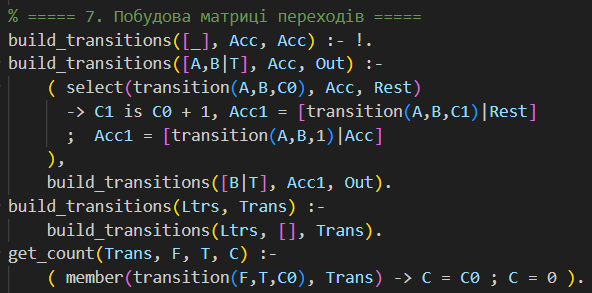
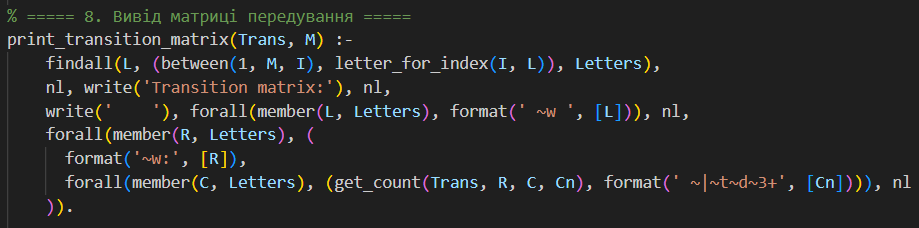
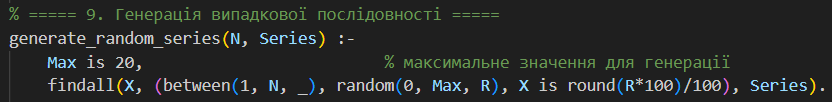
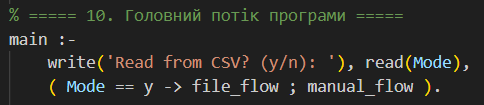
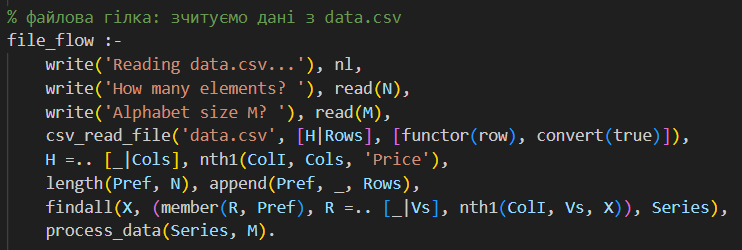
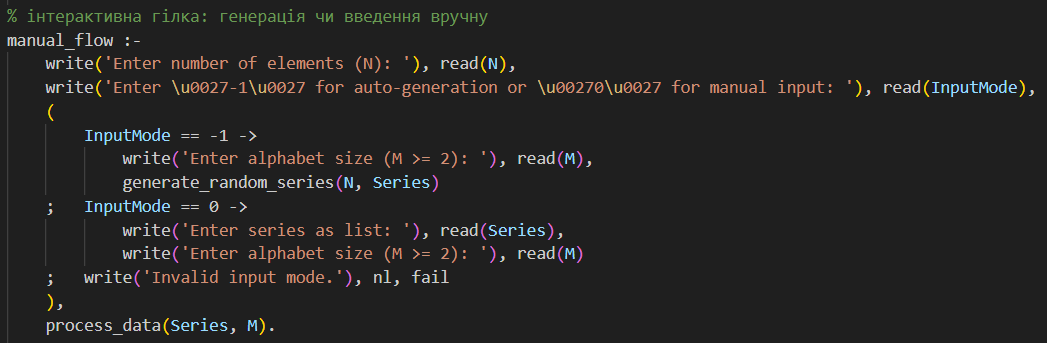
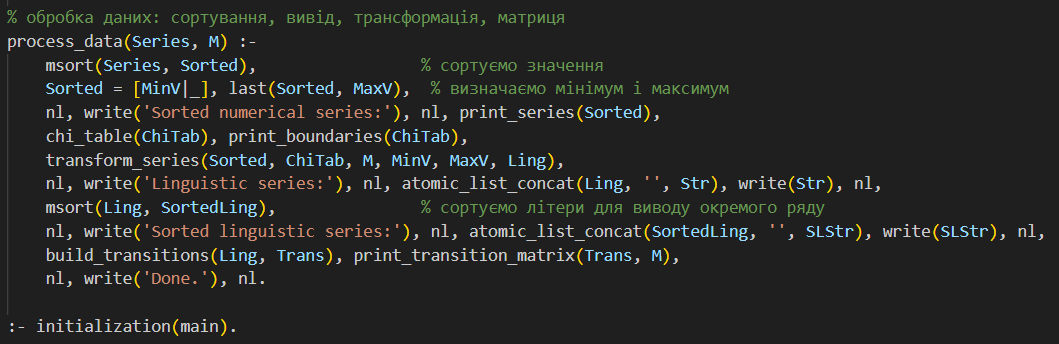
1. Фіксована таблиця χ²-квантилів для df=4 (20 інтервалів + Inf)
2. Вивід числової послідовності
3. Вивід меж інтервалів
4. Визначення літери за індексом
5. Відображення числа на інтервал через χ²-таблицю та нормалізацію
6. Перетворення всієї числової послідовності
7. Побудова матриці переходів
8. Вивід матриці передування
9. Генерація випадкової послідовності
10. Головний потік програми
    1. файлова гілка: зчитуємо дані з data.csv
    2. інтерактивна гілка: генерація чи введення вручну
    3. обробка даних: сортування, вивід, трансформація, матриця

