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

Лабораторна робота № 5

з дисципліни «Мультипарадигмене програмування»

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

Київ - 2025

Завдання

За допомогою продукційного 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

CLIPS.

Хід роботи

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

Правила

1. Дані зчитуються з текстового файлу за допомогою функції load-numbers. Кожне значення файлу додається до факту numbers, що дозволяє зберігати числовий ряд для подальшої обробки.
2. Числовий ряд сортується за допомогою функції insert-ordered. Числа додаються до списку, рісля чого список сортується для побудови інтервалів.
3. Після сортування числового ряду, інтервали для алфавіту розраховуються за допомогою функції assign-symbols. Кількість інтервалів визначається кількістю символів алфавіту. Кожне число в числовому ряді потрапляє в один з інтервалів, і відображається у відповідний символ з алфавіту.
4. Після того як числа прив’язуються до символів, створюється лінгвістичний ряд, що відображає розподіл чисел по символах.
5. За лінгвістичним рядом створюється матриця передування, що показує кількість випадків, коли одна буква слідує за іншою.
6. Виведення результатів

При виконанні роботи було помічено, що обробка 5000 значень на CLIPS відбувалася значно довше, ніж у попередніх лабораторних робітах. На це є кілька причин.

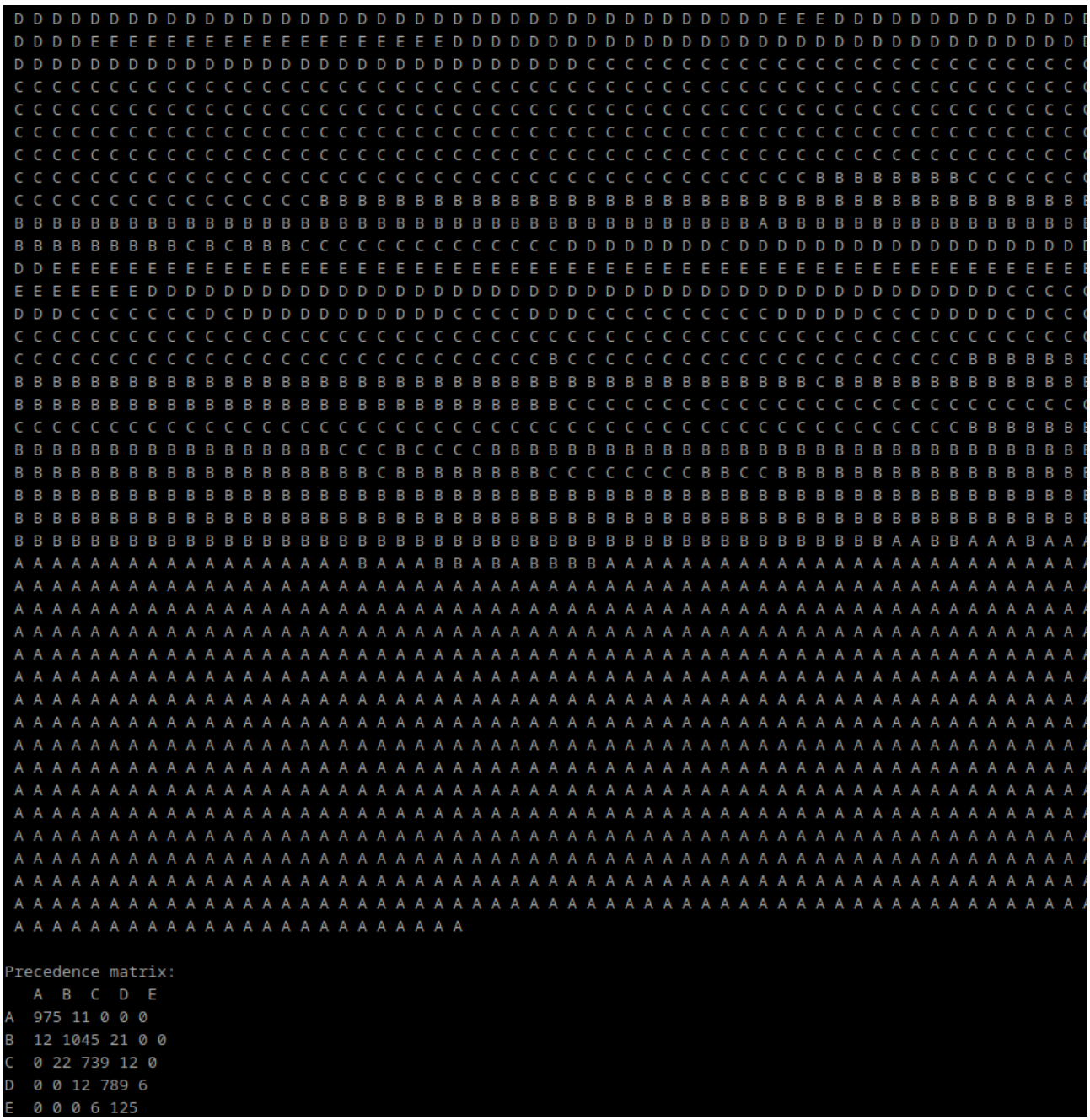
По-перше, CLIPS не має вбудованого швидкого сортування, тому для сортування числового ряду було написано код для сортування алгоритмом вставки, що є дуже повільним на великих масивах.

