MARMARA UNIVERSITY
FACULTY OF ENGINEERING

CSE2260 - Principles of Programming Languages

Project 1_2

Due Date: 15.05.2023

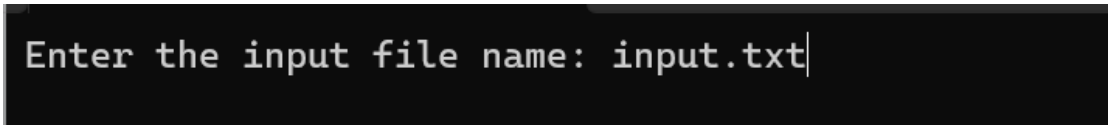
	Department	Student Id	Name Surname
1	CSE	150120027	Mert Muslu
2	CSE	150120051	Erkut Dönmez
3	CSE	150121539	Gülsüm Ece Günay

PROJECT-2

This project aims to analyze a recursive descent parser. That is a top-down parsing technique that is used to analyze grammar.

We create a different function to properly analyze correct grammar according to the production rule given in the project file. Some functions are taken from Project 1 which are the functions to detect tokens, and necessary additions for the taken functions have been made. Additionally, we have used the C programming language.

Our code starts by asking for the name of the input file.



Then according to data that stored in .txt file, program works and examines the grammar of text with the help of recursive functions that we used.

a.) Completed Parts

As it is wished program prints out the necessary output with the appropriate spaces. As far as we can we applied the rules on the recursive descent parser table, and haven't encountered with a major error. As long as the input is not too much complicated, the output gives a correct answer.

b.) Incompleted Parts

Since we haven't encountered with a major error, as an incompleted part we may say that, when it prints out a syntax error if the given input's parentheses numbers doesn't match, it is wished to warn the user with an error message such as:
SYNTAX ERROR [7:18]: 'identifier' is expected.
However, we couldn't find a way to calculate the number of rows and columns.
So, instead of we've printed out something like that:
SYNTAX ERROR [?:?]: ')' is expected.
And as it is seen our program still warns the user about the error but it doesn't give an info about which row and column error has occurred.

Example-1:

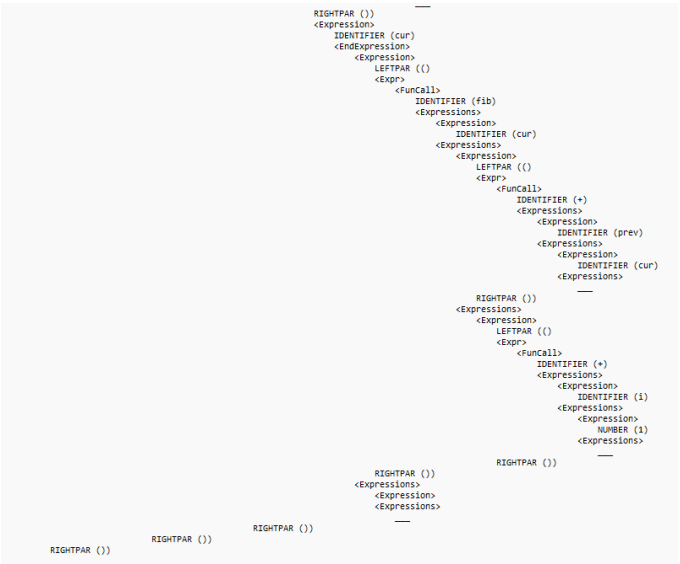
For the first example, the input and output file is given below.

input file:

```
(define (fibonacci n)
   ( let fib ((prev 0) (cur 1) (i 0))
         ( if (= i n) cur (fib cur (+ prev cur) (+ i 1)))))
```

output file:

```
<Program>
    <TopLevelForm>
        LEFTPAR (()
        <SecondLevelForm>
            <Definition>
                DEFINE (define)
                <DefinitionRight>
                    LEFTPAR (()
                    IDENTIFIER (fibonacci)
                    <ArgList>
                        IDENTIFIER (n)
                        <Arglist>
                            ——
                    RIGHTPAR ())
                    <Statements>
                        <Expression>
                            LEFTPAR (()
                            <Expr>
                                <LetExpression>
                                    LET (let)
                                    <LetExpr>
                                        IDENTIFIER (fib)
                                        LEFTPAR (()
                                        <VarDefs>
                                            LEFTPAR (()
                                            IDENTIFIER (prev)
                                            <Expression>
                                                NUMBER (0)
                                            RIGHTPAR ())
                                            <VarDef>
                                                <VarDefs>
                                                    LEFTPAR (()
                                                    IDENTIFIER (cur)
                                                    <Expression>
                                                        NUMBER (1)
                                                    RIGHTPAR ())
                                                    <VarDef>
                                                        <VarDefs>
                                                            LEFTPAR (()
                                                            IDENTIFIER (i)
                                                            <Expression>
                                                                NUMBER (0)
                                                            RIGHTPAR ())
                                                            <VarDef>
                                                                ——
                                        RIGHTPAR ())
                                        <Statements>
                                            <Expression>
                                                LEFTPAR (()
                                                <Expr>
                                                    <IfExpression>
                                                        IF (if)
                                                        <Expression>
                                                            LEFTPAR (()
                                                            <Expr>
                                                                <FunCall>
                                                                    IDENTIFIER (=)
                                                                    <Expressions>
                                                                        <Expression>
                                                                            IDENTIFIER (i)
                                                                        <Expressions>
                                                                            <Expression>
                                                                                IDENTIFIER (n)
                                                                            <Expressions>
                                                                                ——
                                                            RIGHTPAR ())
                                                            <Expression>
                                                                IDENTIFIER (cur)
                                                                <EndExpression>
                                                                    <Expression>
                                                                        LEFTPAR (()
                                                                        <Expr>
                                                                            <FunCall>
                                                                                IDENTIFIER (fib)
                                                                                <Expressions>
                                                                                    <Expression>
                                                                                        IDENTIFIER (cur)
                                                                                    <Expressions>
                                                                                        <Expression>
                                                                                            LEFTPAR (()
                                                                                            <Expr>
                                                                                                <FunCall>
                                                                                                    IDENTIFIER (+)
                                                                                                    <Expressions>
                                                                                                        <Expression>
                                                                                                            IDENTIFIER (prev)
                                                                                                        <Expressions>
                                                                                                            <Expression>
                                                                                                                IDENTIFIER (cur)
                                                                                                            <Expressions>
                                                                                                                ——
                                                                                RIGHTPAR ())
```



