README project 5 Function Code Generation

Command to compile

**javac Pair.java Parser.java Parsing.java RegularExpressions.java TokenNames.java Scan.java Token.java Node.java Proscanner.java**

Command to run

**java Parser <inputfilename>**

where inputfilename is the name of the file to be tested. (make sure the file is in the same directory where all the java source code files are.

The submission folder contains a folder java\_code\_files of 9 java files. The provided scanner is used. Reference is taken from the skeleton of IR, I integrated it with my parser and corrected the bugs of my IR. From the skeleton IR the symbol table is generated. The number of local variables for each function is maintained in the hash map.

The Parser.java file generates the function code. All the storage of local variables is obtained from the IR, epilogue of function and the code to emit epilogue is in the func Z of the provide IR in Parser file. A single function is maintained as a stack, the top and base pointers for the jump of functions when called is maintained. Lastly a jump register containing the labels is maintained so that the program execution can know where to jump.

An arraylist ‘finalfile’ stores all the code to emit in terms of token and each element of the arraylist.

Separate hashmap with key as function name and values as parameters to store the parameters.

Arraylist to store all the function from left brace to right brace.

Not able to handle the condition of global arrays and hence the test cases like sort, Fibonacci, are showing exceptions and failing.

Following are the cases that passed

Recursion, parameter, funcall, funcall2, branch, expression, mandel,parameter, tax, times8,trivial, max and others also except the ones having the condition of arrays.