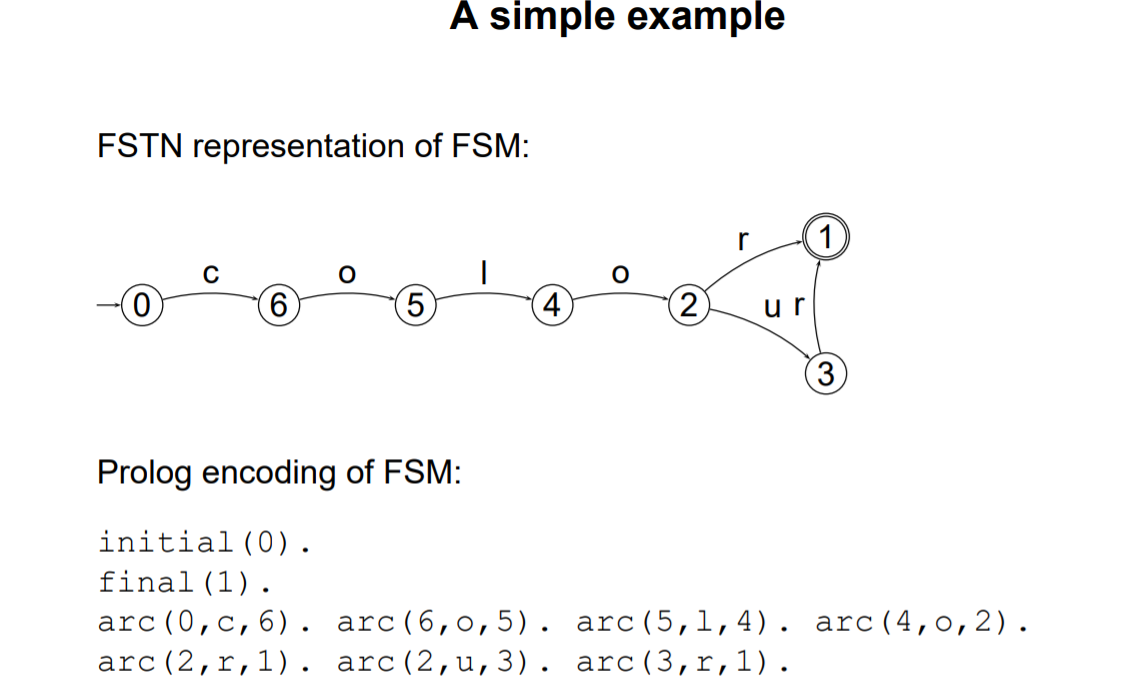
**Write a prolog program that implements Semantic Networks (ATN/RTN).**

**SN1**

****

initial(0).

final(1).

arc(0,c,6). arc(6,o,5). arc(5,l,4).

arc(4,o,3). arc(3,u,2). arc(3,r,1). arc(2,r,1).

start:-write('RTN Test:colo(u)r\n'),

write('Enter character list: '),

read(L),

test(L).