Example-2

input file:

```
(define (fibonacci n)
        ( let fib ((prev 0) (cur 1) (i 0))
              ( if (= i n) cur (fib cur (+ prev cur) (+ i 1)))
               define junk n)))
```

output file:

As it is indicated above, we can not calculate the location of the error, however, what is absent has been given.

```
SYNTAX ERROR [?:?]: ')' is expected. Lack of Right Parentheses
```

Luckily, the remaining part gives the correct answer.

```
<Program>
    <TopLevelForm>
        LEFTPAR (()
        <SecondLevelForm>
            <Definition>
                DEFINE (define)
                <DefinitionRight>
                    LEFTPAR (()
                    IDENTIFIER (fibonacci)
                    <ArgList>
                        IDENTIFIER (n)
                        <ArgList>
                                    ___
                    RIGHTPAR ())
                    <Statements>
                        <Expression>
                            LEFTPAR (()
                            <Expr>
                                <LetExpression>
                                    LET (let)
                                    <LetExpr>
                                        IDENTIFIER (fib)
                                        LEFTPAR (()
                                        <VarDefs>
                                            LEFTPAR (()
                                            IDENTIFIER (prev)
                                            <Expression>
                                                NUMBER (0)
                                            RIGHTPAR ())
                                            <VarDef>
                                                <VarDefs>
                                                    LEFTPAR (()
                                                    IDENTIFIER (cur)
                                                    <Expression>
                                                        NUMBER (1)
                                                    RIGHTPAR ())
                                                    <VarDef>
                                                        <VarDefs>
                                                            LEFTPAR (()
                                                            IDENTIFIER (i)
                                                            <Expression>
                                                                NUMBER (0)
                                                            RIGHTPAR ())
                                                            <VarDef>
                                                                ___
                                        RIGHTPAR ())
                                        <Statements>
                                            <Expression>
                                                LEFTPAR (()
                                                <Expr>
                                                    <IfExpression>
                                                        IF (if)
                                                        <Expression>
                                                            LEFTPAR (()
                                                            <Expr>
                                                                <FunCall>
                                                                    IDENTIFIER (=)
                                                                    <Expressions>
                                                                        <Expression>
                                                                            IDENTIFIER (i)
                                                                        <Expressions>
                                                                            <Expression>
                                                                                IDENTIFIER (n)
                                                                            <Expressions>
                                                                                ___
                                                            RIGHTPAR ())
                                                            <Expression>
                                                                ___
                                                    RIGHTPAR ())
                                                    <Expressions>
                                                        <Expression>
                                                            IDENTIFIER (cur)
                                                        <Expressions>
                                                            <Expression>
                                                                LEFTPAR (()
                                                                <Expr>
                                                                    <FunCall>
                                                                        IDENTIFIER (fib)
                                                                        <Expressions>
                                                                            <Expression>
                                                                                IDENTIFIER (cur)
                                                                            <Expressions>
                                                                                <Expression>
                                                                                    LEFTPAR (()
                                                                                    <Expr>
                                                                                        <FunCall>
                                                                                            IDENTIFIER (+)
                                                                                            <Expressions>
                                                                                                <Expression>
                                                                                                    IDENTIFIER (prev)
                                                                                                <Expressions>
                                                                                                    <Expression>
                                                                                                        IDENTIFIER (cur
                                                                                                    <Expressions>
                                                                                                        ___
                                                                                    RIGHTPAR ())
                                                                                <Expressions>
                                                                                    <Expression>
                                                                                        LEFTPAR (()
                                                                                        <Expr>
                                                                                            <FunCall>
                                                                                                IDENTIFIER (+)
                                                                                                <Expressions>
                                                                                                    <Expression>
                                                                                                        IDENTIFIER (i)
                                                                                                    <Expressions>
                                                                                                        <Expression>
                                                                                                            NUMBER (1)
                                                                                                        <Expressions>
                                                                                                            ___
                                                                                    RIGHTPAR ())
                                                                        RIGHTPAR ())
                                                                    <Expressions>
                                                                        <Expression>
                                                                            LEFTPAR (()
                                                                            <Expr>
                                                                                <FunCall>
                                                                                    IDENTIFIER (define)
                                                                                    <Expressions>
                                                                                        <Expression>
                                                                                            IDENTIFIER (junk)
                                                                                        <Expressions>
                                                                                            ___
                                                                RIGHTPAR ())
                                                            <Expressions>
                                                                ___
SYNTAX ERROR [?:?]: ')' is expected. Lack of Right Parentheses
```