Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

О Т Ч Е Т

по домашней работе

Дисциплина: Прикладная теория цифровых автоматов

Тема: Игра «ножницы – камень - бумага»

Студент	ИУ6-41Б		28.05.2022	И.В.Ротанков
	(Группа)		(Подпись, дата)	(И.О. Фамилия)
Преподаватель				Ю.И.Бауман
			(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящей работе реализован цифровой автомат «Игра ножницы - камень - бумага».

В настоящей работе использован программный способ реализации цифрового автомата, так как этот способ подразумевает вариативность реализации, возможность отладки и тестирования в процессе разработки программы. К программам (в отличие от аппаратной реализации цифровых автоматов) можно добавлять новые функции по мере изменения целей, под которые она разрабатывается.

Задание: построить автомат, реализующий игру ножницы – камень – бумага (вариант 13).

Цель работы – применив знания, полученные на лекциях и семинарах, построить конечный автомат в виде ориентированного графа состояний с входными и выходными сигналами и таблицы переходов, а также реализовать построенный автомат с помощью выбранного программного средства. Для реализации поставленной цели необходимо выполнить следующие задачи.

Задачи:

− Изучить задание в соответствии со своим вариантом;

− Описать автомат, соответствующий условию задачи;

− Изучить способы реализации цифровых автоматов;

− Выбрать один из способов реализации автоматов;

− Реализовать описанный цифровой автомат.

ОСНОВНАЯ ЧАСТЬ

Спецификация автомата

Необходимо построить автомат в соответствие с следующим заданием.

Задание: реализовать конечный автомат для игры «ножницы – камень - бумага». Два игрока одновременно называют один из трех перечисленных предметов. Если они назвали одно и то же – ничья. Если названы разные предметы, то выигрыш определяется следующим образом: ножницы режут бумагу, бумага закрывает камень, камень тупит ножницы. Количество партий фиксировано.

По определению конечным автоматом Мили называется шестерка объектов: $A =< S, X, Y, s_0, \delta, \lambda >$, где

S – конечное непустое множество состояний

X – конечное непустое множество входных сигналов

Y – конечное непустое множество выходных сигналов

$s_0 \in S$ – начально состояние

$\delta: S \times X \to S$ – функция переходов

$\lambda: S \times X \to S$ – функция выходов

Определим данные объекты в случае полученного задания. По условию количество партий фиксировано. Таким образом, мы можем ограничить общее количество партий - 3 партиями. Тогда:

1) Состояния автомата

Н_C – начальный счет 0:0 первая игра еще не сыграна

Н_1 – ничья после первой партии

Н_2 – ничья после второй партии

Н_K – ничья после третьей партии

П_1 – игрок проигрывает после первой партии

$П_2$ – игрок проигрывает после второй партии

$П_К$ – игрок проигрывает после третьей партии

$В_1$ – игрок ведет в счете после первой партии

$В_2$ – игрок ведет в счете после второй партии

$В_К$ – игрок ведет в счете после третьей партии

2) Входные сигналы

КК – оба игрока показали «камень»

ББ – оба игрока показали «бумагу»

НН – оба игрока показали «ножницы»

КН – первый игрок показал «камень» второй – «ножницы»

КБ – первый игрок показал «камень» второй – «ножницы»

НБ – первый игрок показал «ножницы» второй – «бумагу»

НК – первый игрок показал «ножницы» второй – «камень»

БК – первый игрок показал «бумагу» второй – «камень»

БН – первый игрок показал «бумагу» второй – «ножницы»

Данные сигналы можно свести всего к трем

x_1 – ничейные комбинации

x_2 – комбинации, при которых в партии выиграет игрок 1

x_3 – комбинации, при которых в партии выиграет игрок 2

3) Выходные сигналы

y_1 – Игрок 1 выиграл партию

y_2 – Игрок 2 выиграл партию

y_3 – Ничья в партии

y_4 – Игрок 1 выиграл игру

y_5 – Игрок 2 выиграл игру

y_6 – Ничья в игре

Таким образом будет построен автомат, в котором состояниями являются статусы для какого-то из игроков. Для простоты выберем первого игрока.

