Project Report

on

Compiler for Comparison

Developed by

Harsh Mayavanshi(IT067)

Pranay Makwana(IT069)

Guided By:

Prof. Nikita P. Desai

Dept. of Information Technology

DEPARTMENT OF INFORMATION TECHNOLOGY

FACULTY OF TECHNOLOGY,

DHARMSINH DESAI UNIVERSITY

COLLEGE ROAD, NADIAD- 387001

DHARMSINH DESAI UNIVERSITY
NADIAD-387001, GUJARAT

CERTIFICATE

This is to certify that the project entitled "**Compiler for Comparison"** is a bonafide report of the work carried out by

1) Mr. Harsh Mayavanshi, Student ID No: 18ITUSS096

2) Mr. Pranay Makwana, Student ID No: 18ITUBS027

of Department of Information Technology, semester VI, under the guidance and supervision for the award of the degree of Bachelor of Technology at Dharmsinh Desai University, Nadiad (Gujarat). They were involved in Project in subject of "**Language Translator**" during academic year 2020-2021.

Prof. N.P. Desai
(Lab Incharge)
Department of Information Technology,
Faculty of Technology,
Dharmsinh Desai University, Nadiad
Date:

Prof. (Dr.)V K Dabhi,
Head , Department of Information Technology,
Faculty of Technology,
Dharmsinh Desai University, Nadiad
Date:

Index

1.0 INTRODUCTION

1.0.1 Project Details

Grammar Name: Comparison program.

Grammar Rules:

- **Valid sentences in language :**
    1. Which is bigger number 7 or 2?
    2. From 7 and 2 which is larger ?
    3. Is 7 bigger than 2?
    4. Is 2 smaller than 4?
    5. Which is smaller between 4 and 9?
    6. Which is smaller number 4 or 2?

Regular Expression:

Keyword	"Which"|"WHICH"|"which"|"Is"|"IS"|"is"|"From"|"FROM" |"from"|"number"|"or"|"and"|"between"|"than"
Comparison	"larger"|"bigger"|"greater"|"smaller"
Number	[0-9]+
Float	[0-9]+(\.[0-9]+)?(E[+\-]?[0-9]+)?
EndOfLine	"?"
Space	[\t\n]+

Token Table:

LEXEMES	TOKEN NAME	ATTRIBUTE VALUE
int	INTEGER	Point to table entry
float	FLOAT	Point to table entry
string	OPERATOR	Point to table entry
?	EOS	--
"Which"|"WHICH"|"which"|"Is"|"IS"|"is"|"From"|"FROM"|"from"|"number"|"or"|"and"|"between"|"than"	Keyword	--

