**­**

|  |  |
| --- | --- |
| **Total Marks:** | **04** |
| **Obtained Marks:** |  |

**Compiler Construction**

**Assignment # 02**

**Last date of Submission: 24 Mar 2025**

**Submitted To: Safi Ullah**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Student Name: Ubaid Bin Waris**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Reg Number: 2212416**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

***Instructions****: Copied or shown assignments will be marked zero. Late submissions are not entertained in any case.*

**Assignment:**

Develop a program using LEX/FLEX and YACC/BISON that recognizes strings following the grammar {aⁿbⁿ | n ≥ 0}. The program should correctly identify valid strings where the number of 'a's matches the number of 'b's and reject invalid inputs. Ensure proper lexical analysis and syntax parsing in your implementation.

Test Data: aabb, abb, aab, aaabbb

**Note:**

1. Change the filename to your ID, e.g. 2073105.doc
2. Upload the .doc on Google Classroom.
3. Poor indentation and wrong format will result in deduction of marks.
4. Maximum group members: 2

**Solution**

**Lexer.l**

%{

#include "parser.tab.h"

%}

%%

a { return A; }

b { return B; }

\n { return END; }

. { return 0; } // Ignore other characters

%%

int yywrap() {

return 1;

}

**Parser.y**

%{

#include <stdio.h>

#include <stdlib.h>

void yyerror(const char \*s) {

fprintf(stderr, "Error: %s\n", s);

}

int yylex(); // Declare the lexer function

%}

%token A B END

%%

input : sequence END { printf("Valid string\n"); }

;

sequence: A sequence B { $$ = $2 + 1; } // Ensures equal A and B counts

| A B { $$ = 1; } // Base case: "AB"

;

%%

int main() {

yyparse();

return 0;

}

**Output**

A screenshot of a computer screen

AI-generated content may be incorrect.