По-друге, обробка кожного факту у CLIPS відбувається окремо, що призводить до значного навантаження на процесор за великої кількості значень.

Оскільки CLIPS орієнтований на невеликі обсяги даних і обробку бази фактів лімітованого розміру, робота з великикими обсягами даних стає значно менш ефективною порівняно з

іншими мовами програмування, що підтримують більш оптимальні для цього алгоритми та структури даних.

Результати виконання



```
Precedence matrix:
   A  B  C  D  E
A  975 11 0 0 0
B  12 1045 21 0 0
C  0 22 739 12 0
D  0 0 12 789 6
E  0 0 0 6 125
```

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

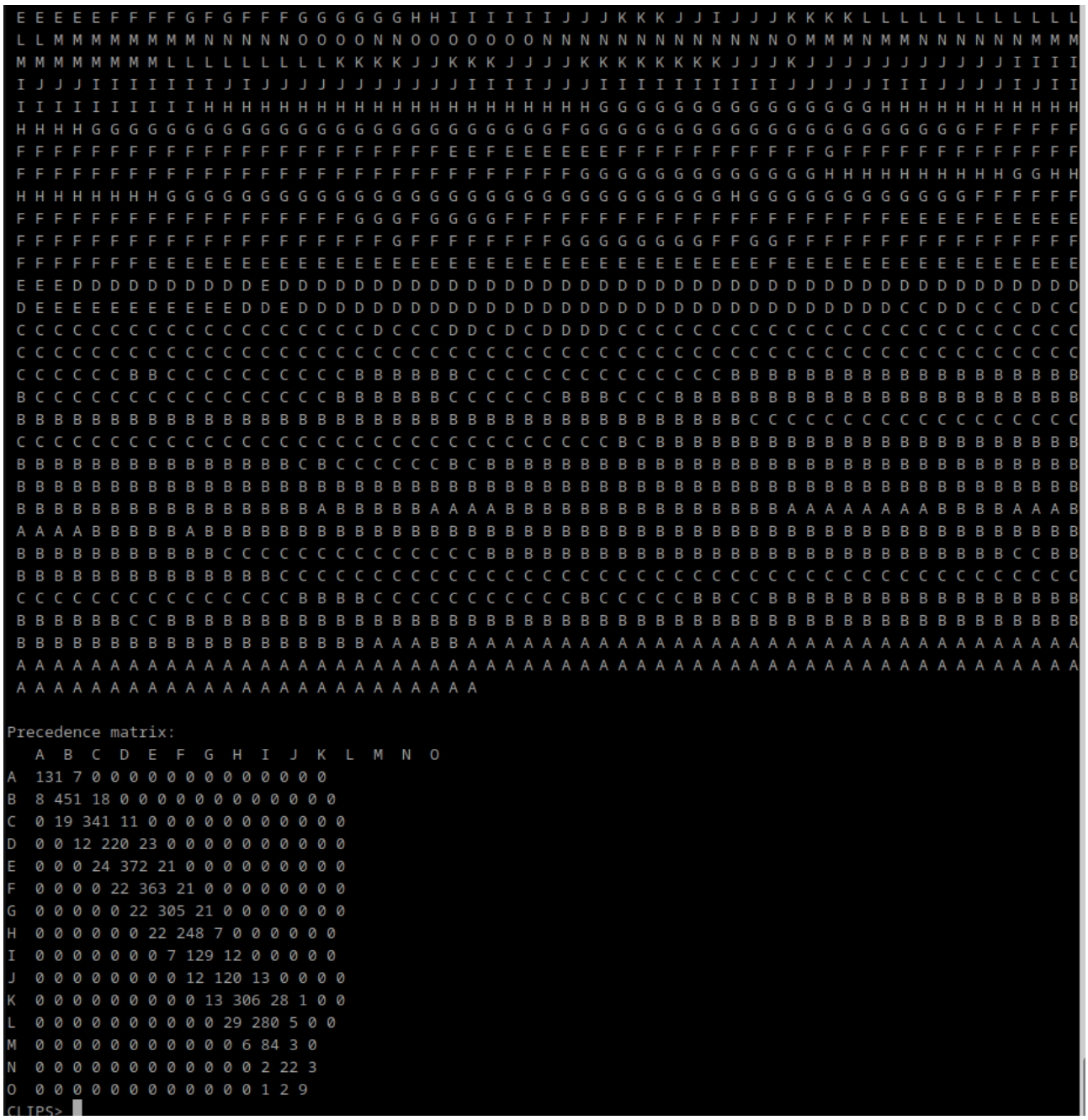


Рисунок 2: Перший числовий ряд B-C-D-E-F-Brent Oil Futures Historical Data (5000 значень - 15 символів)

Лістинг коду

main.clp

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Templates
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(deftemplate numbers (slot value)) ;; Template for a number series
(deftemplate symbol-mapped (slot value) (slot symbol)) ;; Template for mapping numbers to
symbols
(deftemplate linguistic-pair
    (slot from) ;; Template for linguistic pairs
    (slot to)
    (slot pair-id))
(deftemplate status (slot stage)) ;; Template to save statuses for every step
(deftemplate sorted-list (multislot values)) ;; Template for a sorted list


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Globals
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; Alphabet definition
(defglobal
    ?*alphabet* = (create$ A B C D E F G H I J K L M N O))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Functions
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; File reading function
(deffunction load-numbers (?filename)
    (bind ?opened (open ?filename filein))
    (if (eq ?opened FALSE) then
        (printout t "Could not open " ?filename crlf)
        (return))
    (loop-for-count (?i 1 10000)
        (bind ?val (read filein))
        (if (eq ?val EOF) then (return))
        (assert (numbers (value ?val)))) ;; Reading the numbers as facts
    (close filein)
    (printout t "Data read from " ?filename crlf)
)

;; Insertion into sorted sequence function
(deffunction insert-ordered (?val ?sorted)
    ;; Функція для вставки значення в відсортований ряд
    (bind ?result (create$))
    (bind ?inserted FALSE)
    (foreach ?x ?sorted
        (if (and (not ?inserted) (< ?val ?x)) then
        (bind ?result (create$ ?result ?val))
        (bind ?inserted TRUE))
        (bind ?result (create$ ?result ?x)))
    (if (not ?inserted) then
        (bind ?result (create$ ?result ?val)))
    ?result)