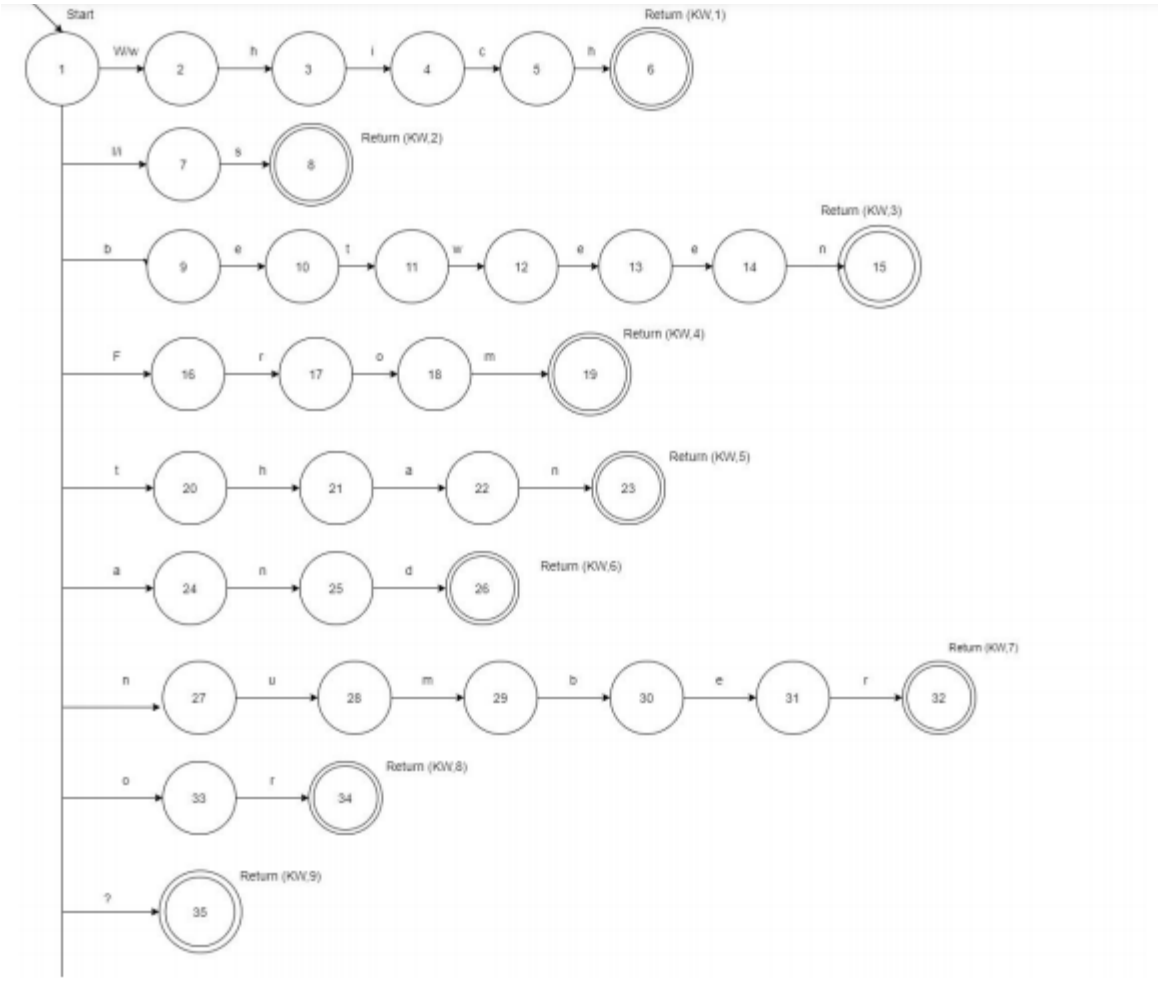
1.0.2 Project Planning

List of Students with their Roles/Responsibilities:

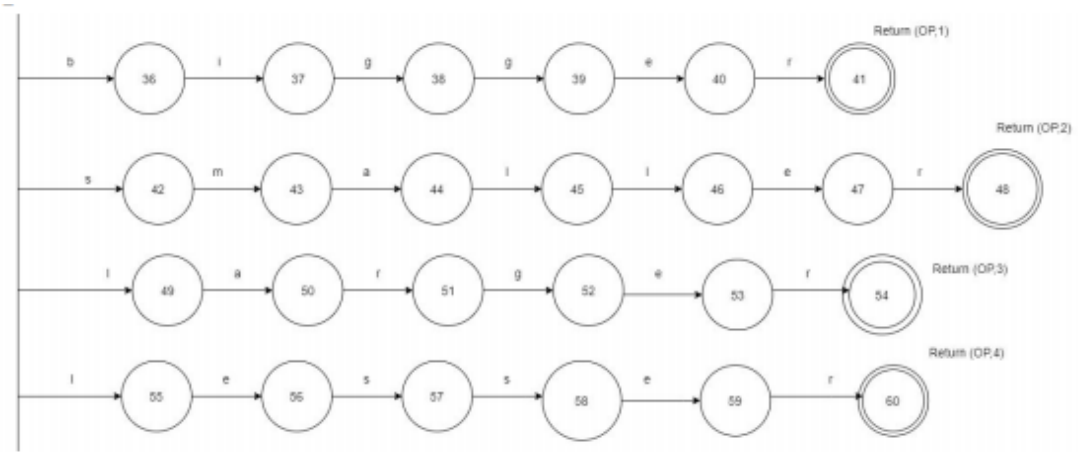
1. **Harsh Mayavanshi(IT067) -** Grammar Rules, Regular Expression, Final Code Verification, Final Report.
2. **Pranay Makwana(IT069) -** DFA Design, Algorithm Design, Scanner Implementation, Final Code Verification