Полученный цифровой автомат

Составим таблицу, описывающую конечный автомат, составленный по условию задачи в результате анализа (см. таблицу 1).

Таблица 1 – Талица переходов автомата

Состояние	δ			λ		
	x_1	x_2	x_3	x_1	x_2	x_3
Н_C	Н_1	B_1	П_1	y_3	y_1	y_2
Н_1	Н_2	B_2	П_2	y_3	y_1	y_2
Н_2	Н_K	B_K	П_K	y_6	y_4	y_2
Н_K	–	–	–	–	–	–
П_1	П_2	Н_2	П_K	y_3	y_1	y_5
П_2	П_K	Н_K	П_K	y_5	y_6	y_5
П_K	–	–	–	–	–	–
B_1	B_2	B_K	Н_2	y_3	y_4	y_2
B_2	B_K	B_K	Н_K	y_4	y_4	y_6
B_K	–	–	–	–	–	–

Построим граф переходов по данной таблице (рисунок 1).

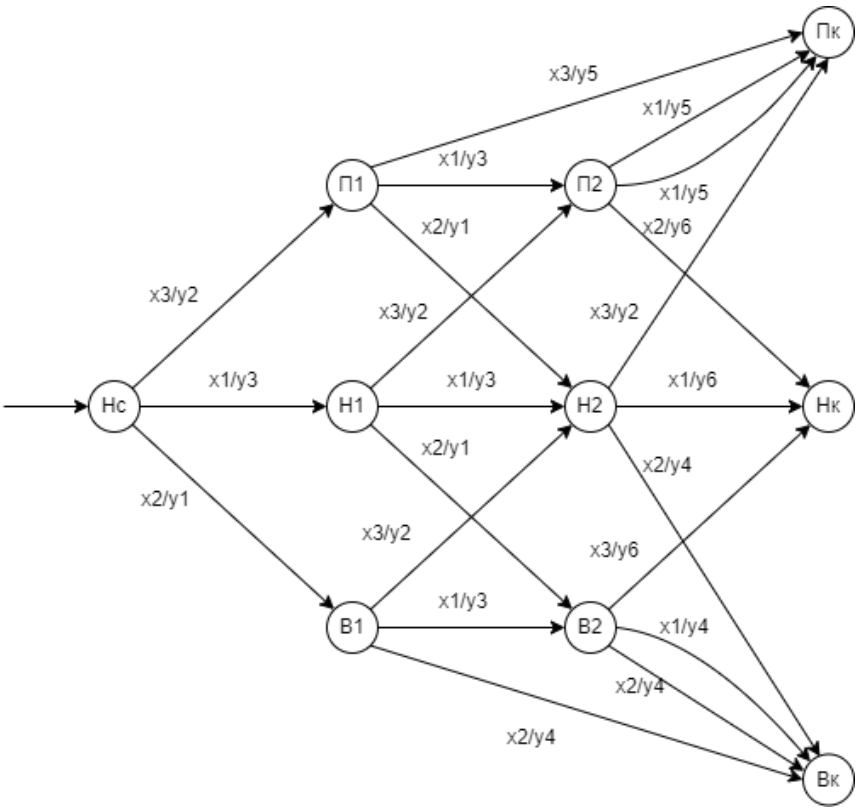

Рисунок 1 – Построенный автомат

Программная реализация конечного автомата для игры «ножницы – камень - бумага»

Данный автомат был реализован с помощью программных средств языка C++. Ниже представлена схема алгоритма данной программы (рисуноки 2 - 3) и листинг программы.