**Давайте розглянемо код для кожного пункту, в якому наведено коментарі українською мовою для кращого розуміння:**

Інтерфейс Visual Studio Code for Prolog для запуску програми

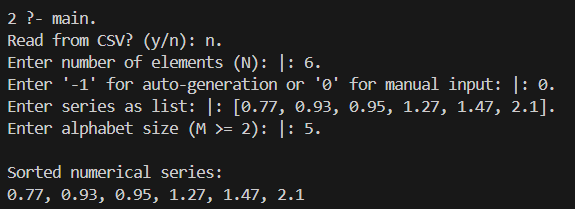


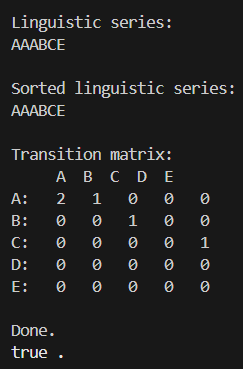
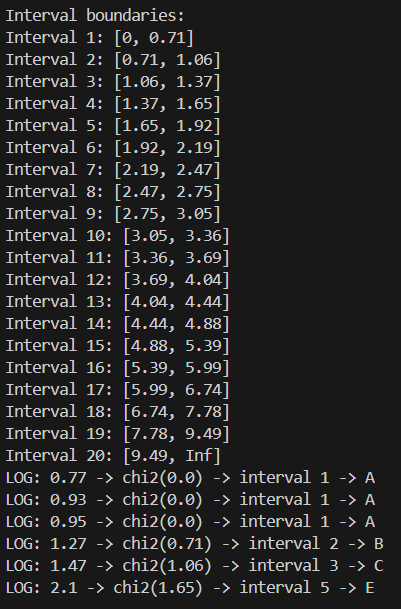
Етапи:

1. 
2. 
3. 
4. 
5. 
6. 
7. 
8. 
9. 
10. 
    1. 
    2. 
    3. 