2.0 LEXICAL PHASE DESIGN

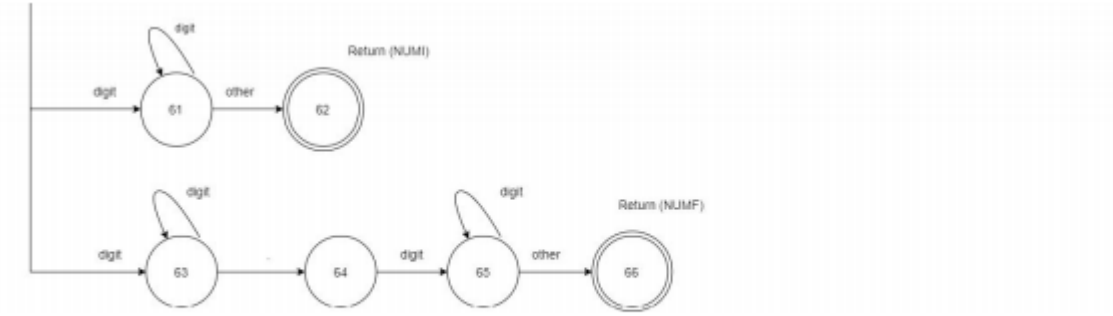
2.0.1 Deterministic Finite Automaton design for lexer



Opertion Transitions :-



Number Transitions:-



2.0.2 Algorithm of lexer

Pseudo code :

```
state = 0;
while(!EOF)
{
        switch(state)
        {
                case 0:
                        input(ch);
                        if(isWhich(ch) or isWHICH(ch))
                                state = 1;
                        else if(isFrom(ch) or isFROM(ch))
                                state = 8;
                        else if(isIS(ch) or isIs(ch))
                                state = 15;
                        else
                                return false;

                Case 1:
                        input(ch);
                        if(isis(ch))
                                state = 2;

                Case 2:
                        input(ch);
                        if(iscomparison(ch))
                                state = 3;

                Case 3:
                        input(ch);
                        if(isInt(ch) or IsFloat(ch))
                                state = 4;

                Case 4:
                        input(ch);
                        if(isor(ch))
                                state = 5;
```

```
Case 5:
        input(ch);
        if(isInt(ch) or isFloat(ch))
                state = 6;

Case 6:
        input(ch);
        if(isEndOfLine(ch))
                state = 7;

Case 7:
        return true;

Case 8:
        input(ch);
        if(isInt(ch) or isFloat(ch))
                state = 9;

Case 9:
        input(ch);
        if(isand(ch))
                state = 10;

Case 10:
        input(ch);
        if(isInt(ch) or isFloat(ch))
                state = 11;

Case 11:
        input(ch);
        if(iswhich(ch))
                state =12;

Case 12:
        input(ch);
        if(isis(ch))
                state = 13;

Case 13:
        input(ch);
```

```
                if(iscomparison(ch))
                        state = 14;

        Case 14:
                input(ch);
                if(isEndOfLine(ch))
                        state = 7;

        Case 15:
                input(ch);
                if(isInt(ch) or isFloat(ch))
                        state = 16;

        Case 16:
                input(ch);
                if(iscomparison(ch))
                        state = 17;

        Case 17:
                input(ch);
                if(isthan(ch))
                        state = 18;

        Case 18:
                input(ch);
                if(isInt(ch) or isFloat(ch))
                        state = 19;

        Case 19:
                input(ch);
                if(isEndOfLine(ch))
                        state = 7;

        default :
                return false;

    }
}
```