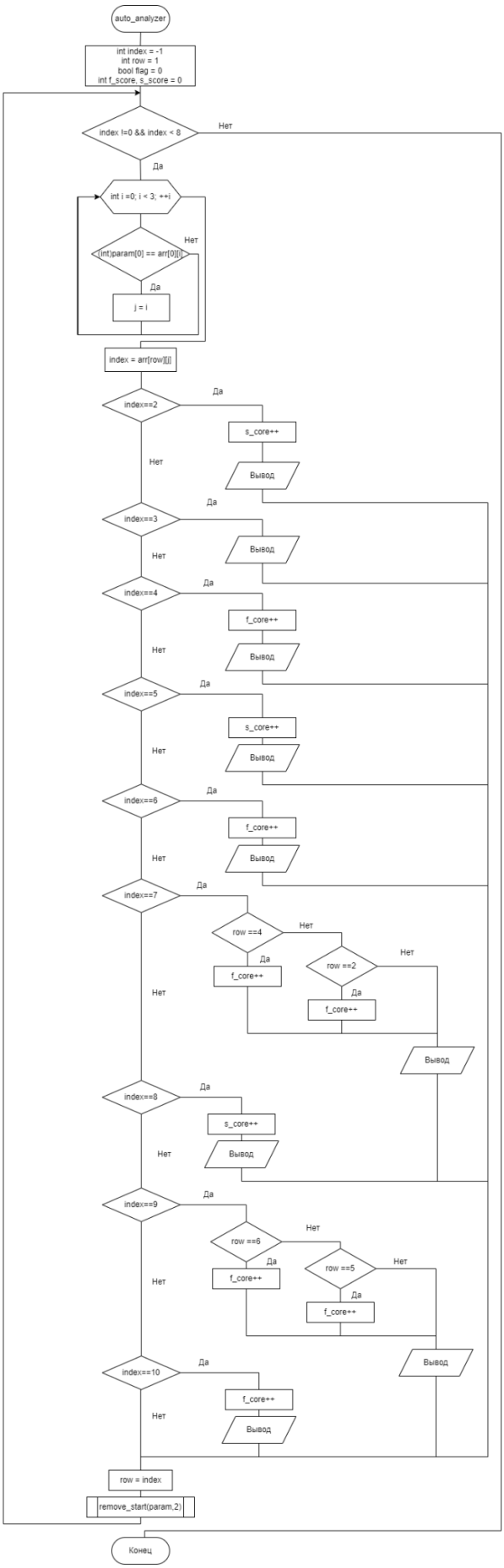

Рисунок 2 – схема алгоритма

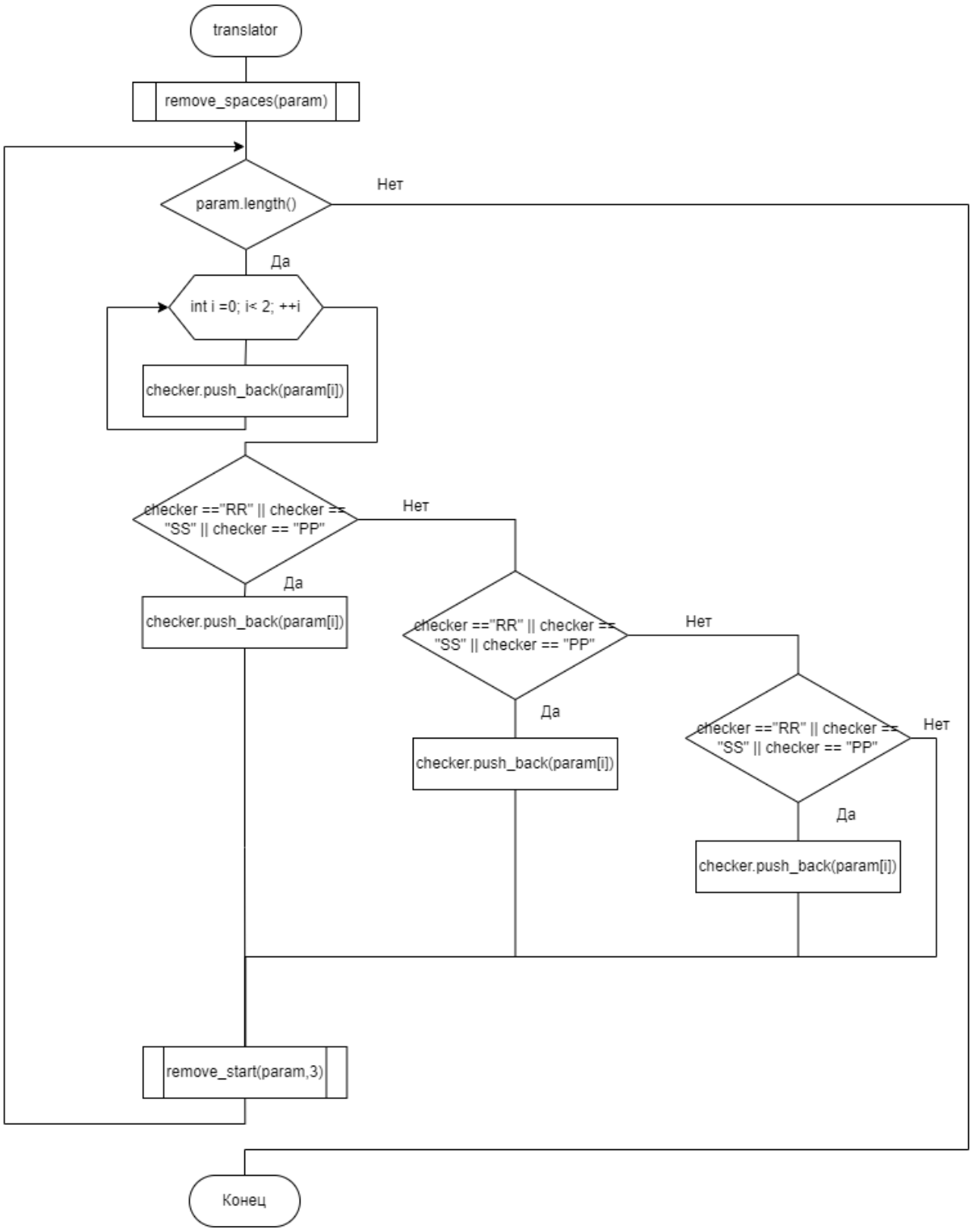

Рисунок 3 – Схема алгоритма

Реализация автомата на C++ (листинг 1)

```
#include <iostream>
#include <string>
```

```
#include <vector>
#include <algorithm>


int remove_start(std::string &param, int count){
   param.erase(param.begin(), param.begin() + count - 1);
   return 0;
}

void remove_spaces(std::string &param){
   std::string::iterator end_pos = std::remove(param.begin(), param.end(), ' ');
        param.erase(end_pos, param.end());
}

void auto_analyzer(std::string &param){
        std::vector<std::vector<int>>                                arr                                =
{{97,98,99},{3,4,2},{5,7,8},{7,6,5},{6,10,7},{8,9,8},{10,10,9},{9,10,8}};
        int index = -1, row = 1;
        int str_ind = 0;
        bool flag = false;
        int f_score = 0, s_score = 0;
        while(index != 0 && index < 8){
                int j;
                for(int i = 0; i < 3; ++i){
                        if((int)param[0] == arr[0][i]){
                                j = i;
                        }
                }
                index = arr[row][j];
                if(index == 2){
                        s_score++;
                        std::cout << "Second player won first game" << std::endl;
                        std::cout << "Current score - "<< f_score <<":"<< s_score <<
std::endl;
                }
                if(index == 3){
                        std::cout << "Tie" <<std::endl;
                        std::cout << "Current score - "<< f_score <<":"<< s_score <<
std::endl;
                }
                if(index == 4){
