

CSE2260 Principles of Programming Languages 2023

Project#2 Report

Project Team: Akın Ateş, Can Mayda, Nur Betül Çakır

Due:25.05.2023, 23:59

In this Project we made a Recursive Descent parser for a spesific grammar that we are given.

Fort his Project to be succesfully made we needed tokens from our first Project. Let me explain all details we made.

Fort he grammar part I will place a screenshot and under that Picture a screenshot from the code representing that grammar:

1.)For the first one the grammar must etiher be null or the given part and below the code represantation fort his:



A picture containing text, screenshot, font

Description automatically generated

2.) In second part 2 similar grammars given and their represantation in java code:

A screen shot of a computer program

Description automatically generated with low confidenceA picture containing text, font, white, calligraphy

Description automatically generated

3.)For the definiton parts and their java codes:



A screen shot of a computer program

Description automatically generated with low confidence

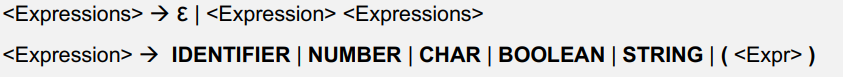
4.)For Arg\_list we need it to be etiher null or the given part:



A picture containing text, screenshot, font

Description automatically generated

5.)For expression and Expressionns here is the grammar and their represantations in java code:



A screen shot of a computer program

Description automatically generated with low confidenceA screen shot of a computer program

Description automatically generated with low confidence

6.)For the expr here is the grammar and represantation:



A screen shot of a computer program

Description automatically generated with low confidence

6.)Next one is the FunCall



A screen shot of a computer code

Description automatically generated with low confidence

7.)Fort he Let expression in any code these are the grammars will be called and here is the grammar of them and their represantiaons:

A picture containing text, font, screenshot, white

Description automatically generated

A screen shot of a computer program

Description automatically generated with low confidence

8.)Two varDef grammars, I coded the Vardefs with help of the paranthesis.

A picture containing text, font, white, typography

Description automatically generated

A screen shot of a computer program

Description automatically generated with low confidence

9.)For the cond expressions there are 3 diffrerent grammars and if there is a cond token in the code these would execute.In the test cases there wasn’t ant cond expression so these functions were tested with my own inputs and they work correcly:

A picture containing text, font, screenshot, white

Description automatically generated

A screen shot of a computer program

Description automatically generated with low confidence

10.) For the next grammar piece we have IfExpression and EndExpression these work mutulally in the code so here is the represantations:

A picture containing text, font, screenshot, white

Description automatically generated

A picture containing text, screenshot, software

Description automatically generated

11.)Finally there is the BeginExpression which didn’t happen to be in the test input so I tested this too with my own inputs this grammar piece’s representation in java works correctly.



A picture containing text, screenshot, font

Description automatically generated

Here is the explanation what happens when the code runs:

In this piece of code arr holds the tokens and it’s size is 200. Position values iterates through the arr so our parser knows which token it takes from arr. Count variable holds how much indentation needed for console output. Finally calculate funtion calculates the how much indentation needed throughout the code it takes count as parameter.

A screen shot of a computer program

Description automatically generated with low confidence

Here is the main function:

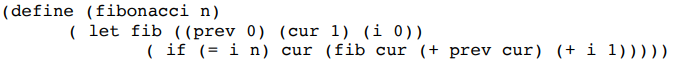
A picture containing text, screenshot, font

Description automatically generated

This main function takes the input file as inputFile and then places them one by one to the ArrayList arr created earlier.

Then with the last 2 lines the recursive descent parser runs correctly.

Finally here is the input file which I created using scanner Project using this input:



A screen shot of a computer

Description automatically generated with medium confidence

Last Notes:

* In the Project I used the input file as I stated before and it came from the scanner but when given complex inputs our scanner Project doesn’t always gives the correct output so I suggest using this Project with the %100 percent correct output from scanner.
* Also in this Project we couln’t make the syntax error part. So our code when run with the incorrect input only gives error and prints the incorrect place that it occured but doesn’t specifies what would come in place of the incorrect par tor the index where it occured. Also it doesn’t specify the identifiers actual values.
* When making the Project I (Akın Ateş) wrote the report and also coded everything. Because of my team members’s health issues, They helped me debug the code.
* My last comments about this Project and the prior one is that we build a complete compiler from scratch. I it was very interesting to work on how the compilers evalute our code and give errors.