Paradigmas de la Programacion

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Trabajo Práctico 2

Grupo Borbotones

Integrante	LU	Correo electrónico
Hernandez, Nicolas	122/13	nicoh22@hotmail.com
Tripodi, Guido	843/10	guido.tripodi@hotmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Contents

1 Ejercicio

1.1 Codigo

```
symbol(a).
symbol(b).
symbol(c).

% Algunas regex de ejemplo

regexEj(1, a). % a
regexEj(2, or(a, b)). % a|b
regexEj(3, concat(E1, E2)) :- regexEj(1, E1), regexEj(2, E2). % a(a|b)
regexEj(4, star(E2)) :- regexEj(2, E2). % (a|b)*
regexEj(5, or(star(E1), E4)) :- regexEj(1, E1), regexEj(4, E4). % (a*|(a(a|b))*)
regexEj(6, star(or(a, ab))). %(a|ab)*
regexEj(7, concat(or(a, concat(a,b)), or(b, empty))). %(a|ab)(b|)
regexEj(8, concat(star(a), star(b))). %a*b*
regexEj(9, star(or(star(a), star(b)))).


% Ejercicio 1: tieneEstrella(+RegEx)


tieneEstrella(concat(X,_)) :- tieneEstrella(X).
tieneEstrella(concat(_,Y)) :- tieneEstrella(Y).
tieneEstrella(or(X,_)) :- tieneEstrella(X).
tieneEstrella(or(_,Y)) :- tieneEstrella(Y).
tieneEstrella(star(_)).


% Ejercicio 2: longitudMaxima(+RegEx, -Length)

longitudMaxima(empty, 0).
longitudMaxima(Cadena,Long) :- symbol(Cadena), Long is 1.
longitudMaxima(or(X,Y), Long) :-
    not(tieneEstrella(or(X,Y))),
    longitudMaxima(X, Long1),
    longitudMaxima(Y,Long2),
    Long is max(Long1,Long2).
longitudMaxima(concat(X,Y), Long) :-
    not(tieneEstrella(concat(X,Y))),
    longitudMaxima(X, Long1),
    longitudMaxima(Y, Long2),
    Long is Long1 + Long2.

% Ejercicio 3: cadena(?Cadena)

cadena([]).
cadena([X | XS]):- cadena(XS), symbol(X).
```

```
% Ejercicio 4: match_inst(+Cadena, +RegEx)

match_inst([], empty).
match_inst([X], X) :- symbol(X).
match_inst(Cadena, or(X,_)) :- match_inst(Cadena, X).
match_inst(Cadena, or(_,Y)) :- match_inst(Cadena, Y).
match_inst(Cadena, concat(Y,Z)) :-
    append(C1, C2, Cadena),
    match_inst(C1,Y),
    match_inst(C2, Z).
match_inst([], star(_)). %0 apariciones.
match_inst(Cadena, star(Y)) :-
    append(C1, C2, Cadena),
    not(length(C1,0)),
    match_inst(C1, Y),
    match_inst(C2, star(Y)).

% Ejercicio 5: match(?Cadena, +RegEx)

match(Cadena, RegEx) :- cadena(Cadena), match_inst(Cadena, RegEx).

% Ejercicio 6: diferencia(?Cadena, +RegEx, +RegEx)

diferencia(Cadena, Exp1, Exp2) :-
    match(Cadena, Exp1),
    not(match(Cadena,Exp2)).

% Ejercicio 7: prefijoMaximo(?Prefijo, +Cadena, +RegEx)
prefijoMaximo(Prefijo, Cadena, Exp) :-
    append(Prefijo,_,Cadena),
    match(Prefijo, Exp),
    length(Prefijo, T),
    not(hayPrefijoMayor(Cadena,Exp,T)).

%hayPrefijoMayor(+Cadena, +Exp, +T)
hayPrefijoMayor(Cadena, Exp, T):-
    append(Prefijo,_,Cadena),