                        f_score++;
                        std::cout << "First player won first game" << std::endl;
```

```
                std::cout << "Current score - "<< f_score <<":"<< s_score <<
std::endl;
        }
        if(index == 5){
                s_score++;
                std::cout << "Second player won second game" <<std::endl;
                std::cout << "Current score - "<< f_score <<":"<< s_score <<
std::endl;
        }
        if(index == 6){
                f_score++;
                std::cout << "First player won second game" << std::endl;
                std::cout << "Current score - "<< f_score <<":"<< s_score <<
std::endl;
        }
        if(index == 7){
                if(row == 4){
                        s_score++;
                }
                if(row == 2){
                        f_score++;
                }
                std::cout << "Tie" <<std::endl;
                std::cout << "Current score - "<< f_score <<":"<< s_score <<
std::endl;
        }
        if(index == 9){
                if(row == 6){
                        s_score++;
                }
                if(row == 5){
                        f_score++;
                }
                std::cout << "Tie game!" <<std::endl;
               std::cout << "Final score - "<< f_score <<":"<< s_score << std::endl;
        }
        if(index == 10){
                f_score++;
                std::cout << "First player won the game!" <<std::endl;
               std::cout << "Final score - "<< f_score <<":"<< s_score << std::endl;
        }
        if(index == 8){
                s_score++;
                std::cout << "Second player won the game!" << std::endl;