;; Function to reflect numbers into symbols
(deffunction assign-symbols (?vals)
    (bind ?count (length$ ?*alphabet*)) ;; Length of the alphabet
    (bind ?min (nth$ 1 ?vals)) ;; Minimal value
    (bind ?max (nth$ (length$ ?vals) ?vals)) ;; Maximal value
    (bind ?step (/ (- ?max ?min) ?count)) ;; Interval width

    ;; Creating the intervals
    (bind ?intervals (create$))
    (bind ?i 1)
    (while (<= ?i ?count)
        (bind ?start (+ ?min (* (- ?i 1) ?step)))
```

```
        (bind ?end (+ ?start ?step))
        (bind ?sym (nth$ ?i ?*alphabet*)) ;; Determining the number
        (bind ?intervals (create$ ?intervals ?start ?end ?sym))
        (bind ?i (+ ?i 1)))

    ;; Binding numbers to symbols
    (do-for-all-facts ((?n numbers)) TRUE
        (bind ?v ?n:value)
        (bind ?j 0)
        (while (< ?j (* ?count 3))
            (bind ?a (nth$ (+ ?j 1) ?intervals))
            (bind ?b (nth$ (+ ?j 2) ?intervals))
            (bind ?s (nth$ (+ ?j 3) ?intervals))
            (if (or (and (>= ?v ?a) (< ?v ?b))
                (and (= ?v ?max) (= ?b ?max))) then
                (assert (symbol-mapped (value ?v) (symbol ?s))) ;; Reflecting the symbol
                (bind ?j (* ?count 3))) ; вихід з циклу
            (bind ?j (+ ?j 3)))))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Rules
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Number-sorting rule
(defrule sort-values
    =>
        (bind ?raw (create$))
        (do-for-all-facts ((?n numbers)) TRUE
            (bind ?raw (create$ ?raw ?n:value)))
        (bind ?sorted (create$))
        (foreach ?val ?raw
            (bind ?sorted (insert-ordered ?val ?sorted))) ;; Using a sorting function
        (assert (sorted-list (values ?sorted))) ;; Saving a sorted sequence
        (assert (status (stage ready))) ;; Ready status
)

;; Symbol-number mapping rule
(defrule map-values-to-symbols
    ?sorted <- (sorted-list (values $?vals)) ;; Checking if a number sequence is sorted
    (status (stage ready)) ;; Checking the ready status
    =>
        (assign-symbols ?vals) ;; Calling a reflection function
)

;;  Rule to build linguistic pairs
(defrule build-pairs
    =>
        (bind ?all (find-all-facts ((?s symbol-mapped)) TRUE)) ;; Finding all facts
        (bind ?len (length$ ?all)) ;; Taking a number of facts
        (loop-for-count (?i 1 (- ?len 1))
            (bind ?from (fact-slot-value (nth$ ?i ?all) symbol)) ;; Taking a symbol from a
current fact
            (bind ?to (fact-slot-value (nth$ (+ ?i 1) ?all) symbol)) ;; Taking a symbol
from the next fact
            (assert (linguistic-pair (from ?from) (to ?to) (pair-id ?i)))) ;; Creating a
linguistic pair
)

;; Rule to print the linguistic sequence
(defrule print-linguistic-sequence
    =>
        (printout t crlf "Linguistic sequence: ")
        (do-for-all-facts ((?s symbol-mapped)) TRUE
            (printout t ?s:symbol " ")) ;; Print every symbol in a sequence
        (printout t crlf)
)

;; Rule to print the precedence matrix
(defrule print-precedence-matrix
=>
    (printout t crlf "Precedence matrix:" crlf)
```

```
    (printout t " ")
    (foreach ?col ?*alphabet*
        (printout t "  " ?col)) ;; Printing row header
    (printout t crlf)
    (foreach ?row ?*alphabet*
        (printout t ?row " ") ;; Printing column header
        (foreach ?col ?*alphabet*
            (bind ?count (length$ (find-all-facts ((?t linguistic-pair))
                (and (eq ?t:from ?row) (eq ?t:to ?col)))))
            (printout t " " ?count)) ;; Printing the amount of pairs for every symbol combination
        (printout t crlf))
)
```