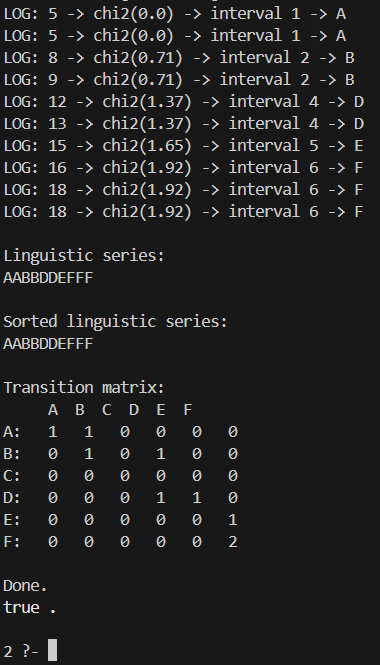
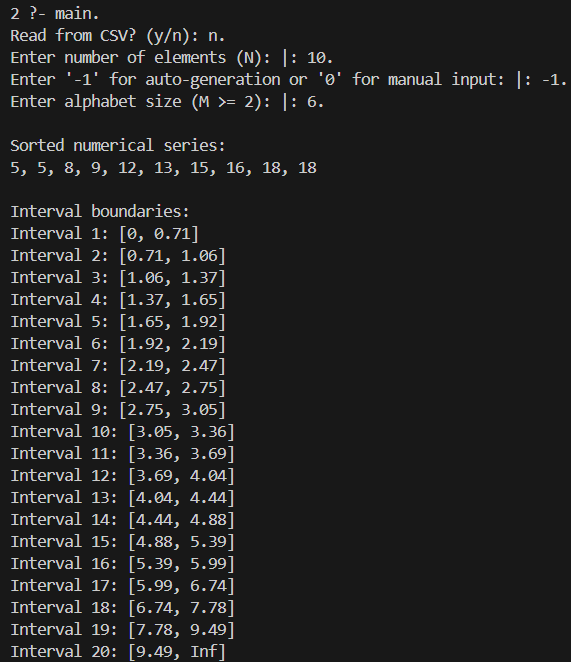
**Вхідні данні і результат виконання програми:**

Введення значень від руки:

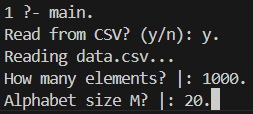


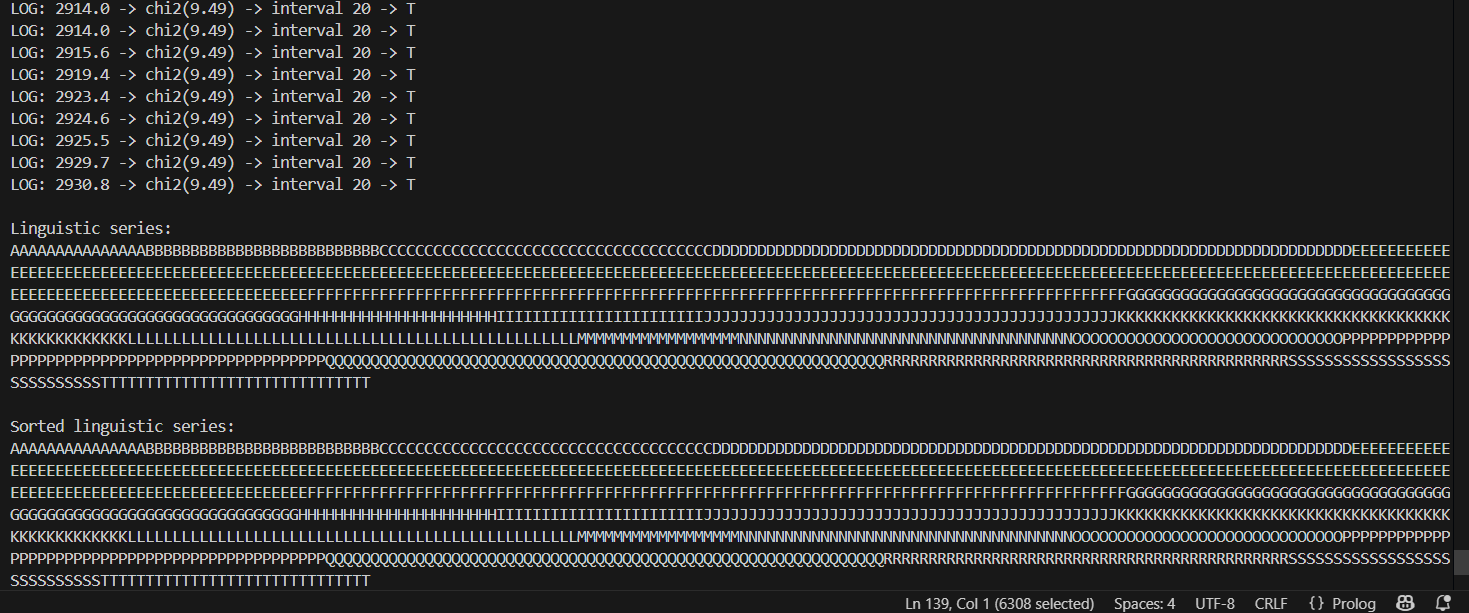


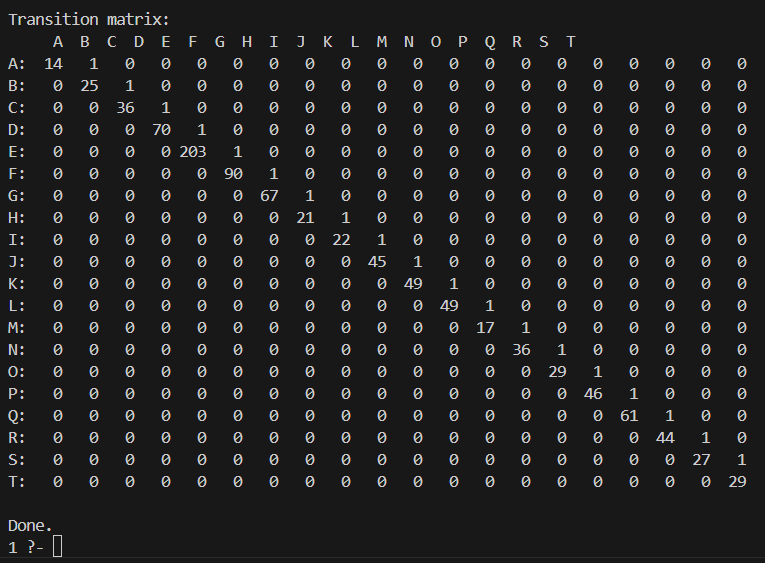
Генерація випадкових значень:



Читання з файлу часового ряду “B-C-D-E-S&P 500 Historical Data” з архіву, а саме поля Price:



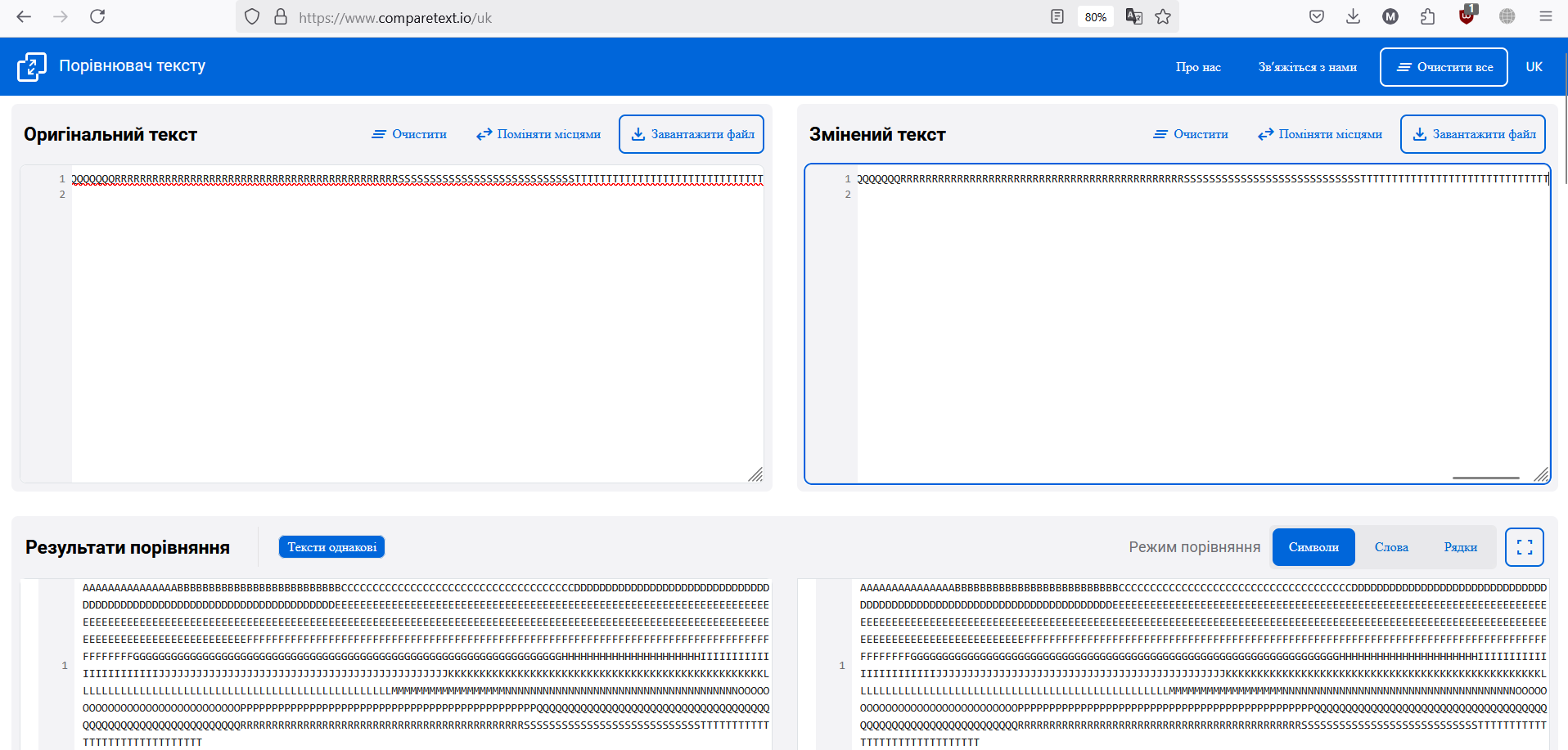




Поточний лінгвістичний ряд:

AAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGHHHHHHHHHHHHHHHHHHHHHHIIIIIIIIIIIIIIIIIIIIIIIJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKKLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLMMMMMMMMMMMMMMMMMMNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRSSSSSSSSSSSSSSSSSSSSSSSSSSSSTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT

Порівняння лінгвістичних рядів лабораторної роботи 3 лабораторної роботи 4:

****

**Лінгвістичні ряди ідентичні.**

**Висновок**

У рамках лабораторної роботи №4 реалізовано програму на мові Prolog, яка:

* Зчитує з консолі кількість елементів числового ряду (N > 0), розмір алфавіту (M ≥ 2) та сам ряд (ручний ввід або автогенерація χ²-послідовності з df = 4).
* Сортує вхідний числовий ряд у порядку зростання та обчислює мінімальне й максимальне значення.
* Обчислює M + 1 меж χ²-квантілей між Min і Max за допомогою оберненої кумулятивної функції та бінарного пошуку.
* Перетворює відсортовані числа у лінгвістичний рядок, зіставляючи кожне значення з відповідною літерою алфавіту за індексом інтервалу.
* Будує матрицю передування — підрахунок частот переходів між кожною парою сусідніх символів.
* Виводить у консоль відсортований ряд, межі інтервалів, лінгвістичний рядок та матрицю передування із цілими значеннями частот.

Результати тестів із ручним введенням і автогенерацією підтверджують коректність алгоритму:

* Відсортовані значення відповідають очікуваним порядкам.
* Межі інтервалів обчислені через рівномірний розподіл ймовірності χ² і гарантують приблизно рівномірну місткість інтервалів.
* Лінгвістичний рядок точно відображає відповідність чисел обраним літерам.
* Матриця передування правильно фіксує кількість появ кожної пари символів.