```

```
                std::cout << "Final score - "<< f_score <<":"<< s_score << std::endl;
            }
            row = index;
            remove_start(param,2);
        }
}

std::string translator(std::string &param){
        remove_spaces(param);
        std::string translated;
        while(param.length()){
            std::string checker;
            for(int i = 0; i < 2; ++i){
                checker.push_back(param[i]);
            }
            if(checker == "RR" || checker == "PP" || checker == "SS"){
                translated.push_back('a');
            }else if(checker == "RS" || checker == "SP" || checker == "PR"){
                translated.push_back('b');
            }else if(checker == "RP" || checker == "SR" || checker == "PS"){
                translated.push_back('c');
            }
            if(remove_start(param,3)){
                std::cout << "Error" <<std::endl;
            }
        }
        return translated;
}


int main(){
   std::string input = "RS PS PP";
   std::cout << "Welcome to the game!" <<std::endl;
   std::cout << "Input string: ";
   std::cout << input << std::endl;
   std::string res_str = translator(input);
   auto_analyzer(res_str);
   return 0;
}
```

Тестирование программы

Протестируем написанную программу (рисунки 4 -13).

Для успешного тестирования необходимо проверить все ветви условий,

по которым может переходить программа. Ниже представлены входные строки, покрывающие все ветви.

1) «КН КН»
2) «КН НН КН»
3) «КН НН НН»
4) «КН НН БН»
5) «КН БН ББ»
6) «КН БН БН»
7) «НН КН КН»
8) «НН КН НН»
9) «НН КН НК»
10) «НН НН НН»

*Примечание: R – камень, S – ножницы, P – бумага.

