Lab Report

Topic: Compiler Project

Course No: CSE 3212

Course Title: Compiler Design Laboratory

Submitted To:

Dola Das
Lecturer
Department of Computer Science and Engineering
Khulna University of Engineering and Technology.

Md. Ahsan Habib Nayan
Lecturer
Department of Computer Science and Engineering
Khulna University of Engineering and Technology

Submitted By

Kaab islam
Roll: 1707084
Department of Computer Science and Engineering
Khulna University of Engineering and Technology

Submission Date : 15-06-21

Objectives: Through this project we will be able to learn about:

- •How a compiler works.
- •Know about flex and bison.
- •Difficulties of building a language.
- •How tokens are generated.
- •Learn through own defined language.

Introduction :

Compiler, computer software that translates (compiles) source code written in a high-level language (e.g., C++) into a set of machine-language instructions that can be understood by a digital computer's CPU. Compilers are very large programs, with error-checking and other abilities. Some compilers translate high-level language into an intermediate assembly language, which is then translated (assembled) into machine code by an assembly program or assembler. Other compilers generate machine language directly. The term *compiler* was coined by American computer scientist Grace Hopper, who designed one of the first compilers in the early 1950s.

Flex and Bison:

Flex and bison are tools designed for writers of compilers and interpreters, althoughthey are also useful for many applications that will interest noncompiler writers. Any application that looks for patterns in its input or has an input or command language is a good candidate for flex and bison. Furthermore, they allow for rapid application prototyping, easy modification, and simple maintenance of programs.Furthermore, they allow for rapid application prototyping, easy modification, and simple maintenance of programs.Few things people have used flex and bison, or their pred-ecessors lex and yacc, to develop:

- The desktop calculator bc

• The tools eqn and pic, typesetting preprocessors for mathematical equations and complex pictures

• Many other “domain-specific languages” targeted for a particular application

•PCC, the Portable C Compiler used with many Unix systems

• Flex itself

• A SQL database language translator.

Problem Statement:

In this lab, we were assigned to write a program to design a lexical analyzer of our own that will include the followings:

1.Data types and Variables

2.Key words

3.Operators

4.Macro preprocessor

5.Array Declarations

6.Series Mathematical and Bitwise operations

7.Loops

8.Functions

9.Conditional Statements

10.Struct and class declarations

11.try catch block

12.comments

We were also assigned to write the lex program to recognize our ID, Keyword, Operator and comments using a source file containing our designed lexeme.

Tokens: Following tokens are used in this project :

```
"DEFINE"            {return DEFINE;}
"CASE"              {return CASE;}
```

```
"SWITCHCASE"        {return SWITCH;}
"DEFAULTCASE"       {return DEFAULT;}
{digit}+            {yylval = atoi(yytext); return NUM; }
"PRINTFUNCTION"     {return PRINTFUNCTION; }
"IF"                {return IF;}
"ELSE"              {return ELSE; }
"{"                 {return BRACKETSTART;  }
"}"                {return BRACKETEND;    }
"FSPIN"            {return FOR;}
"WSPIN"            {return WHILE;}
"FACTORIAL"        {return FACTORIAL;}
"ISDIVISIBLE"      {return ISDIVISIBLE;}
"MOD"              {return MOD;}
"OR"               {return OR;}
"AND"              {return AND;}
"XOR"              {return XOR;}
"POW"              {return POW;}
"ISGREATER"        {return ISGREATER;}
"ISLESS"           {return ISLESS;}
"ISEQUAL"          {return ISEQUAL;}
"ODDEVEN"          {return ODDEVEN;}
"TAN"              {return TAN;}
"INTEGER"          {return INT;}
"BOOL"             {return BOOL;}
"CHAR"             {return CHAR;}
"FLOAT"            {return FLOAT;}
```

```
"ARRAY"            {return ARRAY;}

"TRY"              {return TRY;}

"CATCH"            {return CATCH;}

"FUNCTION"         {printf(" "); return FUNCTION;}

"STRUCT"           {return STRUCT;}

"CLASS"            {return CLASS;}

"PRIVATE"          {return PRIVATE;}

"PUBLIC"           {return PUBLIC;}

"STRCMP"           {return STRCMP;}

"MAIN"             {printf("\nMain Function Start\n"); return
MAIN; }

{comment}          {printf("\nSingle liness Comment found ::
");printf("\n");}

"IMPORT".*         {return HEADER;}

[-+/*<>=,():;%|&?^^]     {yylval = yytext[0];    return
*yytext;    }

{variable}     {printf("\n"); yylval = *yytext - 'a'; return
VAR; }

{variable}+    {yylval = atoi(yytext);printf("\nstring
found!\n");return STRING;}

[ ]*                      {}

[\n]*                     {}

[\t]*                     {}

.                         {printf("\nUnknown Syntax :
%s\n",yytext);}
```