Таким чином, завдання ЛР №4 виконане повністю й відповідає вимогам лабораторної роботи з логічного програмування.

**Додатки**

Код програми:

:- set\_prolog\_flag(encoding, utf8).  % встановлюємо кодування UTF-8

:- use\_module(library(csv)).       % підключаємо бібліотеку для роботи з CSV-файлами

:- use\_module(library(random)).    % підключаємо бібліотеку для генерації випадкових чисел

% ===== 1. Фіксована таблиця χ²-квантилів для df=4 (20 інтервалів + Inf) =====

chi\_table([0.0,0.71,1.06,1.37,1.65,1.92,2.19,2.47,2.75,3.05,

           3.36,3.69,4.04,4.44,4.88,5.39,5.99,6.74,7.78,9.49,inf]).

% ===== 2. Вивід числової послідовності =====

print\_series([]) :- nl.                     % порожня послідовність → перехід на новий рядок

print\_series([X]) :- format('~w~n', [X]).    % одиночний елемент → вивід з новим рядком

print\_series([X,Y|T]) :-                    % два й більше елементи → вивід через кому

    format('~w, ', [X]),

    print\_series([Y|T]).

% ===== 3. Вивід меж інтервалів =====

print\_boundaries(Bounds) :-

    length(Bounds, L),

    M is L - 1,                            % кількість реальних інтервалів = довжина списку - 1

    nl, write('Interval boundaries:'), nl,

    forall(between(1, M, I),

      ( nth1(I, Bounds, B1),               % нижня межа

        I1 is I + 1,

        nth1(I1, Bounds, B2),              % верхня межа

        ( B1 == inf -> B1r = 'Inf' ; B1r is round(B1\*100)/100 ),

        ( B2 == inf -> B2r = 'Inf' ; B2r is round(B2\*100)/100 ),

        format('Interval ~d: [~w, ~w]~n', [I, B1r, B2r])

      )).

% ===== 4. Визначення літери за індексом =====

letter\_for\_index(I, L) :-

    Code is 64 + I,                      % A -> код 65, тому додаємо I до 64

    char\_code(L, Code).

% ===== 5. Відображення числа на інтервал через χ²-таблицю та нормалізацію =====

numeric\_to\_letter(X, ChiTab, M, MinV, MaxV, Letter) :-

    % нормалізація значення у діапазон [0.0001,0.9999]

    P0 is (X - MinV) / (MaxV - MinV),

    P  is max(0.0001, min(0.9999, P0)),

    % обчислення індексу квантиля

    IdxF is P \* M,

    Idx  is ceiling(IdxF),

    nth1(Idx, ChiTab, Q),                  % знаходимо відповідний квантиль

    % формуємо список інтервалів (нижня, верхня межі χ²)

    findall((I, Lb, Ub),

        ( between(1, M, I), nth1(I, ChiTab, Lb), I2 is I+1, nth1(I2, ChiTab, Ub) ),

        Intervals),

    member((J, Low, High), Intervals),     % знаходимо, куди потрапляє Q

    Q >= Low, Q < High,

    !,                                     % перестаємо шукати далі

    letter\_for\_index(J, Letter),          % перетворюємо індекс на літеру

    format('LOG: ~w -> chi2(~w) -> interval ~d -> ~w~n', [X, Q, J, Letter]).

numeric\_to\_letter(X, ChiTab, M, \_, \_, Letter) :-

    % catch-all для значень >= останньої межі

    nth1(M, ChiTab, Low),

    X >= Low,

    letter\_for\_index(M, Letter),

    format('LOG: ~w >= ~w -> interval ~d -> ~w~n', [X, Low, M, Letter]).

% ===== 6. Перетворення всієї числової послідовності =====

transform\_series([], \_, \_, \_, \_, []).

transform\_series([H|T], ChiTab, M, MinV, MaxV, [L|Ls]) :-

    numeric\_to\_letter(H, ChiTab, M, MinV, MaxV, L),

    transform\_series(T, ChiTab, M, MinV, MaxV, Ls).

