BBM 301 PROJECT REPORT

- 21627782 Kerem ÜREMEZ
- 21627026 Kaan Nuri BEYHAN
- 21627744 Neşe Gamze UYANIK

Due date : 17.12.2019

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Informations

In this project we are developing a programming language to use Geographical Maps and Satellite data (GPS) easily. Our programming language will be used in programming a navigation system and the system shows 3D buildings and structures and allows users to monitor traffic speeds. Graph structure provide define roads and crossroads. Thus, thanks to the features provided by the language and the graphical structure we define, a GPS application can be easily written to enable the users to find the most suitable route to the specified destination.

1.2 Parser

Makefile:

```
KNK_parser: lex.yy.c y.tab.c
	gcc -o KNK_parser y.tab.c -lfl

y.tab.c: project.y
	yacc -d project.y

lex.yy.c: project.l
	lex project.l

clean:
	rm -rf lex.yy.c y.tab.c y.tab.h parser
```

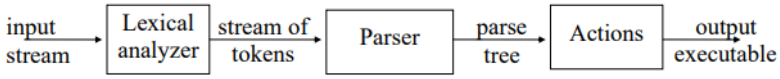
We can use this parser two ways :

- *make*
- *./KNK_parser test.txt*

- *lex project.l*
- *yacc -d project.y*
- *gcc -o KNK_parser y.tab.c -lfl*
- *./KNK_parser test.txt*

2. Lexical Analyzer

2.1 Lex Definition



Scans the input stream and converts sequences of characters into tokens. Reads a specification file containing regular expressions and generates a C routine that performs lexical analysis. Matches sequences that identify tokens.

2.2 Defined Token List

First we defined tokens in lex then we use them in yacc with @token specifier.

Our BUILTIN Methods :

- BLTIN_ADD
- BLTIN_REMOVE
- BLTIN_CALCULATE_ROTATE
- BLTIN_CONNECT
- BLTIN_3D
- BLTIN_PRINT
- BLTIN_SHOWONMAP
- BLTIN_SEARCHLOCATION
- BLTIN_GETROADSPEED
- BLTIN_GETLOCATION
- BLTIN_SHOWTARGET

And we have different data types from C language. We have defined these types to make the desired operations easier:

- ROAD
- CROSSROAD
- LONGTITUDE
- LATITUDE
- IMAGE

And we defined simple definitons and operations like C language.

3. Yacc

3.1 BNF Grammer Of Our Language

Our grammer is start from **statement_list**

And it can call recursively itself and statement or statement

```
statement_list:   statement
                | statement_list statement
                ;
```

In statement you can do anything defined in grammar

```
statement :   if_structure
            | for_structure
            | while_structure
            | declarations SEMICOLON
            | function_call SEMICOLON
            | bltin_function_call SEMICOLON
            | BREAK SEMICOLON
            | CONTINUE SEMICOLON
            | RETURN SEMICOLON
            | RETURN argument_list SEMICOLON
            | function
            | calculations SEMICOLON
            | struct_definition SEMICOLON
            | struct_object_assignment SEMICOLON
            ;
```

This is how you define functions

```
function :    FUNCTION IDNTF LEFT_PARANTHESIS parameter_list
RIGHT_PARANTHESIS returns_types  block
            | FUNCTION IDNTF LEFT_PARANTHESIS RIGHT_PARANTHESIS returns_types
block  ;
```

You can define a block inside like this

```
block :   LEFT_BRACKET statement_list RIGHT_BRACKET
        | LEFT_BRACKET RIGHT_BRACKET
        ;
```

You can call functions as follows

```
function_call :  IDNTF LEFT_PARANTHESIS argument_list RIGHT_PARANTHESIS
                       |IDNTF LEFT_PARANTHESIS RIGHT_PARANTHESIS
                       ;
```

And our data types like

```
parameter_elements : VOID
                   | STRING
                   | CHAR
                   | INT
                   | FLOAT
                   | BOOL
                   | LONGITUDE
                   | LATITUDE
                   | ROAD
                   | ARRAY
                   ;
```

And other simple definitions made.

