

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Лабораторна робота № 4
з дисципліни “Мультипарадигмене програмування”

Виконав:
Студент групи ІО-23
Швед А. Д.

Київ - 2025

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 та матриця передування.

Мова програмування

SWI-Prolog.

Хід роботи

Програма реалізує перетворення числового ряду у лінгвістичний ланцюжок з наступним побудуванням матриці передування.

Правила

- Введення алфавіту користувачем
- Читання даних з файлу

Визначення діапазону значень

Правила встановлюють мінімальне та максимальне значення в числовому ряді для подальшого розбиття діапазону.

Побудова рівномірних інтервалів

Використовуючи знайдені границі та потужність алфавіту, визначаються рівномірні інтервали, кожен з яких асоціюється з символом алфавіту.

Визначення відповідності числа інтервалу

Для кожного елемента числового ряду визначається інтервал, в який він потрапляє. З цієї інформації числу надається відповідний символ алфавіту.

Побудова лінгвістичного ряду

На основі попереднього правила кожне число замінюється на відповідну літеру, утворюючи послідовність лінгвістичний ряд.

Побудова пар

Формується список усіх сусідніх пар символів у лінгвістичному ряді.

Побудова матриці передування

Кожна можлива пара символів розглядається як елемент матриці. Підраховується кількість кожної пари, що зустрічається у списку пар. Отримані частоти заносяться у таблицю – матрицю передування, яка є результатом аналізу.

Також були написані службові предикати що не мають особливого логічного сенсу але є частиною реалізації роботи.

Результати виконання

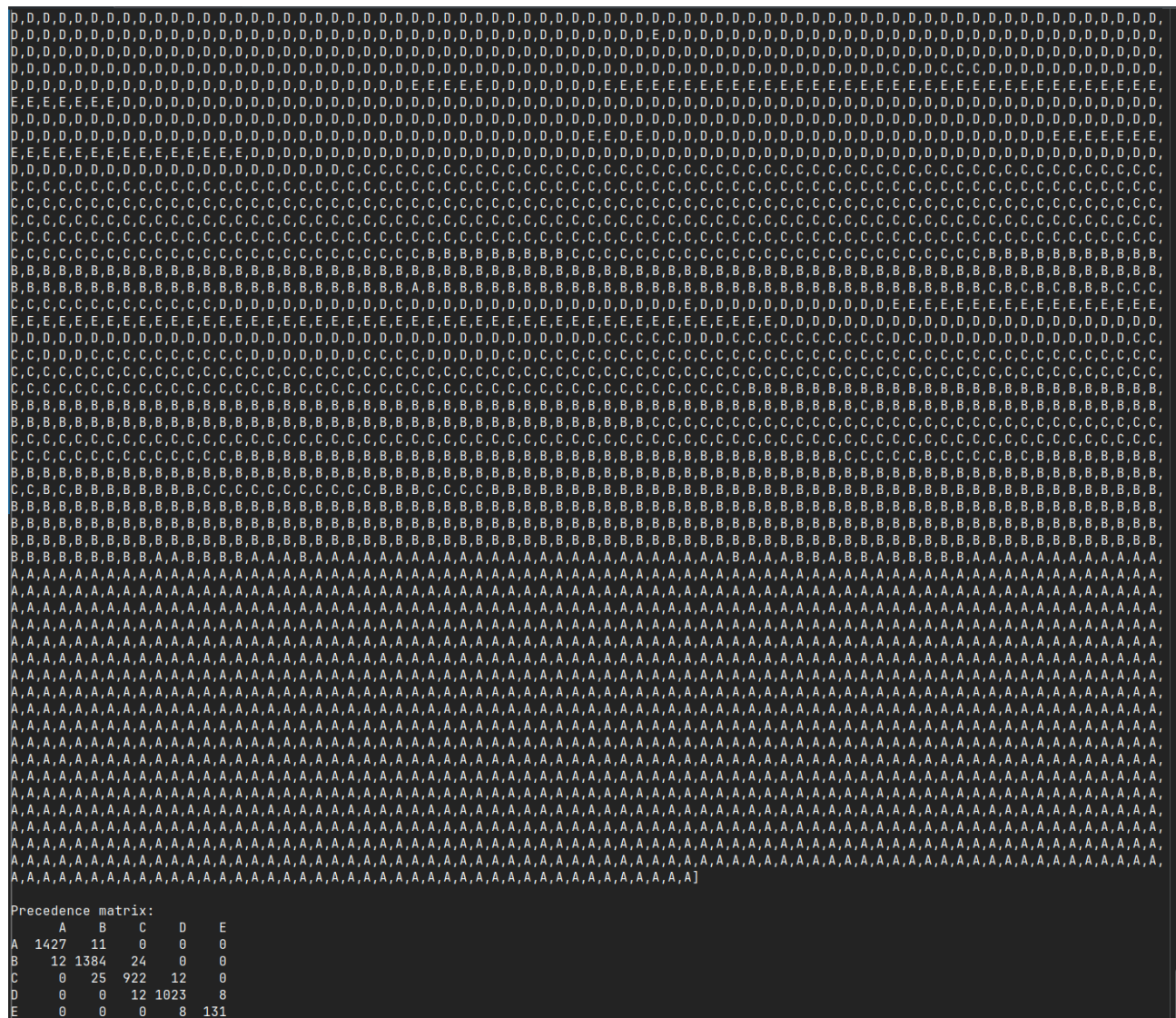


Figure 1: Перший числовий ряд B-C-D-E-F-Brent Oil Futures Historical Data (5000 значень - 5 символів)

