Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №5

По дисциплине: «Языковые процессы интеллектуальных систем»

# Тема: «Компилятор компиляторов»

Выполнил:

Студент 4 курса

Группы ИИ-21

Кирилович А.А.

Проверил:

Монтик Н.С.

Брест 2024

**Цель работы**: изучить и опробовать на практике генератор синтаксических анализаторов YACC.

17. Входной язык содержит многоуровневые арифметические выражения, разделенные символом; (точка с запятой). Выражения могут включать идентификаторы, числовые константы, скобки для задания приоритетов, а также функции (например, sin, cos). Операции могут включать сложение, вычитание, умножение и деление.

l.l

%{

#include "tab.h"

#include <stdio.h>

#include <stdlib.h>

%}

%%

[0-9]+(\.[0-9]+)? { yylval.num = atof(yytext); return NUMBER; }

"sin" { return SIN; }

"cos" { return COS; }

"+" { return PLUS; }

"-" { return MINUS; }

"\*" { return MUL; }

"/" { return DIV; }

"(" { return LPAREN; }

")" { return RPAREN; }

";" { return SEMICOLON; }

[a-zA-Z][a-zA-Z0-9]\* { yylval.id = strdup(yytext); return ID; }

[ \t\n] ; /\* Игнорируем пробелы, табуляцию и переносы строк \*/

. { return yytext[0]; }

%%

int yywrap() {

return 1;

}

y.y

%{

#include <stdio.h>

#include <stdlib.h>

void yyerror(const char \*s);

int yylex();

%}

%union {

double num;

char\* id;

}

%token <num> NUMBER

%token <id> ID

%token PLUS MINUS MUL DIV

%token LPAREN RPAREN

%token SEMICOLON

%token SIN COS

%type <num> expr term factor func

%%

program:

statement\_list

;

statement\_list:

statement\_list SEMICOLON statement

| statement

;

statement:

expr { printf("Expression evaluated\n"); }

;

expr:

expr PLUS term { printf("Operation: Addition\n"); }

| expr MINUS term { printf("Operation: Subtraction\n"); }

| term

;

term:

term MUL factor { printf("Operation: Multiplication\n"); }

| term DIV factor { printf("Operation: Division\n"); }

| factor

;

factor:

NUMBER { printf("Number constant\n"); }

| ID { printf("Identifier: %s\n", $1); }

| LPAREN expr RPAREN

| func

;

func:

SIN LPAREN expr RPAREN { printf("Function: sin()\n"); }

| COS LPAREN expr RPAREN { printf("Function: cos()\n"); }

;

%%

void yyerror(const char \*s) {

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

}

int main() {

return yyparse();

}

**Вывод:** в ходе выполнения лабораторной работы изучил работу синтаксических анализаторов YACC и LEX.