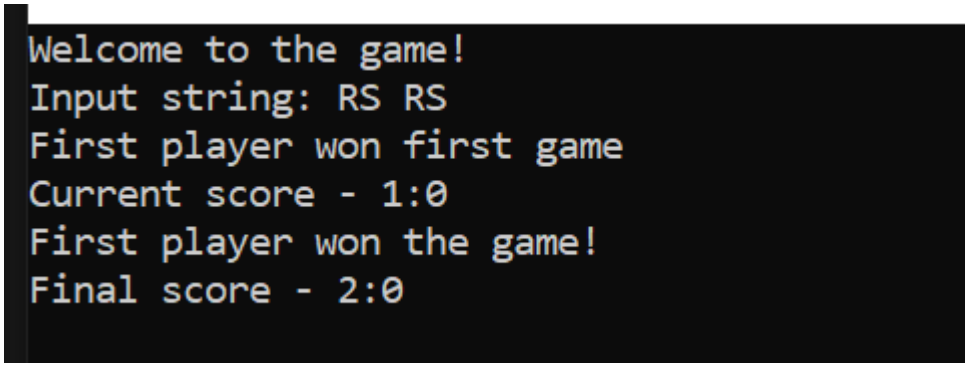

Рисунок 4 – Тест 1

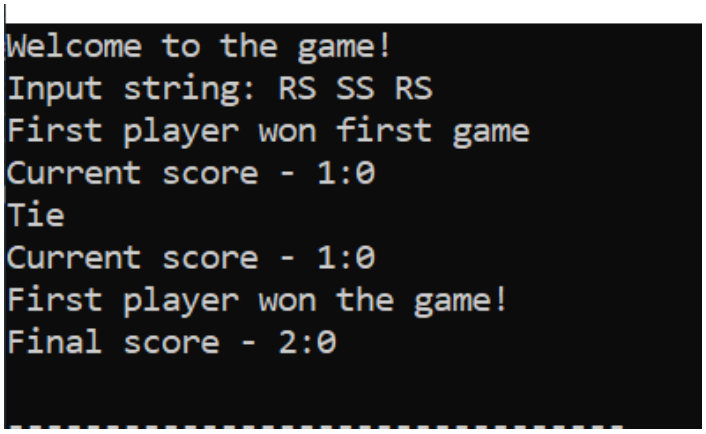

Рисунок 5 – Тест 2

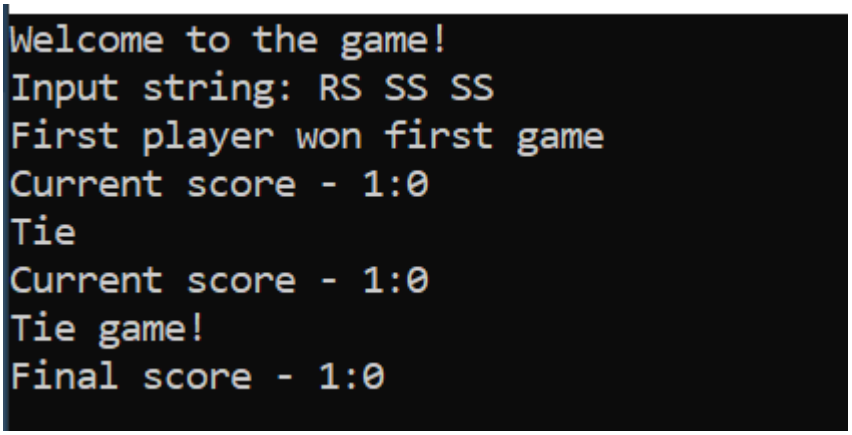

Рисунок 6 – Тест 3

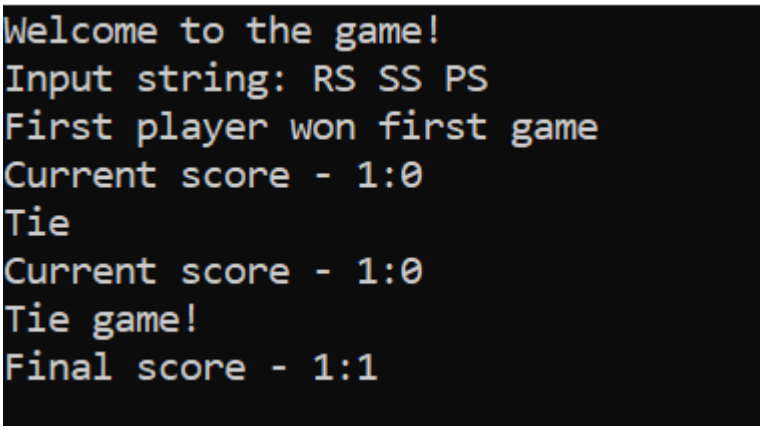

Рисунок 7 – Тест 4

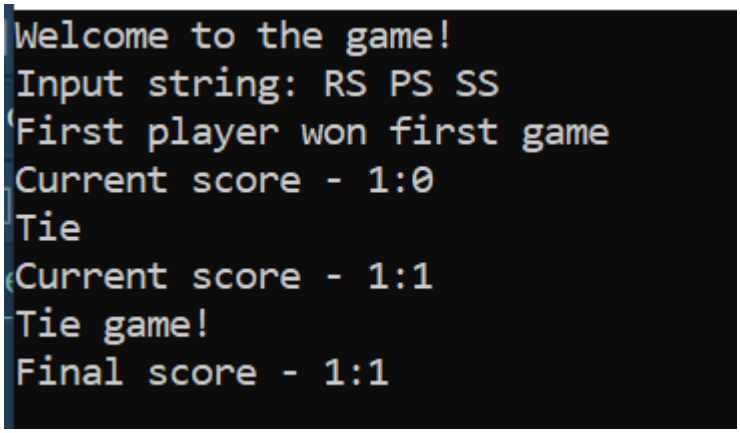

Рисунок 8 – Тест 5

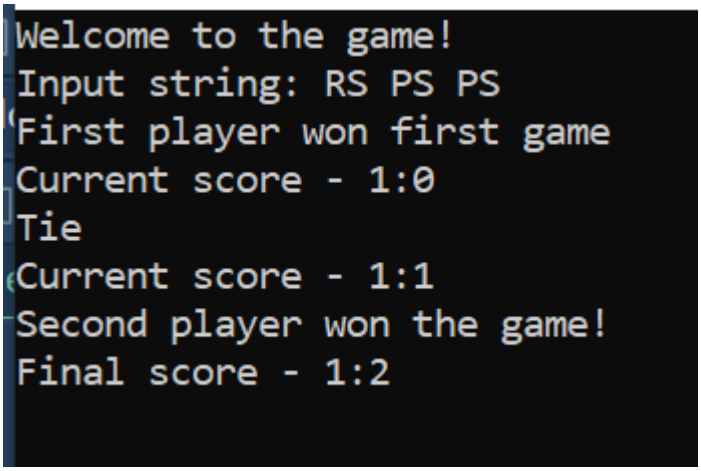

Рисунок 9 – Тест 6

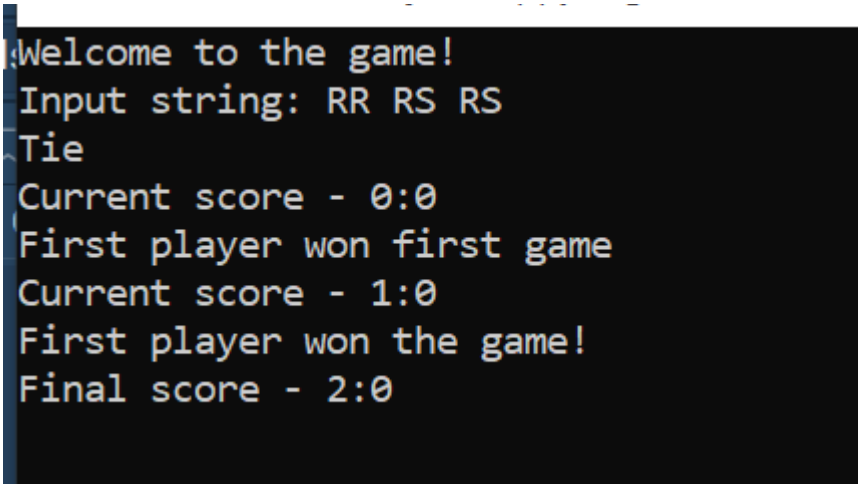

Рисунок 10 – Тест 7

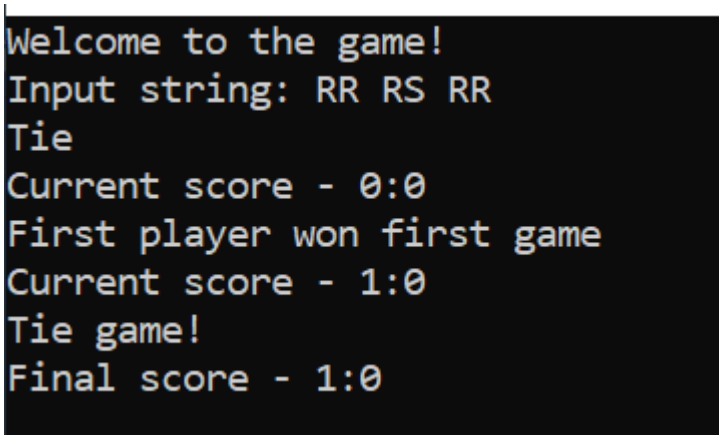

Рисунок 11 – Тест 8

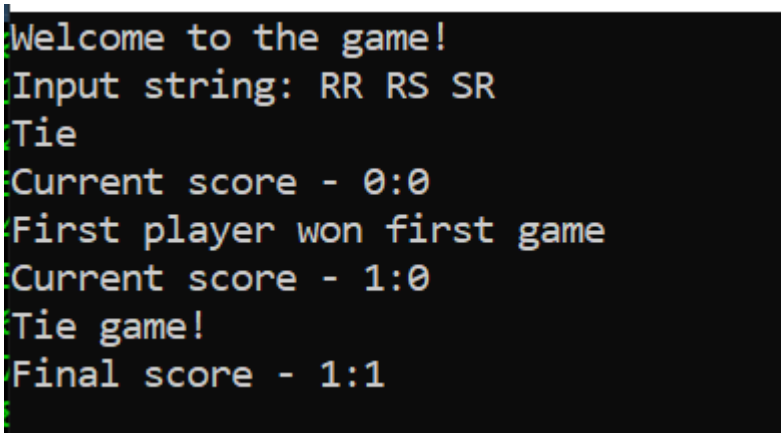

