МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИИТ. ПРОГРАМНАЯ ИНЖЕНЕРИЯ

Теория автоматов и формальных языков

ЛАБОРАТОРНАЯ РАБОТА №1-2

ПОСТРОЕНИЕ МОДЕЛИ АВТОМАТА МИЛИ И МУРА

Исполнитель

Студент Зернов Владислав

Группа 1ПИб-02-2оп-22

Руководитель Ганичева Оксана

Георгиевна

Ф.И.О. преподавателя

2023 год

**Задание:**

1. Сделать описание работы автомата выдачи напитков.

2. Сделать формальное описание модели (X, Y, S, s0, δ, λ) все … подробно описать, составить 3 таблицы, построить граф автомата Мили.

3. Проверить работу автомата Мили составив протокол работы (не менее 20 символов).

4. Построить граф автомата Мура эквивалентный автомату Мили.

5 Проверить работу автомата Мура составив протокол работы (не менее 20 символов).

6. Вывод

**Решение:**

1. Описание работы автомата выдачи напитков

Автомат выдачи напитков - это устройство, предназначенное для автоматизированной выдачи различных напитков. Рассмотрим его работу с точки зрения теории автоматов:

Начало: Пользователь начинает взаимодействие вставляя монеты номинала 5 (вход x5) или 10 (вход x10) рублей. Это переводит автомат в состояние S5 или S10 соответственно. После того, как автомат достигнет состояния S40, S45 или S50, можно будет выбрать напиток.

Выбор напитка: Пользователь может выбрать напиток кофе (вход XCoffee) или шоколад (вход XChocolate), переводя автомат в состояния Sorg. После выбора напитка можно добавить или убавить сахар

Выбор сахара: После выбора напитка пользователь может выбрать добавить (вход XSugarPlus) или убавить (взод XSugarMinus) сахар переводя автомат в состояние Sorg.

Завершение операции: По окончании операции, автомат выдает выбранный напиток и возвращает сдачу (если она есть), переходя в состояние S0 или при попытке вставить монеты, когда сумма всех вставленных монет и этой монеты будет больше 50 рублей, то автомат выдаст сдачу и перейдет в состояние S0.

Эта работа автомата выдачи напитков с точки зрения теории автоматов демонстрирует, как устройство принимает входные данные, переходит между состояниями и выполняет операции в соответствии с определенными правилами. Это типичный пример автомата с конечным числом состояний, который используется для автомата выдачи напитков.

1. Составные компоненты автомата Мили

Входной алфавит X

* X5 – опущена монета 5 рублей
* X10 – опущена монета 10 рублей
* Xchocolate – нажата кнопка «Горячий шоколад»
* xCoffee – нажата кнопка «Кофе»
* xSugarPlus – нажата кнопка «Увеличить сахар»
* xSugarMinus – нажата кнопка «Уменьшить сахар»
* xChancge – нажата кнопка «Сдача»

Выходной алфавит Y

* y1 – выбран горячий шоколад
* y2 – выбран кофе
* y3 – выбран напиток и сдача
* y5 – принято 5 рублей
* y10 – принято 10 рублей
* ySugarPlus – Сахар увеличен
* ySugarMinus – Сахар уменьшен
* yChancge – «Переполнение автомата и выдача сдачи»

Множество состояний S

* S0 – принято сумма 0 рублей
* S5 – принято сумма 5 рублей
* S10 – принято сумма 10 рублей
* S15 – принято сумма 15 рублей
* S20 – принято сумма 20 рублей
* S25 – принято сумма 25 рублей
* S30 – принято сумма 30 рублей
* S35 – принято сумма 35 рублей
* S40 – принято сумма 40 рублей
* S45 – принято сумма 45 рублей
* S50 – принято сумма 50 рублей
* Sorg – напиток выбран, можно добавить сахар

Функция переходов

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X\S | S0 | S5 | S10 | S15 | S20 | S25 | S30 | S35 | S40 | S45 | S50 | Sorg |
| X5 | S5 | S10 | S15 | S20 | S25 | S30 | S35 | S40 | S45 | S50 | S0 |  |
| X10 | S10 | S15 | S20 | S25 | S30 | S35 | S40 | S45 | S50 | S0 | S0 |  |
| Xchocolate |  |  |  |  |  |  |  |  | Sorg | Sorg | Sorg |  |
| XCoffee |  |  |  |  |  |  |  |  | Sorg | Sorg | Sorg |  |
| XSugarPlus |  |  |  |  |  |  |  |  |  |  |  | Sorg |
| XSugarMinus |  |  |  |  |  |  |  |  |  |  |  | Sorg |
| XChancge |  |  |  |  |  |  |  |  |  |  |  | S0 |