2.0.3 Implementation of lexer

```
%{
#include<stdio.h>
int count=0;
%}
Keyword
"Which"|"WHICH"|"which"|"Is"|"IS"|"is"|"From"|"FROM"|”from"|"numbe
r"|"or"|"and"|"between"|"tha
n"
Comparison "larger"|"bigger"|"greater"|"smaller"
Number [0-9]+
Float [0-9]+(\.[0-9]+)?(E[+\-]?[0-9]+)?
EndOfLine "?"
Space [\t\n]+
%%
{Keyword} {printf("<Keyword : %s>\n",yytext);count++;}
{Comparison} {printf("<Comparison : %s>\n",yytext);count++;}
{Number} {printf("<Number : %s>\n",yytext);count++;}
{Float} {printf("<Float No : %s>\n",yytext);count++;}
{EndOfLine} {printf("\n");}
{Space} {}
. {}
%%
int main()
{
```

```
yylex();
printf("Total number of tokens generated: %d\n",count);
return 0;
}
int yywrap(){return 1;}
```

2.0.4 Execution environment setup

Step by Step Guide to Install FLEX and Run FLEX

Program: Operating system: Ubuntu 20.04.2 LTS

Step 1: Installing Flex

Command to install flex: sudo apt-install flex

Step 2: Saving Flex file

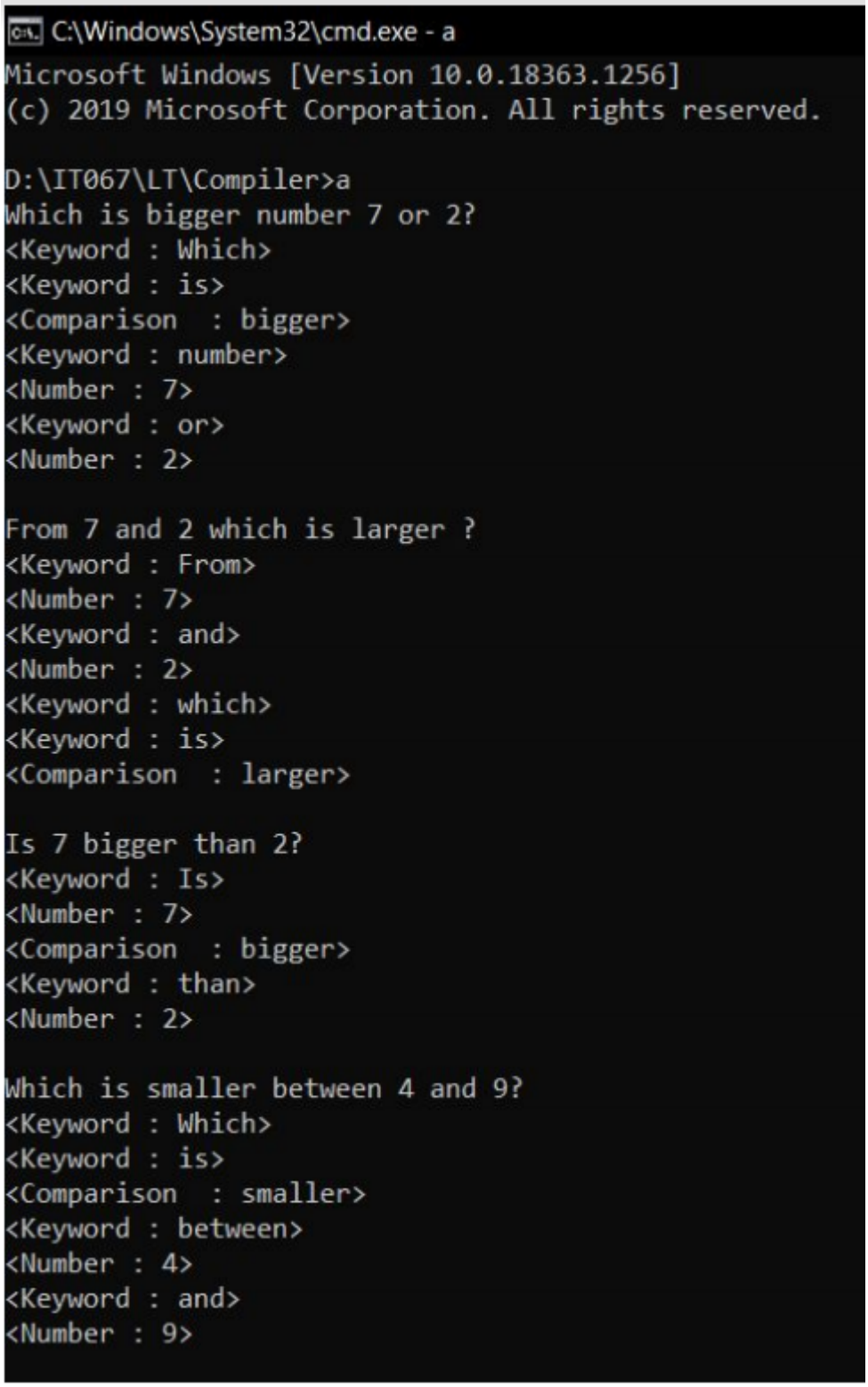
- Writing the source code in a file and saving it with " .l " extension at any desired directory / location in the system.