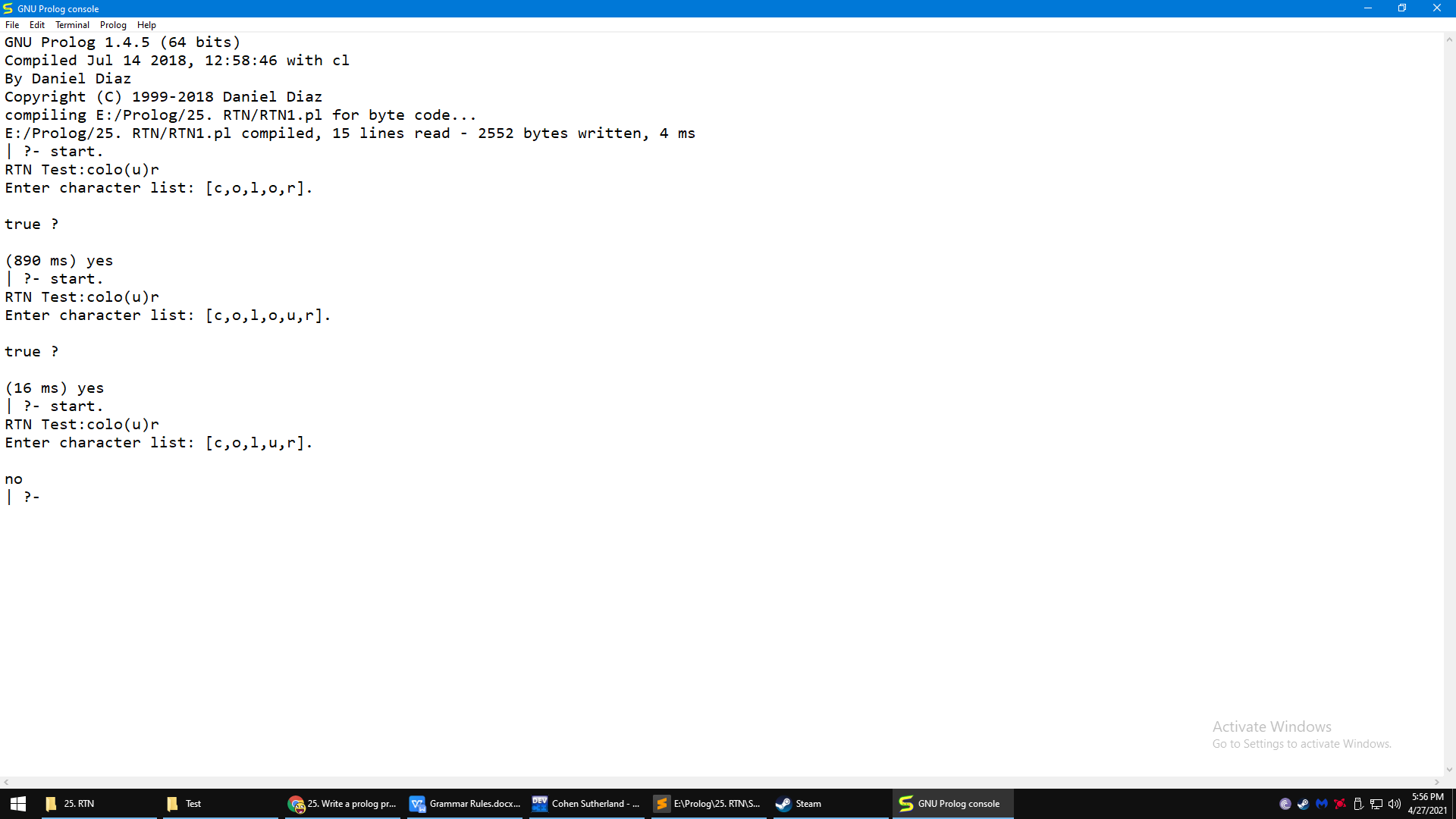
test(Word):-initial(Node),recognise(Node,Word).

recognise(Node,[]):-final(Node).

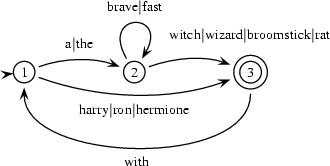
recognise(FromNode,String):-arc(FromNode,Label,ToNode),traverse(Label,String,NewString),recognise(ToNode,NewString).

traverse(First,[First|Rest],Rest).).

**Output**



**SN2**



initial(1).

final(3).

arc(1,det,2).

arc(2,adj,2).

arc(2,cn,3).

arc(1,n,3).

arc(3,prep,1).

lex('The',det). lex('A',det). lex(a,det). lex(the,det).

lex(brave,adj). lex(fast,adj).

lex(witch,cn). lex('Witch',cn). lex(wizard,cn). lex('Witch',cn).

lex('Broomstick',cn). lex(broomstick,cn). lex('Rat',cn). lex(rat,cn).

lex('Harry',n). lex('Ron',n). lex('Hermione',n).

lex(with,prep).

start:-write('RTN Test:Sentence\n'),

write('Enter word list: '),

read(L),

test(L).

test(Words):-initial(Node),recognise(Node,Words).

recognise(Node,[]):-final(Node).

recognise(FromNode,String):-arc(FromNode,Label,ToNode),traverse(Label,String,NewString),recognise(ToNode,NewString).

traverse(Label,[First|Rest],Rest):-lex(First,Label).

**Output**