Grammar:

Some grammar for this project is given below:

Grammar for For loops:

```
| FOR '(' NUM ',' NUM ',' NUM ')' BRACKETSTART  statement
BRACKETEND {Expression}

Example : FSPIN (2,18,1)
          {
               4+8;
          }
```

Grammar for WHILE loops:

```
| WHILE '(' NUM '<' NUM ')' BRACKETSTART statement BRACKETEND
{ Expression}

Example : WSPIN (2<5)
          {
               1+3;
          }
```

Grammar for SWITCH case:

```
| SWITCH '(' NUM ')' BRACKETSTART  SWITCHCASE BRACKETEND
{cases}
```

Grammar for function with no parameter:

```
| FUNCTION VAR '(' ')' BRACKETSTART statement BRACKETEND {
Expression}
```

Grammar for function with two parameter:

```
| FUNCTION VAR '(' expression ',' expression ')' BRACKETSTART
statement BRACKETEND {Expression}
```

Grammar for some built in functions:

ISDIVISIBLE: Check if first number is divisible by second one.

```
| ISDIVISIBLE '(' NUM ',' NUM ')' ';'
```

XOR: Return Bitwise XOR of two numbers.

| XOR '(' NUM ',' NUM ')' ';'

ISLESS:Check if two numbers are equal or not.

| ISEQUAL '(' NUM ',' NUM ')' ';'

POW:Return power operations of two numbers.

| POW '(' NUM ',' NUM ')' ';'

User Maunal

How a user can use different properties for this language is given below :

1)Data types and Variable Declarations:

There are four data types are available in this language.They are INT,FLOAT,BOOL and CHAR.to declare a variable is almost same as declaration of c language.To declare a INT variable we have to call Like this : INT a;then we assign value to a like this : a = 2;

2)Keywords:

Every language has some keywords.INT,CHAR,FSPIN,FWHILE,MOD,IF,ELSE are some of the keywords associated with this language.

3)Macro preprocessor:

preprocessor is a macro preprocessor (allows to define macros) that transforms program before it is compiled.Unlike c programs Macros can be defined inside the body of the main function.It can be defined in the following way:

-> DEFINE m 3;

4)Header:

In a language header files is needed to import something or defined template library etc.so header file could be defined in the following way:

-> IMPORT math;

5)Array Declarations:

Array is used as an important and most useful data structure in almost every languages.Arrays can be declared alongside with arrays datatypes and size of the array.Array could be defined in the following way:

ARRAY -> CHAR -> A(100);

here, 100 is the size of the array and A is the name of the array.

6)Loops:

Loops are considered as one of the most important properties of a language.loops can be Different types.There are two types of loops used in this language,FSPIN and WSPIN.

i.FSPIN: It works almost similar as for loop is c language.There will be 3 variables.One for initialization,one for testExpression and one for update.It's syntax will be:

```
FSPIN(initializationStatement;testExpression;updateStatement){
    Expression;
    }
```

ii.WSPIN: It works similar as while loop.while conditions are true loops will be executed.It's syntax will be:

```
WSPIN(conditons){
    Expression;
}
```

7)Conditional Statements:

Conditional statements are used to decided something in a language.There are Three types of conditions available in this language.One is just a block of if and else and another is Nested if else.Another is conditional statement using ternary operator "??". so first one is defined as follows:

```
IF(conditions){
     Expression;
}
ELSE{
     Expression;
}
```

Nested if else can be defined as follows:

```
IF(conditions){
     IF(conditions){
          Expression;
     }
     ELSE{
          Expression;
     }
}
ELSE{
     IF(conditions){
          Expression;
     }
     ELSE{
          Expression;
     }
}
```

Conditional statement using “??” has two side of it.in left side of ?? it will be given the conditions and right side has two conditions about if conditions are true or false.It’s syntax will be as follows:

```
(conditions) ?? expressionOne expressionTwo
```

expressionOne will be executed if statements are true and expressionTwo will be executed if false.

8)Functions: Functions are important for a language to make the code more readable,make work simple.There is a opportunity to

use three type of function in this language : without parameter,with one parameter and with two parameters.It's declarations will be as follows:

Type - 1:Without parameter:

```
FUNCTION functionName(){
    Expression;
}
```

Type - 2:With One parameter:

```
FUNCTION functionName(parameter one){
    Expression;
}
```

Type - 3:With two parameter:

```
FUNCTION functionName(parameter one,parameter two){
    Expression;
}
```

9)Struct Declarations:

A structure is a user-defined data type in this language.It is used to contain different types of datatypes into a single types.It's declarations will be as follows:

```
STRUCT structName{
    Expressions;
}
```

10)Class Declarations:

Class is also used as an user defined data types which allows to work with various modes.In this language there has been four opportunity to declare a class.

i)Default Declaration: In default declaration class will be declared as a PUBLIC class which allows user to access it's

attribute from anywhere in the program.Whenever modes is not declared it will considered as a PUBLIC class.It's syntax:

```
CLASS className{
    Expressions;
}
```

ii)explicitly declared as PUBLIC: Here it'll be announced it as a PUBLIC Class.it's syntax:

```
CLASS className:PUBLIC{
    Expressions;
}
```

iii)explicitly declared as PRIVATE: Through this declarations attributes can not access from everywhere in the program.it's syntax:

```
CLASS className:PRIVATE{
    Expressions;
}
```

iv)Inheritance CLASS declarations: Inheritance is an important properties for an object oriented languages.This language has this opportunity.To declarations a inheritance class no need to declare it's mode as it will be the same mode as it's parent class.It's syntax:

```
CLASS className:parentClassName{
    Expressions;
}
```

11)Switchcase:

In switchcase of this language there is a expressions which has to evaluated with the cases.If it didn't match with any of the cases then it will go to default classes which is in the end of the switchcase.It's syntax is as below:

```
SWITCHCASE (expressions1)
{
CASE 1: expressions;
```

```
    CASE 2: expressions;

    CASE 3: expressions;
    .
    .
    CASE n: expressions;

    DEFAULTCASE : expressions;

    }
```

12)Built in function:A function which is already defined in a program or programming framework with a set of statements, which together performs a task and it is called Build-in function. So users need not create this type of function and can use directly in their program or application.Built in functions makes a language more advance.There has been some built in functions named POW,MOD,OR,ISLESS,FACTORIAL,ODDEVEN etc in this language.ODDEVEN and FACTORIAL functions need one parameters and rest of the built in functions need two parameters.For example calculate XOR of two numbers just pass these numbers like this : XOR(1,2) and it will return a result equal to 3 which is the XOR value of 1 and 2.To get factorial pass the number n like this FACTORIAL(5) and it will return 120.Other functions also works like the above two examples.

13)Comments: Comments are useful to make code readable.Single Line comments are allowed in this language.To make a comment '!' is used first.It's syntax :

```
!Comment
```

Conclusion: Compilers are consist of many functionalities.There has been some drawbacks in good

languages also.Despite all this things this languages tries to made things easier and user friendly to it's user.

Reference:

i. flex & bison by John R. Levine

ii. https://www.britannica.com/technology/compiler