3.2 Special Data-Types

We have 3 different data type

- ARRAY DATA TYPE
- GRAPH DATA TYPE
- STRUCT DATA TYPE

3.3 Built-in Functions Of Our Language

For printing messages for user;

bltin_print: BLTIN_PRINT LEFT_PARANTHESIS argument_list RIGHT_PARANTHESIS

;

It gets longitude and latitude variables and then it creates a window after that print the map on the screen with given longitude and latitude

bltin_showonmap: BLTIN_SHOWONMAP LEFT_PARANTHESIS IDNTF COMMA IDNTF RIGHT_PARANTHESIS

| BLTIN_SHOWONMAP LEFT_PARANTHESIS INT_LTRL COMMA INT_LTRL RIGHT_PARANTHESIS

;

Getting address as parameter and if the address exists it returns true if not exists return false.

bltin_searchlocation: BLTIN_SEARCHLOCATION LEFT_PARANTHESIS IDNTF RIGHT_PARANTHESIS

| BLTIN_SEARCHLOCATION LEFT_PARANTHESIS STR_LTRL RIGHT_PARANTHESIS

;

It gets road vertices as edge and showing road speed

bltin_getroadspeed : BLTIN_GETROADSPEED LEFT_PARANTHESIS IDNTF RIGHT_PARANTHESIS

| BLTIN_GETROADSPEED LEFT_PARANTHESIS STR_LTRL RIGHT_PARANTHESIS

;

It gets struct object and shows current location

bltin_getlocation : BLTIN_GETLOCATION LEFT_PARANTHESIS IDNTF RIGHT_PARANTHESIS

;

It gets target address as parameter and show convenient roads to g oto target

bltin_showtarget : BLTIN_SHOWTARGET LEFT_PARANTHESIS IDNTF RIGHT_PARANTHESIS

| BLTIN_SHOWTARGET LEFT_PARANTHESIS STR_LTRL RIGHT_PARANTHESIS

;

It creates 3D object like bridges, wall etc. It gets vertices as parameter

bltin_threed_object_definition : BLTIN_3D LEFT_PARANTHESIS INT_LTRL COMMA INT_LTRL COMMA INT_LTRL RIGHT_PARANTHESIS

;

It gets target address as parameter and it shows convenient road you can choose

bltin_calculate_route: BLTIN_CALCULATE_ROUTE LEFT_PARANTHESIS IDNTF RIGHT_PARANTHESIS

;

It provides linking between other users. And it shows that users current location and calculate the distance between them

bltin_connect: BLTIN_CONNECT LEFT_PARANTHESIS IDNTF RIGHT_PARANTHESIS

;

It works like scanf in C Language. It get inputs from users

bltin_input : BLTIN_INPUT LEFT_PARANTHESIS RIGHT_PARANTHESIS

;

It provides adding new roads and crossroad to graph data type

bltin_add : IDNTF DOT ROAD DOT BLTIN_ADD LEFT_PARANTHESIS identifier RIGHT_PARANTHESIS

| IDNTF DOT CROSSROAD DOT BLTIN_ADD LEFT_PARANTHESIS identifier RIGHT_PARANTHESIS

;

It provides to remove roads and crossroad to graph data type

bltin_remove : IDNTF DOT ROAD DOT BLTIN_REMOVE LEFT_PARANTHESIS identifier RIGHT_PARANTHESIS

| IDNTF DOT CROSSROAD DOT BLTIN_REMOVE LEFT_PARANTHESIS identifier RIGHT_PARANTHESIS

;

3.4 Error Messages

If there is a unknown character in source file it prints syntax error in line : line number

```
{printf("\nSyntax error in line:%d\n",yylineno);yylval=1;}
```

```
[b21627744@rdev bbm301]$ ./KNK_parser test.txt

Syntax error in line:2
```

If there is a parse error it prints parse error in line : line number

```
int yyerror(char *s){
        printf("\nParse error in line:%d\n",yylineno);
        yylval=1;
}
```

```
[b21627744@rdev bbm301]$ ./KNK_parser test.txt

Parse error in line:3
[b21627744@rdev bbm301]$
```

4. Tutorial Of Our Language

```
function main ( ) void {
//write parameter list in parenthesis if you have
longitude  a = 39.896671 ;
latitude b = 32.733719 ;
showonmap ( a , b ) ;

// all other operations can be done here
return ;
}
```

References

- https://gist.github.com/codebrainz/2933703
- http://dinosaur.compilertools.net/