МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №  43

ОТЧЁТ

ЗАЩИЩЁН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

### ст.пр.                                   Рогачёв С.А.

должность, уч. Степень, звание   подпись, дата           инициалы, фамилия

ОТЧЁТ О ЛАБОРАТОРНОЙ РАБОТЕ №2.

Машина Тьюринга.

по курсу: Теория вычислительных процессов

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. 4136                                                                                Бобрович Н. С.

                                                                         подпись, дата                      инициалы, фамилия

Санкт-Петербург 2024

1. **Цель работы:**

Изучить Машину Тьюринга.

1. **Основные сведения из теории:**

Содержательно Машина Тьюринга (МТ) как абстрактный автомат, реализующий алгоритм вычисления некоторой вычислимой функции, состоит из трех компонентов:

1. Управляющее устройство (УУ), которое может находиться в одном из состояний, образующих конечное множество } z ,q n Q {q ,q ,...,q 0 1  — внутренний алфавит машины Тьюринга;
2. Бесконечная лента, разбитая на ячейки, в каждой из которых может быть записан один из символов конечного алфавита { , ,..., , } A  a1 a2 am  — внешний алфавит машины Тьюринга;

3. Устройство обращения к ленте — считывающая и записывающая головка, которая в текущий момент времени считывает или записывает значение одной (текущей) ячейки ленты;

1. **Постановка задачи:**

Необходимо написать программу для машины Тьюринга, реализующую вычисление арифметической функции согласно выданному варианту задания. Должна быть составлена совокупность команд P.

Для выполнения данного задания следует использовать приложение Algo2000. Аргументы задаются набором ”1”.

Пример 2\*3, будет выглядеть следующим образом 11\*111.

Работа машины Тьюринга должна начинаться со стандартной начальной конфигурации и заканчиваться стандартной конечной конфигурацией.

**Вариант 30: 2x**

1. **Совокупность команд для машины Тьюринга:**

q0 1 -> q0 \_ >

q0 \* -> q1 \* >

q1 1 -> q2 \* <

q1 \* -> q1 \* >

q1 \_ -> q5 \_ <

q2 \* -> q3 \* <

q2 \_ -> q3 1 <

q3 1 -> q3 1 <

q3 \* -> q3 \* <

q3 \_ -> q4 1 <

q4 1 -> q4 1 >

q4 \* -> q1 \* >

q4 \_ -> q4 1 >

q5 1 -> ! 1 >

q5 \* -> q5 \_ <

1. **Листинг программы на языке высокого уровня с комментариями:**

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <map>

#include <sstream>

#include <iomanip>

using namespace std;

// Структура для представления одной команды машины Тьюринга

struct Command {

string current\_state;

char read\_symbol;

string next\_state;

char write\_symbol;

char move\_direction;

string smth;

};

int main() {

setlocale(LC\_ALL, "Russian");

// Чтение входных данных из файла input.txt

ifstream input\_file("input.txt");

if (!input\_file.is\_open()) {

cerr << "Не удалось открыть файл input.txt!" << endl;

return 1;

}

string tape;

getline(input\_file, tape);

input\_file.close();

// Чтение алфавита из файла alphabet.txt

ifstream alphabet\_file("alphabet.txt");

if (!alphabet\_file.is\_open()) {

cerr << "Не удалось открыть файл alphabet.txt!" << endl;

return 1;

}

vector<char> alphabet;

string line;

while (getline(alphabet\_file, line)) {

for (char c : line) {

if (c != ' ') {

alphabet.push\_back(c);

}

}

}

alphabet\_file.close();

// Проверка наличия всех необходимых символов в алфавите

bool valid\_tape = true;

for (char c : tape) {

if (find(alphabet.begin(), alphabet.end(), c) == alphabet.end()) {

cerr << "Символ '" << c << "' не найден в алфавите." << endl;

valid\_tape = false;

}

}

if (!valid\_tape) {

return 1;

}

// Чтение команд из файла commands.txt

ifstream commands\_file("commands.txt");

if (!commands\_file.is\_open()) {

cerr << "Не удалось открыть файл commands.txt!" << endl;

return 1;

}

map<string, map<char, Command>> transition\_table;

while (getline(commands\_file, line)) {

istringstream iss(line);

string state, symbol, smth, new\_state, write, direction;

iss >> state >> symbol >> smth >> new\_state >> write >> direction;

Command command{ state, symbol[0], new\_state, write[0], direction[0], smth }; // Передали значение smth

transition\_table[state][symbol[0]] = command;

}

commands\_file.close();

// Выполнение программы машины Тьюринга

int head\_position = 0;

string current\_state = "q0";

ofstream output\_file("output.txt");

if (!output\_file.is\_open()) {

cerr << "Не удалось создать файл output.txt!" << endl;

return 1;

}

while (current\_state != "!") {

// Выводим текущее состояние ленты и положение головки

output\_file << tape << endl;

output\_file << setw(head\_position + 1) << '^' << endl;