Рисунок 12 – Тест 9

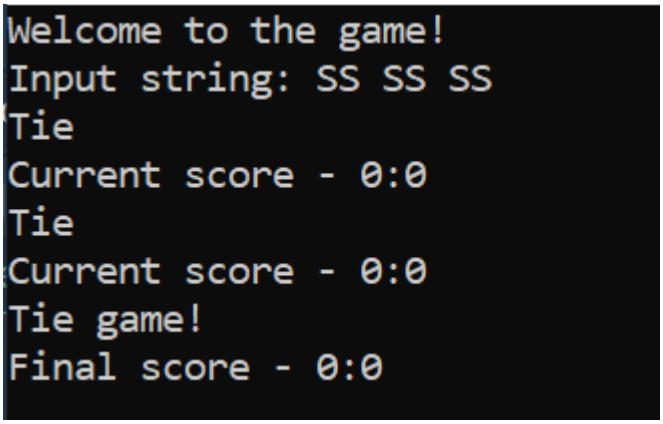

Рисунок 13 – Тест 10

Все протестированные маршруты работают корректно. В силу симметричности автомата достаточно рассмотреть только 10 маршрутов. То есть остальные маршруты будут аналогичными, только для второго игрока, поэтому без ограничения общности можно сказать, что данных тестов достаточно.

ЗАКЛЮЧЕНИЕ

1) При выполнении домашнего задания изучен программный способ реализации конечных цифровых автоматов.

2) В ходе выполнения данного домашнего задания спроектирован и реализован конечный автомат для игры «ножницы – камень – бумага». Создана программная реализация автомата на языке C++.

3) Закреплены навыки подготовки и оформления отчета по проделанной работе с учетом требований ГОСТ 7.32-2017.

ИСТОЧНИКИ

1) ГОСТ 7.32-2017 Система стандартов информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. – URL: https://docs.cntd.ru/document/1200157208 (дата обращения 28.05.2022).

Электронные ресурсы:

1) Лекции по дисциплине «Прикладная теория цифровых автоматов». – URL: https://lks.bmstu.ru/teacher (дата обращения 28.05.2022).