CS 3513 - Programming Languages

Programming Project 01

Problem Description

You are required to implement a lexical analyzer and a parser for the RPAL language. Refer RPAL_Lex.pdf for the lexical rules and RPAL_Grammar.pdf for the grammar details.
You should not use 'lex'. 'yacc' or any such tool.

Output of the parser should be the Abstract Syntax Tree (AST) for the given input program. Then you need to implement an algorithm to convert the Abstract Syntax Tree (AST) in to Standardize Tree (ST) and implement CSE machine.

Your program should be able to read an input file which contains a RPAL program.
Output of your program should match the output of "rpal.exe" for the relevant program.

You must use C/C++, Java or Python for this project.

Input and Output Requirements

Your program should execute using the following
For c/c++:

```
$ ./myrpal file_name
```

For java:

```
$ java myrpal file_name
```

For Python:

```
python .\myrpal.py file_name
```

Where file_name is the name of the file that has the RPAL program as the input.

Required switches: -ast. This switch prints the abstract syntax tree, and nothing else.

Input Format

```
let Sum(A) = Psum (A,Order A )
                  where rec Psum (T,N) = N eq 0 -> 0
                         | Psum(T,N-1)+T N
in Print ( Sum (1,2,3,4,5) )
```

OutPut Format of -ast switch

```
let
.function_form
..<ID:Sum>
..<ID:A>
..where
...gamma
....<ID:Psum>
....tau
.....<ID:A>
.....gamma
......<ID:Order>
......<ID:A>
...rec
....function_form
.....<ID:Psum>
.....,
......<ID:T>
......<ID:N>
.....->
......eq
.......<ID:N>
.......<INT:0>
......<INT:0>
......+
.......gamma
........<ID:Psum>
........tau
.........<ID:T>
.........-
..........<ID:N>
..........<INT:1>
.......gamma
........<ID:T>
........<ID:N>
.gamma
..<ID:Print>
..gamma
...<ID:Sum>
...tau
....<INT:1>
....<INT:2>
....<INT:3>
....<INT:4>
```

```
....<INT:5>
```

Output Format without -ast switch

Output of the above program is:

```
15
```

Submission

You must submit the following:

1.Makefile: Your make file must be directly under the zip folder. No nested directories.

2. Source Program: Provide comments.

3. REPORT:
· The report should be in PDF format.
· The report should contain your basic info: Name, and Student ID.
· Present function prototypes showing the structure of your programs. Include the structure of your program.

To submit, Please compress all your files together using a zip utility and submit to the Moodle system.
Your submission should be named ***IndexNumber.zip.***

All email submission will be ignored without further notification. Please note that the due day is a hard deadline. No late submission will be allowed. Any submission after the deadline will not be accepted.

Grading Policy

Grading will be based on the correctness of algorithms. Below are some details of the grading policy.
Correct implementation and execution: 70%
Comments and readability: 10%
Report: 20%
To grade your project, we will run your program on our test programs, and we will compare your output with the correct one. Full credit will be given a perfect match in every case. If your output does not match the correct output you will receive 0 marks for that test case.
Note: ***If you do not follow the input/output or submission requirements above, 25% of your score will be deduced.*** In addition, we may ask you to demonstrate your projects.