**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЯ ОБРАЗОВАНИЯ**

**“ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ ЕВФРОСИНИИ ПОЛОЦКОЙ”**

Факультет информационных технологий

Кафедра технологий программирования

**Отчёт по Лабораторной работе №6**

**«Разработка интерпретатора кода»**

Выполнил студент 2 курса, группа 21-ИТ-1 Макеёнок Д.И.

Проверил Сыцевич Д.Н.

Полоцк, 2022 г.

**Цель работы:** создать интерпретатор языка BASIC. Входные данные: исходный текст программы на языке BASIC. Выходные данные: консоль с результатами ввода-вывода исходной программы, либо сообщения об ошибках, присутствующих в программе.

**Результат работы:**

**main.cpp**

#include "LexicalAnalyzer.h"

int main()

{

table["pi"] = 3.1415926535897932385;

table["e"] = 2.7182818284590452354;

while (std::cin) {

get\_token();

if (curr\_tok == END) break;

if (curr\_tok == PRINT) continue;

std::cout << expr(false) << '\n';

}

return no\_of\_errors;

}

**LexicalAnalyzer.cpp**

#include "LexicalAnalyzer.h"

std::map<std::string, double>table;

Token\_value curr\_tok = PRINT;

double expr(bool get)

{

double left = term(get);

for (;;)

switch (curr\_tok) {

case PLUS:

left += term(true);

break;

case MINUS:

left -= term(true);

break;

default:

return left;

}

}

double term(bool get)

{

double left = prim(get);

for (;;)

switch (curr\_tok) {

case MUL:

left \*= prim(true);

break;

case DIV:

if (double d = prim(true)) {

left /= d;

break;

}

return error("Деление на ноль");

default:

return left;

}

}

double number\_value;

std::string string\_value;

double prim(bool get)

{

if (get) get\_token();

switch (curr\_tok) {

case NUMBER: {

double& v = number\_value;

get\_token();

return v;

}

case NAME: {

double& v = table[string\_value];

if (get\_token() == ASSIGN) v = expr(true);

return v;

}

case MINUS:

return -prim(true);

case LP: {

double e = expr(true);

if (curr\_tok != RP) return error("Ожидалась )");

get\_token();

return e;

}

default:

return error("Ожидалось первичное выражение");

}

}

Token\_value get\_token()

{

char ch = 0;

do {

if (!std::cin.get(ch)) return curr\_tok = END;

} while (ch != '\n' && isspace(ch));

switch (ch) {

case 0:

return curr\_tok = END;

case ';':case '\n':

return curr\_tok = PRINT;

case '\*':case'/':case '+':case '-':case '(':case ')':case '=':

return curr\_tok = Token\_value(ch);

case '0':case '1':case '2':case '3':case '4':

case '5':case '6':case '7':case '8':case '9':case '.':

std::cin.putback(ch);

std::cin >> number\_value;

return curr\_tok = NUMBER;

default:

if (isalpha(ch)) {

string\_value = ch;

while (std::cin.get(ch) && isalnum(ch)) string\_value.push\_back(ch);

std::cin.putback(ch);

return curr\_tok = NAME;

}

error("Неправильная лексема");

return curr\_tok = PRINT;

}

}

int no\_of\_errors = 0;

int error(const std::string& s)

{

no\_of\_errors++;

std::cerr << "Ошибка: " << s << '\n';

return no\_of\_errors;

}

Рисунок 1 – Результат работы программы

**Вывод:** в результате выполнения лабораторной работы был создан интерпретатор языка BASIC.