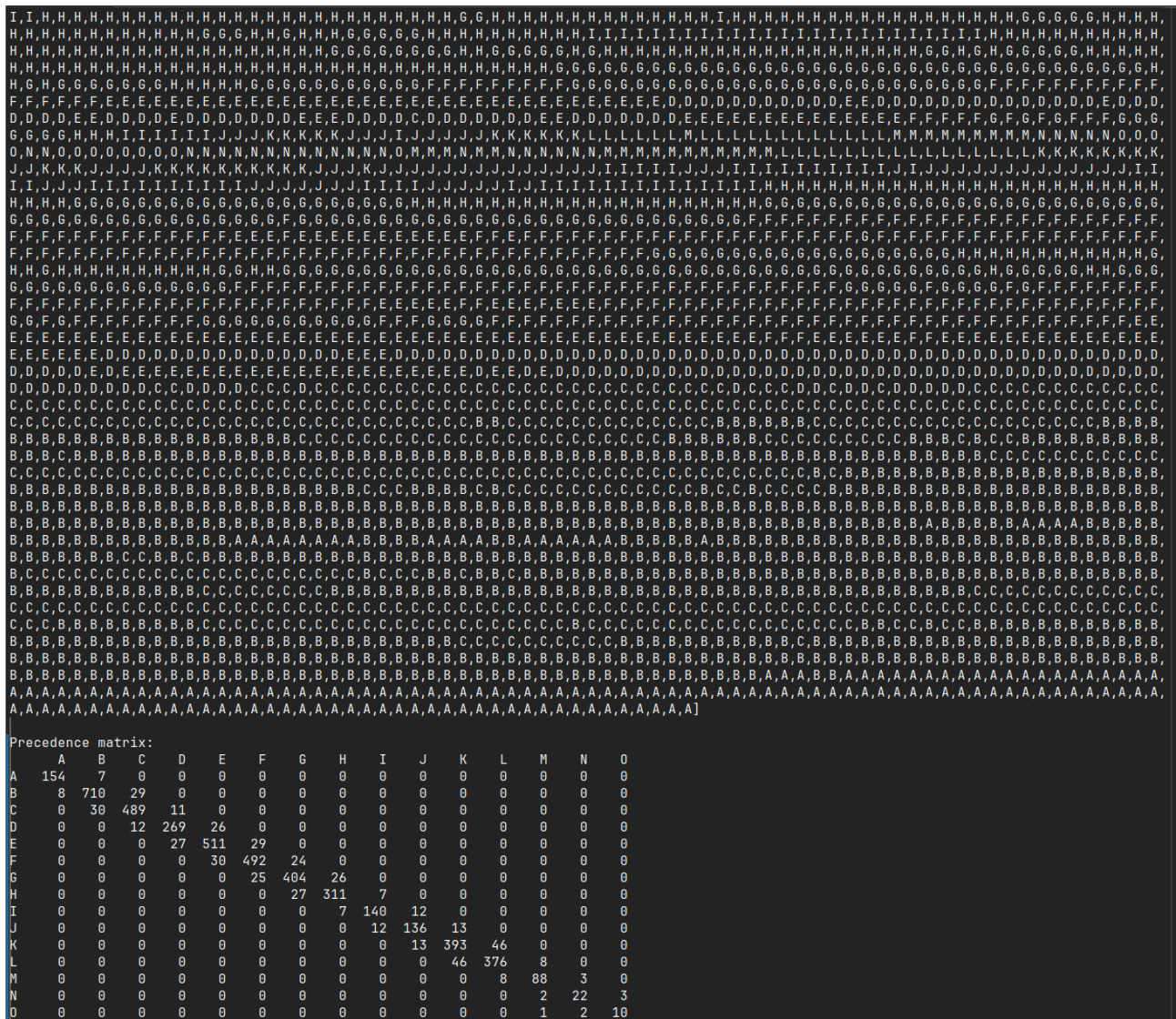


Figure 2: Перший числовий ряд B-C-D-E-F-Brent Oil Futures Historical Data (5000 значень - 15 символів)

