МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ОТЧЁТ ПО ПРАКТИЧЕСКИМ ЗАНЯТИЯМ

по дисциплине

«Системное программное обеспечение»

Выполнил студент группы ИКБО-03-18 Маковецкий И. А.

Принял Соболев О. В.

Практические занятия выполнены «____» _____ 2021 г. (подпись студента)

Практические занятия зачтены «____» _____ 2021 г. (подпись руководителя)

Москва 2021

СОДЕРЖАНИЕ

1 ПРАКТИЧЕСКАЯ РАБОТА № 1

1.1 Тема

Генерация кода для выражения с константами и переменными.

1.2 Задание

Реализовать генерацию ассемблерного кода для арифметического выражения из констант и переменных.

1.3 Ход выполнения работы

В ходе выполнения работы был реализован модуль транслятора, осуществляющий генерацию asm-кода для арифметических выражений, состоящих из целочисленных констант и операций сложения, вычитания, умножения, целочисленного деления, взятия остатка от деления. Разработка опирается на лексический и синтаксический анализатор, реализованные в результате освоения дисциплины «Теория автоматов и формальных языков». Входными данными для модуля является абстрактное синтаксическое дерево (AST), получаемое в результате лексического и синтаксического разбора математических выражений, записанных в текстовом виде. Выходными данными является asm-код, представляющий собой код вычисления выражения, поданного в качестве входных данных.

Генерация кода для узлов AST выполняется по правилам, описанным ниже. Для узлов, представляющих константу — команда ассемблера $PUSH$

$QWORD < n >$, где $< n >$ — константа, хранящаяся в узле. Для узлов с бинарной операцией — команды ассемблера:

- $POP\ RBX$; второй операнд бинарной операции — сверху стека;
- $POP\ RAX$; первый операнд бинарной операции — под вторым;
- Код конкретной операции;
- $PUSH\ RAX$; помещение результата операции на вершину стека.

Ниже приведен фрагмент кода обхода AST.

```
\linespread{1.0}
public static void generateASM(Set<String> vars) {
    for (String var: vars) {
        System.out.println("MOV RCX, promt_" +  var);
        System.out.println("MOV R11, printf");
        System.out.println("CALL R11");

        System.out.println(" ");
        System.out.println("MOV RDX, scanf_format");
        System.out.println("MOV RDX, " + var);
        System.out.println("MOV R11, scanf");
        System.out.println("CALL R11\n");
    }
}

public static void main(String[] args) {
    String text = "x + y + 2";

    Lexer l = new Lexer(text);
    List<Token> tokens = l.lex();
    tokens.removeIf(t -> t.type == TokenType.SPACE);
```

```
        Parser p = new Parser(tokens);
        ExprNode node = p.parseExpression();
        Set<String> vars = new LinkedHashSet<>();
        getVars(vars, node);

        System.out.println("section .text\n" +
            " global main\n" +
            " extern printf\n" +
            " extern scanf\n" +
            " \n" +
            "main: ");
    }
```