// Получаем текущую команду

auto it = transition\_table.find(current\_state);

if (it == transition\_table.end()) {

cerr << "Нет перехода для состояния " << current\_state << endl;

break;

}

auto command\_it = it->second.find(tape[head\_position]);

if (command\_it == it->second.end()) {

cerr << "Нет команды для символа " << tape[head\_position] << " в состоянии " << current\_state << endl;

break;

}

const Command& command = command\_it->second;

output\_file << command.current\_state << " " << command.read\_symbol << " " << command.smth << " "

<< command.next\_state << " " << command.write\_symbol << " " << command.move\_direction << endl;

// Обновляем ленту и головку

tape[head\_position] = command.write\_symbol;

// Реализуем движение головки

if (command.move\_direction == '>') {

if (head\_position == tape.size() - 1) { // Если достигли конца ленты

tape += "\_"; // Добавляем "\_" справа

}

head\_position++; // Двигаем голову вправо

}

else if (command.move\_direction == '<') {

if (head\_position == 0) { // Если достигли начала ленты

tape.insert(0, 1, '\_'); // Добавляем "\_" слева

head\_position = 0; // Сдвигаемся на одну позицию вправо

}

else {

head\_position--; // Двигаем голову влево

}

}

// Переходим в новое состояние

current\_state = command.next\_state;

}

output\_file.close();

cout << "Результат выполнения программы записан в файл output.txt" << endl;

return 0;

}

1. **Пример результата выполнения:**

Файл output.txt:

111\*111

^

q0 1 -> q0 \_ >

\_11\*111

^

q0 1 -> q0 \_ >

\_\_1\*111

^

q0 1 -> q0 \_ >

\_\_\_\*111

^

q0 \* -> q1 \* >

\_\_\_\*111

^

q1 1 -> q2 \* <

\_\_\_\*\*11

^

q2 \* -> q3 \* <

\_\_\_\*\*11

^

q3 \_ -> q4 1 <

\_\_1\*\*11

^

q4 \_ -> q4 1 >

\_11\*\*11

^

q4 1 -> q4 1 >

\_11\*\*11

^

q4 \* -> q1 \* >

\_11\*\*11

^

q1 \* -> q1 \* >

\_11\*\*11

^

q1 1 -> q2 \* <

\_11\*\*\*1

^

q2 \* -> q3 \* <

\_11\*\*\*1

^

q3 \* -> q3 \* <

\_11\*\*\*1

^

q3 1 -> q3 1 <

\_11\*\*\*1

^

q3 1 -> q3 1 <

\_11\*\*\*1

^

q3 \_ -> q4 1 <

\_111\*\*\*1

^

q4 \_ -> q4 1 >

1111\*\*\*1

^

q4 1 -> q4 1 >

1111\*\*\*1

^

q4 1 -> q4 1 >

1111\*\*\*1

^

q4 1 -> q4 1 >

1111\*\*\*1

^

q4 \* -> q1 \* >

1111\*\*\*1

^

q1 \* -> q1 \* >

1111\*\*\*1

^

q1 \* -> q1 \* >

1111\*\*\*1

^

q1 1 -> q2 \* <

1111\*\*\*\*

^

q2 \* -> q3 \* <

1111\*\*\*\*

^

q3 \* -> q3 \* <

1111\*\*\*\*

^

q3 \* -> q3 \* <

1111\*\*\*\*

^

q3 1 -> q3 1 <

1111\*\*\*\*

^

q3 1 -> q3 1 <

1111\*\*\*\*

^

q3 1 -> q3 1 <

1111\*\*\*\*

^

q3 1 -> q3 1 <

\_1111\*\*\*\*

^

q3 \_ -> q4 1 <

\_11111\*\*\*\*

^

q4 \_ -> q4 1 >

111111\*\*\*\*

^

q4 1 -> q4 1 >

111111\*\*\*\*

^

q4 1 -> q4 1 >

111111\*\*\*\*

^

q4 1 -> q4 1 >

111111\*\*\*\*

^

q4 1 -> q4 1 >

111111\*\*\*\*

^

q4 1 -> q4 1 >

111111\*\*\*\*

^

q4 \* -> q1 \* >

111111\*\*\*\*

^

q1 \* -> q1 \* >

111111\*\*\*\*

^

q1 \* -> q1 \* >

111111\*\*\*\*

^

q1 \* -> q1 \* >

111111\*\*\*\*\_

^

q1 \_ -> q5 \_ <

111111\*\*\*\*\_

^

q5 \* -> q5 \_ <

111111\*\*\*\_\_

^

q5 \* -> q5 \_ <

111111\*\*\_\_\_

^

q5 \* -> q5 \_ <

111111\*\_\_\_\_

^

q5 \* -> q5 \_ <

111111\_\_\_\_\_

^

q5 1 -> ! 1 >

1. **Вывод:**

Изучил Машину Тьюринга.