Лістинг коду

main.pl

```
1 % Read a string list from terminal
2 read_atom_list(List) :-
3     read_line_to_string(user_input, Line),
4     split_string(Line, " ", "", StringList),
5     maplist(atom_string, List, StringList).
6
7 % Read a list of floats from a file
8 read_floats_from_file(FilePath, Floats) :-
9     open(FilePath, read, Stream),
10    read_lines(Stream, Lines),
11    close(Stream),
12    maplist(atom_number, Lines, Floats).
13
14 % Read lines from file and return them as atom string list
15 read_lines(Stream, []) :-
16     at_end_of_stream(Stream), !.
17 read_lines(Stream, [LineAtom|Rest]) :-
18     read_line_to_string(Stream, Line),
19     atom_string(LineAtom, Line),
20     read_lines(Stream, Rest).
21
22 % Find min element in a list
23 min([H|T], Min) :- foldl(min, T, H, Min).
24 min(A, B, Min) :- Min is min(A, B).
25
26 % Find max element in a list
27 max([H|T], Max) :- foldl(max, T, H, Max).
28 max(A, B, Max) :- Max is max(A, B).
29
30 % Building [A, B) intervals
31 build_intervals(Min, Max, N, Intervals) :-
32     Step is (Max - Min) / N,
33     build_intervals_helper(Min, Step, N, Intervals).
34
35 build_intervals_helper(_, _, 0, []) :- !.
36 build_intervals_helper(Min, Step, N, [[Min, Max1]|Rest]) :-
37     Max1 is Min + Step,
38     N1 is N - 1,
39     build_intervals_helper(Max1, Step, N1, Rest).
40
41 % Finding the interval to which a value belongs
42 value_interval(Value, [[A,B]|_], 0) :-
43     Value >= A, Value < B, !.
44 value_interval(Value, [_|T], Index) :-
45     value_interval(Value, T, Temp),
46     Index is Temp + 1.
47 value_interval(_, [], 0) :- !. % right limit fallback
48
49 % Converting a value to an alphabet symbol
50 value_to_symbol(Value, Intervals, Alphabet, Symbol) :-
51     value_interval(Value, Intervals, Index),
52     length(Alphabet, L),
53     (Index >= L -> LastIndex is L - 1 ; LastIndex is Index),
54     nth0(LastIndex, Alphabet, Symbol).
55
56 % Converting a number series to a linguistic series
57 build_linguistic_sequence([], _, _, []).
58 build_linguistic_sequence([H|T], Intervals, Alphabet, [S|Rest]) :-
59     value_to_symbol(H, Intervals, Alphabet, S),
60     build_linguistic_sequence(T, Intervals, Alphabet, Rest).
61
62
63 % === Building a precedence matrix ===
64 % Builds a list of pairs (a->b, b->c, ...)
65 pairs([], []).
66 pairs([_], []).
67 pairs([A,B|T], [(A,B)|Rest]) :-
```

```

68     pairs([B|T], Rest).
69
70 % Count the occurrences of each pair
71 count_pairs([], _, 0).
72 count_pairs([(A,B)|T], (A,B), N) :-
73     count_pairs(T, (A,B), N1),
74     N is N1 + 1.
75 count_pairs([(X,Y)|T], (A,B), N) :-
76     (X \= A ; Y \= B),
77     count_pairs(T, (A,B), N).
78
79 % Building a matrix row
80 build_matrix_row(_, [], _, []).
81 build_matrix_row(From, [To|T], Transitions, [Count|Rest]) :-
82     count_pairs(Transitions, (From, To), Count),
83     build_matrix_row(From, T, Transitions, Rest).
84
85 % Building a full precedence matrix
86 build_precedence_matrix(_, [], _, []).
87 build_precedence_matrix(Alphabet, [From|RestFrom], Transitions, [Row|MatrixRest]) :-
88     build_matrix_row(From, Alphabet, Transitions, Row),
89     build_precedence_matrix(Alphabet, RestFrom, Transitions, MatrixRest).
90
91
92 % === Formatted matrix printing ===
93 % Pretty print the matrix
94 print_matrix(Matrix, Labels) :-
95     print_header(Labels),
96     print_rows(Matrix, Labels).
97
98 % Print aligned matrix header
99 print_header(Labels) :-
100     tab(2), % Top-left corner space
101     forall(member(Label, Labels),
102         format('~|~t-w~5+', [Label])),
103     nl.
104
105 % Print aligned matrix rows
106 print_rows([], []).
107 print_rows([Row|RestMatrix], [Label|RestLabels]) :-
108     format('~w ', [Label]),
109     forall(member(Cell, Row),
110         format('~|~t-d~5+', [Cell])),
111     nl,
112     print_rows(RestMatrix, RestLabels).
113
114
115 % Main function
116 main :-
117     write("Input space-separated alphabet: "),
118     read_atom_list(Alphabet),
119     read_floats_from_file("data.txt", Series),
120     min(Series, Min),
121     max(Series, Max),
122     length(Alphabet, N),
123     build_intervals(Min, Max, N, Intervals),
124     build_linguistic_sequence(Series, Intervals, Alphabet, Linguistic),
125
126     write('Linguistic sequence: '), writeln(Linguistic),
127     pairs(Linguistic, Transitions),
128     build_precedence_matrix(Alphabet, Transitions, Matrix),
129     nl, writeln('Precedence matrix:'),
130     print_matrix(Matrix, Alphabet),
131     halt.

```