**CEN202 Programming Languages Final Homework**

**1. (20 points)** Which characteristics of programming languages do you think are the most important and why? Write your own opinion.

Since the question asked directly *‘*my*’* opinion, I will fully explain my own preferences. There are 3 attributes that make a programming language important to me.

It must:

* be a high level language
* have enough resources on the internet
* maintain its popularity at least until the near future
* High level language:  
  Today, it is very important to produce fast products/services for many companies in the sector. High level languages are one step ahead when it comes to producing quickly. Low-level languages don’t fit the works I am interested in.
* Resources:  
  When we encounter a problem while developing, we use the internet a lot. Therefore, it is always a plus for us that the language is popular and the number of developers using this language is high. Because it means you can reach a lot of resources. It makes the development process much more easier.
* Bright Future:

A language is a tool we use and this tool is an investment we make in ourselves. So basically, nobody wants to make a dead investment. If this language is drawing a decline graph, choosing this language will not be a wise choice.

**2. (30 points)** Design a state diagram to recognize variable names in a new language such that a variable name starts with any one of the **%** or **@** or **$** or **\_** characters, then followed by at least one letter, then followed by any combination of letters and digits. Both lowercase and uppercase letters can be used. As examples; while **%a** and **@x1YZ** are valid variable names; **$1** and **a2bc23xYz** are not valid variable names. After that, write BNF or EBNF rules by using your state diagram to generate the valid variable names in the new language. Show that your rules can generate a valid variable name by applying a leftmost derivation.

S -> %LC | @LC | $LC | \_LC

L -> [a-zA-Z]

C -> [a-zA-Z0-9]C | ε

BNF:   
<variablename> ::= <symbol><letter><more>

<more> ::= <letter><more> | <digit><more> | <empty>

<symbol> ::= % | @ | $ | \_

<letter> ::= a | b | c | d | ………….. | x | y | z | A | B | C | ……………… | Y | Z

<digit> ::= 0 | 1 | 2 | ………… | 8 | 9

EBNF:

<variablename> ::= <symbol><letter> {<letter> | <digit>}

<symbol> ::= % | @ | $ | \_

<letter> ::= a | b | c | d | ………….. | x | y | z | A | B | C | ……………… | Y | Z

<digit> ::= 0 | 1 | 2 | ………… | 8 | 9

Let’s generate some variable names with leftmost derivation and see if they are valid or not.

BNF:

|  |  |
| --- | --- |
| <variablename> | |
| <symbol><letter><more> | by <variablename> -> <symbol><letter><more> |
| @<letter><more> | by <symbol> -> @ |
| @f<more> | by <letter> -> f |
| @f<digit><more> | by <more> -> <digit><more> |
| @f8<more> | by <digit> -> 8 |
| @f8<letter><more> | by <more> -> <letter><more> |
| @f8H<more> | by <letter> -> H |
| @f8H | by <more> -> <empty> |

* “@f8H” is a valid variable name

EBNF:

|  |  |
| --- | --- |
| <variablename> | |
| <symbol><letter> {<letter> | <digit>} | by <variablename> -> <symbol><letter> {<letter> | <digit>} |
| %<letter> {<letter> | <digit>} | by <symbol> -> % |
| %n{<letter> | <digit>} | by <letter> -> n |
| %n(<letter> | <digit>) (<letter> | <digit>) | { … } -> this means zero or more repetitions, for our example let’s say it will repeat twice like (<letter> | <digit>)2 |
| %n3(<letter> | <digit>) | by (<letter> | <digit>) -> <digit>  <digit> -> 3 |
| %n3T | by (<letter> | <digit>) -> <letter>  <letter> -> T |

* “%n3T” is a valid variable name

**3. (30 points)** What is computed in the below code? What is the condition that the row satisfies? Explain briefly. Then, rewrite this code without goto in C language. After that, compare readability of your code to that of the code given below.

for (i=0; i < n; i++){

for (j=0; j < n; j++){

if (x[i][j]!=0)

goto reject;

}

printf(“First row that satisfies the condition is %d\n”, i);

break;

reject:

}

What is computed in the code?

This code computes that in a table x(nxn), whether is there any row such that its all elements are zeros. If there is, prints the index of this row and end the loop.

What is the condition that the row satisfies? Explain briefly.

Condition: x[i][j] is 0, for i ∈ [0, n) and ∀j such that 0 ≤ j < n, (i and j ∈ 𝕫).

If condition is true, then we can say ith row satisfies this condition.

Rewrite this code without goto in C language.

int zeroFlag; //If a row satisfies the condition, zero flag is 1, otherwise 0.

for (i=0; i < n; i++){

zeroFlag = 1; //Initially assume that the row satisfies.

for (j=0; j < n; j++){

if (x[i][j]!=0){ //If we encounter any non-zero element…

zeroFlag = 0; //…zero flag is false, not satisfied.

break;

}

}

if(zeroFlag){

printf(“First row that satisfies the condition is %d\n”, i);

break;

}

}

Compare readability of your code to that of the code given.

To be honest, when I check the code out, it confused me at first and took some time to understand what was actually going on. Most of the time, “goto” is not recommended in Computer Science because it makes things hard to follow and decrease our algorithm quality. Instead of goto, I used a flag in the loop. for every ith row, we initially consider this row is an all-zero-row, therefore we set the zero flag 1.  
If row is not an all-zero-row, zero flag is assigned as false. I think this flag logic is easier to understand. Maybe our code has more lines now, but as long as code explains itself better to the programmer, it doesn’t really matter how long or short it is.

**4. (20 points)** Using the structures parent(X,Y), male(X), and female(X), writestructures that define father(X,Y), mother(X,Y), brother(X,Y), sister(X,Y), grandparent(X,Y), and grandmother(X,Y).

I will use colors to increase recognizability

Blue : Male  
Red : Female  
Yellow : Not specified

father (X, Y) :- male(X), parent(X, Y). mother (X, Y) :- female(X), parent(X, Y).

brother (X, Y) :- male(X), parent(P, X), parent(P, Y).

sister (X, Y) :- female(X), parent(P, X), parent(P, Y).

*Note that: In the Cambridge dictionary, brother and sister defined as:*

*brother -> a boy or man who has the same parents as you*

*sister -> a girl or woman who has the same parents as you*

*According to Cambridge dictionary, both parents must be common for siblings. But in the above chart, I considered ‘‘Siblings have at least one common parent’’. This is why parent P’s gender didn’t declared. According to my definition, for example, two children with the same mother and different fathers are siblings. One common parent is enough. I just wanted to keep expressions as simple as possible.*

grandparent (X, Y) :- parent(X, A), parent(A, Y).

grandmother (X, Y) :- female(X), parent(X, A), parent(A, Y).