Step 3: Running the Flex code

- To run the flex code run following commands:

1. flex <filename>.l
2. gcc lex.yy.c -o <exeName>
3. ./<exeName>

2.0.5 Output screenshots of lexer



3.0 SYNTAX ANALYZER DESIGN

3.0.1 Grammar rules

```
program : program start{}
        |
        ;
start : 'w"h"i"c"h"i"s'OPERATOR'n"u"m"b"e"r'INTEGER SEP INTEGER
EOS'\n'
                |
        'W"h"i"c"h"i"s'OPERATOR'n"u"m"b"e"r'INTEGER SEP INTEGER
EOS'\n'
                |
        'W"h"i"c"h"i"s'OPERATOR'b"e"t"w"e"e"n'INTEGER SEP INTEGER
EOS'\n'
                |
        'w"h"i"c"h"i"s'OPERATOR'b"e"t"w"e"e"n'INTEGER SEP INTEGER
EOS'\n'
                |
        'F"r"o"m'INTEGER SEP INTEGER 'w"h"i"c"h"i"s' OPERATOR EOS'\n'
                |
        'f"r"o"m'INTEGER SEP INTEGER 'w"h"i"c"h"i"s' OPERATOR EOS'\n'
                |
        SEP INTEGER OPERATOR 't"h"a"n' INTEGER EOS'\n'
        ;
```

3.0.2 Yacc Implementation

Flex Code:

```
%{
#include<stdio.h>
#include <stdlib.h>
#include "y.tab.h"
int yyerror (char *errormsg);
int yyparse();
int t=0;
%}
%%
[0-9]+                                          { yylval.num = atoi(yytext); return INTEGER; }
"exit"                                          { return EXIT; }
[WhichwhichisbetweenFromfromthannumber]         { return *yytext; }

"bigger"|"Bigger"|"larger"|"Larger"|"smaller"   { yylval.str = strdup(yytext); return OPERATOR;
}
"or"|"and"|"Is"                                 { yylval.str = strdup(yytext); return SEP; }
"?"                                             { return EOS; }
[!\n]                                           { return *yytext; }
[ |\t]                                          { /* Ignoring Whitespace */ }
.                                               { yyerror("Character Not Recognized\n"); }
%%
int yywrap() {return 1;}
int main ()
{
printf("ENTER INPUT:-\n");
yyparse();
return 0;
}
int yyerror (char *errormsg)
{
fprintf (stderr, "%s\n", errormsg);
exit(1);
}

void compare(int num1,int num2,char *oper,char *sep)
{
  if( strcmp(oper,"larger")==0 || strcmp(oper, "bigger")==0 || strcmp(oper , "greater")==0)
        {

        if(num1>num2)
        {
                if(strcmp(sep,"Is")==0)
                printf("Yes \n");
```

```
                else
                printf("%d \n",num1);
        }
        else if(num1<num2)
        {
                if(strcmp(sep,"Is")==0)
                printf("No \n");
                else
                printf("%d \n",num2);
        }
        else
        {
                printf("Both are equal \n");
        }


        }
        else if(strcmp(oper,"smaller")==0)
        {

        if(num1<num2)
        {
                if(strcmp(sep,"Is")==0)
                printf("Yes \n");
                else
                printf("%d \n",num1);
        }
        else if(num1>num2)
        {
                if(strcmp(sep,"Is")==0)
                printf("No \n");
                else
                printf("%d \n",num2);
                }
        else
        {
                printf("Both are equal\n");
        }

        }


}
```

