**1. Implementați în CLIPS sau în PROLOG un program care citește (sau primește ca parametru) o listă de litere și afișează doar vocalele distincte incluse în acea listă.**

**Exemplu: citește A B A C E afișează A E**

**A) Implementare în CLIPS**

(defrule citeste

(not (lista $?))

=>

(assert (lista (explode$ (readline)))

)

(deffacts fapte

(vocale A E I O U)

)

(defrule afiseaza

(lista $? ?x $?)

?a<- (vocale $?b ?x $?e)

=>

(printout t ?x)

(retract ?a)

(assert (vocale $?b $?e))

)

Prolog1:

(apare (L, []).

(apare (L, [P|V]) :- (member (P L)) , apare (L, V).

**B) Implementare în Prolog2**

% Predicate to read input and print distinct vowels

read\_and\_print\_vowels :-

write('Introduceti literele separate prin spatiu: '),

read\_line\_to\_string(user\_input, Input),

split\_string(Input, " ", "", Letters),

list\_to\_set(Letters, UniqueLetters),

find\_vowels(UniqueLetters, Vowels),

format('Vocalele distincte sunt: ~w', [Vowels]).

% Predicate to filter vowels

find\_vowels([], []).

find\_vowels([H|T], [H|Vowels]) :-

member(H, ["A", "E", "I", "O", "U", "a", "e", "i", "o", "u"]),

find\_vowels(T, Vowels).

find\_vowels([\_|T], Vowels) :-

find\_vowels(T, Vowels).

% Run the program

:- read\_and\_print\_vowels.

Explicații Prolog

Predicatul read\_and\_print\_vowels/0:

Solicită utilizatorului să introducă literele separate prin spațiu.

Citește linia de text și o convertește într-o listă de litere.

Utilizează list\_to\_set/2 pentru a obține literele unice.

Filtrează vocalele distincte folosind find\_vowels/2.

Afișează vocalele distincte.

Predicatul find\_vowels/2:

Filtrează vocalele din lista de litere unice.

------------------------------------------------------------------------------

**2. Implementați în CLIPS un program care citește câte un număr de la linia de comandă până când este introdus un număr mai mic decât cel anterior.**

(defrule citeste

(anterior ?)

=>

(assert (citit (read)))

)

(defrule verifica

?a<-(anterior ?x)

?b<-(citit ?y)

(test (< ?x ?y))

=>

(retract ?a ?b)

(assert (anterior ?y))

)

--------------------------------------------------------------

**3. Suma și Media unei Liste de Numere**

**Definirea Faptelor**

;; Regula pentru citirea listei de la utilizator

(defrule citeste

(not (lista $?))

=>

(assert (lista (explode$ (readline)))))

;; Regula pentru calcularea sumei

(defrule calculate-sum

(lista $? numbers)

=>

(bind ?sum 0)

(foreach ?number (create$ numbers)

(bind ?sum (+ ?sum ?number)))

(assert (sum ?sum))

(printout t "Suma numerelor este: " ?sum crlf))

;; Regula pentru calcularea mediei

(defrule calculate-average

(sum ?sum)

(lista $? numbers)

=>

(bind ?count (length$ (create$ numbers)))

(bind ?average (/ ?sum ?count))

(assert (average ?average))

(printout t "Media numerelor este: " ?average crlf))

**4. Definirea Regulilor pentru Găsirea Maximului și Minimului**

;; Regula pentru găsirea maximului

(defrule find-max

(lista $? numbers)

=>

(bind ?max (first$ (create$ numbers)))

(do-for-all-facts ((?number number) (lista $?))

(bind ?max (max ?max ?number)))

(assert (max-number ?max))

(printout t "Numarul maxim este: " ?max crlf))

;; Regula pentru găsirea minimului

(defrule find-min

(lista $? numbers)

=>

(bind ?min (first$ (create$ numbers)))

(do-for-all-facts ((?number number) (lista $?))

(bind ?min (min ?min ?number)))

(assert (min-number ?min))

(printout t "Numarul minim este: " ?min crlf))

**RETE**