    length(Prefijo, TI),
    TI > T,
    match(Prefijo,Exp).

% Ejercicio 8: reemplazar(+X, +R, +E, Res)

reemplazar([], _, _, []).

%si no tengo un prefijo que matchee, busco en la cola de la cadena.
reemplazar([X | XS], Exp, Sust, [X | Rec]) :-
    not(prefijoMaximo(_, [X | XS], Exp)),
    reemplazar(XS, Exp, Sust, Rec).
```

```
%Si el prefijo maximo es el vacio, lo saltamos.
reemplazar([X | XS], Exp, Sust, [X | Rec]) :-
    prefijoMaximo([], [X | XS], Exp),
    reemplazar(XS, Exp, Sust, Rec).

%Si hay un prefijo maximo bueno.
reemplazar(Cadena, Exp, Sust, Res) :-
    prefijoMaximo(P, Cadena, Exp),
    length(P,TamP),
    TamP > 0,
    append(P, D, Cadena),
    reemplazar(D, Exp, Sust, Rec),
    append(Sust, Rec, Res).


% test_2_Y: longitudMaxima

test_2_Y :- test_2_1, test_2_2, test_2_3.

test_2_1 :- longitudMaxima(or(a, b), T), T == 1.
test_2_2 :- longitudMaxima(concat(concat(concat(a,b),b),a),T), T == 4.
test_2_3 :- longitudMaxima(or(concat(concat(a,b),b),a),T), T == 3.
test_2_4 :- longitudMaxima(or(concat(concat(a,star(b)),b),a),T), T == false.

% test_3_Y: longitudMaxima

test_3_Y :- test_3_1, test_3_2, test_3_3, test_3_4.

test_3_1 :- cadena(C), C == [a,b,c].
test_3_2 :- cadena(C), C == [a,a,a,a,a].
test_3_3 :- cadena(C), C == [a,a,b].
test_3_4 :- cadena(C), C == [a,b,c,a,b].

% test_5_Y: match

test_5_Y :- test_5_1, test_5_2, test_5_3, test_5_4.

test_5_1 :- match(X, star(a)), X == [a,a,a,a].
test_5_2 :- match(X, star(concat(star(a), star(b)))), X == [a, b, b, a, a].
test_5_3 :- match(X, star(or(star(a), star(b)))), X == [b, b].
test_5_4 :- match(X,  or(star(a), star(b))), X = [b,b,b,b,b,b,b].

% test_6_Y: diferencia

test_6_Y :- test_6_1, test_6_2, test_6_3.

test_6_1 :- diferencia(X, star(a), star(b)), X == [a, a, a, a, a, a, a].
test_6_2 :- diferencia(X,concat(a,star(b)),c), X == [a, b, b, b, b, b, b, b, b].
test_6_3 :- diferencia(X,concat(a,or(star(b),a)),c), X == [a, b, b, b, b, b, b, b].
```

```
% test_7_Y: prefijoMaximo

test_7_Y :- test_7_1, test_7_2,test_7_3.

test_7_1 :- prefijoMaximo(P, [a,b,a,b,a,b], concat(star(a),star(b))), P == [a,b].
test_7_2 :- prefijoMaximo(P, [a,a,a,b,b,b,b,b,a,b,b,b], concat(star(a),star(b))), P == [a,
test_7_3 :- prefijoMaximo(P, [a,a,a,b,b,b,b,b,a,b,b,b], concat(a,star(b))), P == [a].

% test_8_Y: reemplazar

test_8_Y :- test_8_1, test_8_2,test_8_3.

test_8_1 :- reemplazar([a,a,a,b,b,a,b,c],concat(a,star(b)), [1],X), X == [1, 1, 1, 1, c].
test_8_2 :- reemplazar([a,a,a,b,b,a,b,c],or(a,star(b)), [1],X), X == [1, 1, 1, 1, 1, 1, c]
test_8_3 :- reemplazar([a,a,a,b,b,a,b,c],concat(star(a),star(c)), [1],X), X == [1, b, b, 1

all_Test :- test_2_Y,test_3_Y,test_5_Y,test_6_Y, test_7_Y, test_8_Y.
```