YACC Code:

```
%{
#include <stdio.h>
#include <stdlib.h>
int yylex(void);
int yyerror(const char *s);
void compare(int num1, int num2, char *oper,char *sep);
%}
%union {int num; char *str;}
%token <num> INTEGER
%token <str> OPERATOR
%token <str> SEP
%token EOS
%token EXIT
%%
program : program start{}
        |
        ;
start : 'w''h''i''c''h''i''s'OPERATOR'n''u''m''b''e''r'INTEGER SEP INTEGER EOS'\n'
        {compare($15,$17,$8,$16);}
        |
        'W''h''i''c''h''i''s'OPERATOR'n''u''m''b''e''r'INTEGER SEP INTEGER EOS'\n'
        {compare($15,$17,$8,$16);}
        |
        'W''h''i''c''h''i''s'OPERATOR'b''e''t''w''e''e''n'INTEGER SEP INTEGER EOS'\n'
        {compare($16,$18,$8,$17);}
        |
        'w''h''i''c''h''i''s'OPERATOR'b''e''t''w''e''e''n'INTEGER SEP INTEGER EOS'\n'
        {compare($16,$18,$8,$17);}
        |
        'F''r''o''m'INTEGER SEP INTEGER 'w''h''i''c''h''i''s' OPERATOR EOS'\n'
        {compare($5,$7,$15,$6);}
        |
        'f''r''o''m'INTEGER SEP INTEGER 'w''h''i''c''h''i''s' OPERATOR EOS'\n'
        {compare($5,$7,$15,$6);}
        |
        SEP INTEGER OPERATOR 't''h''a''n' INTEGER EOS'\n'
        {compare($2,$8,$3,$1);}
        |
        EXIT{exit(0);};
```

%%

3.0.3 Execution environment setup

3.0.3 Execution environment setup

Step 1: Installing YACC/Bison

Command to install yacc/bison: sudo apt-install bison

Step 2: Saving Flex file

Writing the source code in a file and saving it with " .y " extension at any desired directory / location in the system.

Step 3: Running the Flex code

Run following commands:

1. flex <filename>.l
2. bison -dy <filename>.y
3. gcc lex.yy.c y.tab.c -o <exeName> -ll
4. ./<exeName>

3.0.4 Output screenshots of YACC based implementation Valid Input:

Valid input:

```
[edith@Jarvis IT067-Compiler]$ flex compiler.l
[edith@Jarvis IT067-Compiler]$ bison -dy compiler.y
[edith@Jarvis IT067-Compiler]$ gcc lex.yy.c y.tab.c -o compiler -lm
[edith@Jarvis IT067-Compiler]$ ./compiler
ENTER INPUT:-
Which is bigger number 7 or 2?
7
From 7 and 2 which is larger?
7
Is 7 bigger than 2?
Yes
Is 2 smaller than 4?
Yes
Which is smaller between 4 and 9?
4
which is smaller number 4 or 2?
2
exit
[edith@Jarvis IT067-Compiler]$
```

```
Which is bigger number 7 or 7?
Both are equal
Is 7 bigger than 7?
Both are equal
exit
[edith@Jarvis IT067-Compiler]$
```

Invalid Output:

```
ENTER INPUT:-
which is super number 7 or 2?
syntax error
[edith@Jarvis IT067-Compiler]$ ./compiler
ENTER INPUT:-
how is number 7 than 2?
syntax error
[edith@Jarvis IT067-Compiler]$ ./compiler
ENTER INPUT:-
7 or 2?
syntax error
[edith@Jarvis IT067-Compiler]$ ./compiler
ENTER INPUT:-
7.7.7 is bigger than 2?
syntax error
```

4.0 CONCLUSION

This project has been implemented by applying the concepts learned in our college curriculum in addition to many rich resources from the web. This project gave us an opportunity to have a better understanding of the theoretical concepts learned in the subject by implementing them.

We would like to thank Prof. Nikita P. Desai for encouraging us to implement this project. This project has indeed helped us improve our problem solving and team work skills and has given us a better insight of the inner workings of a compiler.