**Using operational semantics, describe at least two features of your programming language:**

Note: our answer is based on the information provided in the book of the 10th edition, especially chapter 8. It appears that the operational semantics described in the book are very flexible and don’t follow a specific way of written form.

1. **For loop:**

**Najim Code:**

for (x to y step z){

<statement>

}

**Operational Semantic:**

[ define x as integer that represents start of loop range ]

[ define y as integer that represents the end of loop range ]

[ define z as integer that represents the increment of the loop range]

loop:

if x == y goto out

[ loop body represented in <statement> in Najim code ]

x = x + z

goto loop

out:

... [ other statements after loop body ]

1. **If-else :**

**Najim Code:**

if(condition){

<statement\_1>

}

else {

<statement\_2>

}

**Operational Semantic:**

[ define condition as a boolean that represent either a true or false]

start:

if condition is true then

[ execute <statement\_1> block of code ]

goto out

[ execute <statement\_2> block of code ]

out:

... [ other statements after if-else body ]

**List and explain at least two static semantics, different from the examples that we discussed in class, for the assignment statements of your programming language; describe them using attribute grammar.**

1. Variables that are declared in a child scope (the scope where a statement block is inside another statement block) shall not be accessed from the parent scope. On the other hand, parent scope declared variables can be accessed in the child scopes.

Example:

function example(){

int outside = 10;

if(true){

int inside = 5;

outside = 500; // THIS IS LEGAL STATEMENT

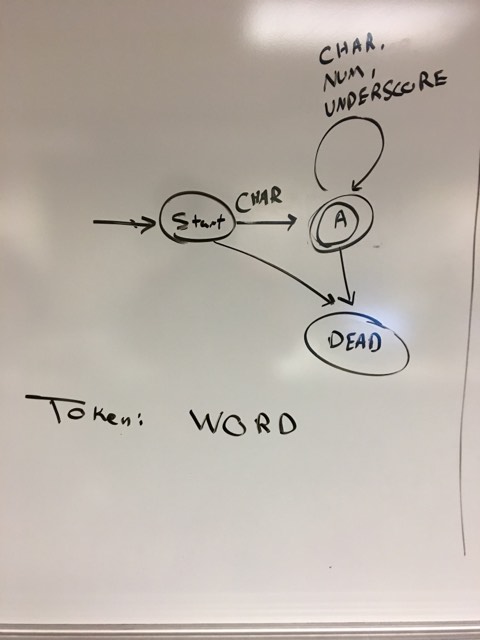
}

inside = 500; // THIS IS ILLEGAL STATEMENT

}

1. A function call is illegal if the function has not been declared prior to the function call.  
     
   Example:  
   function m1(){}  
     
   main(){  
    m1(); // THIS IS LEGAL FUNCTION CALL  
    m2(); // THIS IS ILLEGAL FUNCTION CALL  
   }  
     
   function m2(){}

**Identify and design state transition diagrams for all possible tokens in your programming language. For each state transition diagram, describe the language it represents.  
  
WORD**



**Language description:**

WORD is a string that starts with a valid character and N number of other characters, digits and underscores. CHAR is a collection of characters represented in the following Regular Expression:

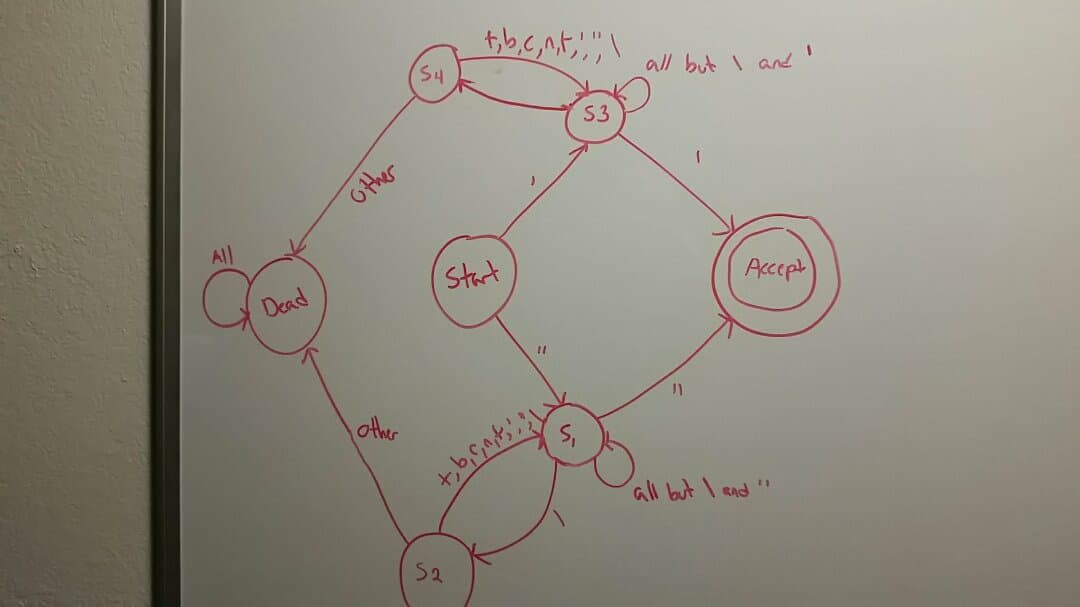
(A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z)

Digits/NUM are represented in the following regex:  
(((0|1)|2|3|4|5|6|7)|8|9)

And UNDERSCORE is represented in the following regular expression: (\_)

Therefore: the language for the token WORD is represented in the following regex:  
((A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z)(((A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z)|(((0|1)|2|3|4|5|6|7)|8|9)|\_)\*))

**STRING-LITERAL**



**Language Description:**

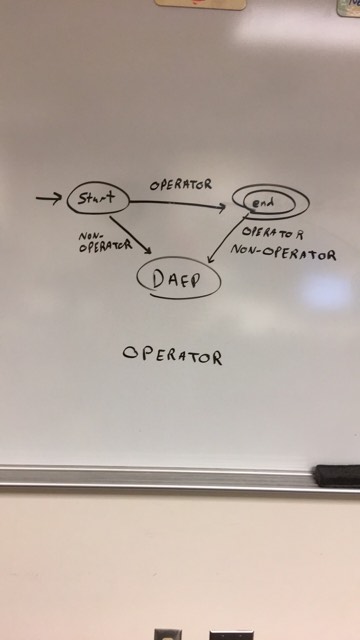
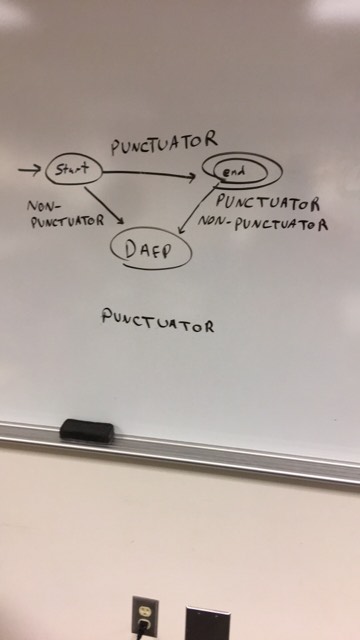
String literals are the any string that starts with ' or " and should end with ' or " respectively and includes anything in between, including escaped characters.

Therefore: the language is represented in the following regular expression:

(('((?:[^\']|\')\*)')|("((?:[^\"]|\")\*)"))

Note: since this language accepts almost all possible data in between the quotes, defining a specific language is not accurate.

**OPERATOR and PUNCTUATOR**



**Language Description:**

Operators and punctuators are any characters that are defined in the following regex (they are combined since they have similar state diagram):

((((\+|\-)?)|\|\|)|(\*|\/|%|&&)|(\\*\\*)|(==|!=|<|<=|>=|>))