% ===== 7. Побудова матриці переходів =====

build\_transitions([\_], Acc, Acc) :- !.

build\_transitions([A,B|T], Acc, Out) :-

    ( select(transition(A,B,C0), Acc, Rest)

      -> C1 is C0 + 1, Acc1 = [transition(A,B,C1)|Rest]

      ;  Acc1 = [transition(A,B,1)|Acc]

    ),

    build\_transitions([B|T], Acc1, Out).

build\_transitions(Ltrs, Trans) :-

    build\_transitions(Ltrs, [], Trans).

get\_count(Trans, F, T, C) :-

    ( member(transition(F,T,C0), Trans) -> C = C0 ; C = 0 ).

% ===== 8. Вивід матриці передування =====

print\_transition\_matrix(Trans, M) :-

    findall(L, (between(1, M, I), letter\_for\_index(I, L)), Letters),

    nl, write('Transition matrix:'), nl,

    write('    '), forall(member(L, Letters), format(' ~w ', [L])), nl,

    forall(member(R, Letters), (

      format('~w:', [R]),

      forall(member(C, Letters), (get\_count(Trans, R, C, Cn), format(' ~|~t~d~3+', [Cn]))), nl

    )).

% ===== 9. Генерація випадкової послідовності =====

generate\_random\_series(N, Series) :-

    Max is 20,                         % максимальне значення для генерації

    findall(X, (between(1, N, \_), random(0, Max, R), X is round(R\*100)/100), Series).

% ===== 10. Головний потік програми =====

main :-

    write('Read from CSV? (y/n): '), read(Mode),

    ( Mode == y -> file\_flow ; manual\_flow ).

% файлова гілка: зчитуємо дані з data.csv

file\_flow :-

    write('Reading data.csv...'), nl,

    write('How many elements? '), read(N),

    write('Alphabet size M? '), read(M),

    csv\_read\_file('data.csv', [H|Rows], [functor(row), convert(true)]),

    H =.. [\_|Cols], nth1(ColI, Cols, 'Price'),

    length(Pref, N), append(Pref, \_, Rows),

    findall(X, (member(R, Pref), R =.. [\_|Vs], nth1(ColI, Vs, X)), Series),

    process\_data(Series, M).

% інтерактивна гілка: генерація чи введення вручну

manual\_flow :-

    write('Enter number of elements (N): '), read(N),

    write('Enter \u0027-1\u0027 for auto-generation or \u00270\u0027 for manual input: '), read(InputMode),

    (

        InputMode == -1 ->

            write('Enter alphabet size (M >= 2): '), read(M),

            generate\_random\_series(N, Series)

    ;   InputMode == 0 ->

            write('Enter series as list: '), read(Series),

            write('Enter alphabet size (M >= 2): '), read(M)

    ;   write('Invalid input mode.'), nl, fail

    ),

    process\_data(Series, M).

% обробка даних: сортування, вивід, трансформація, матриця

process\_data(Series, M) :-

    msort(Series, Sorted),                 % сортуємо значення

    Sorted = [MinV|\_], last(Sorted, MaxV),  % визначаємо мінімум і максимум

    nl, write('Sorted numerical series:'), nl, print\_series(Sorted),

    chi\_table(ChiTab), print\_boundaries(ChiTab),

    transform\_series(Sorted, ChiTab, M, MinV, MaxV, Ling),

    nl, write('Linguistic series:'), nl, atomic\_list\_concat(Ling, '', Str), write(Str), nl,

    msort(Ling, SortedLing),               % сортуємо літери для виводу окремого ряду

    nl, write('Sorted linguistic series:'), nl, atomic\_list\_concat(SortedLing, '', SLStr), write(SLStr), nl,

    build\_transitions(Ling, Trans), print\_transition\_matrix(Trans, M),

    nl, write('Done.'), nl.

:- initialization(main).