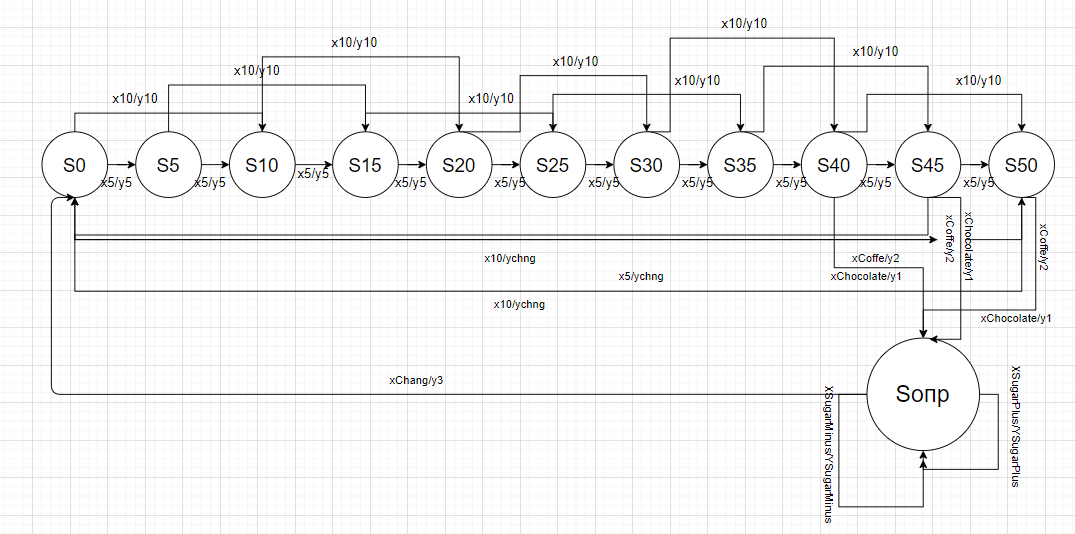
Функция выходов

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X\S | S0 | S5 | S10 | S15 | S20 | S25 | S30 | S35 | S40 | S45 | S50 | Sorg |
| X5 | Y5 | Y5 | Y5 | Y5 | Y5 | Y5 | Y5 | Y5 | Y5 | Y5 | yChancge |  |
| X10 | Y10 | Y10 | Y10 | Y10 | Y10 | Y10 | Y10 | Y10 | Y10 | yChancge | yChancge |  |
| Xchocolate |  |  |  |  |  |  |  |  | Y1 | Y1 | Y1 |  |
| XCoffee |  |  |  |  |  |  |  |  | Y2 | Y2 | Y2 |  |
| XSugarPlus |  |  |  |  |  |  |  |  |  |  |  | YSugarPlus |
| XSugarMinus |  |  |  |  |  |  |  |  |  |  |  | YSugarMinus |
| XChancge |  |  |  |  |  |  |  |  |  |  |  | Y3 |

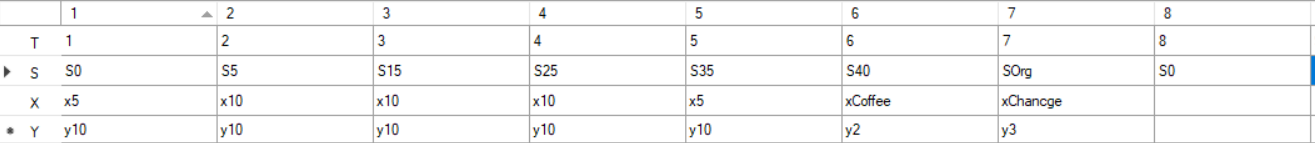
Объединенная функция

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X\S | S0 | S5 | S10 | S15 | S20 | S25 | S30 | S35 | S40 | S45 | S50 | Sorg |
| X5 | S5/Y5 | S5/Y5 | S5/Y5 | S5/Y5 | S5/Y5 | S5/Y5 | S5/Y5 | S5/Y5 | S5/Y5 | S5/Y5 | S0/YChancge |  |
| X10 | S10/Y10 | S10/Y10 | S10/Y10 | S10/Y10 | S10/Y10 | S10/Y10 | S10/Y10 | S10/Y10 | S10/Y10 | S0/YChancge | S0/YChancge |  |
| Xchocolate |  |  |  |  |  |  |  |  | Sorg/Y1 | Sorg/Y1 | Sorg/Y1 |  |
| XCoffee |  |  |  |  |  |  |  |  | Sorg/Y2 | Sorg/Y2 | Sorg/Y2 |  |
| XSugarPlus |  |  |  |  |  |  |  |  |  |  |  | Sorg/ YSugarPlus |
| XSugarMinus |  |  |  |  |  |  |  |  |  |  |  | Sorg/ YSugarMinus |
| XChancge |  |  |  |  |  |  |  |  |  |  |  | S0/Y3 |

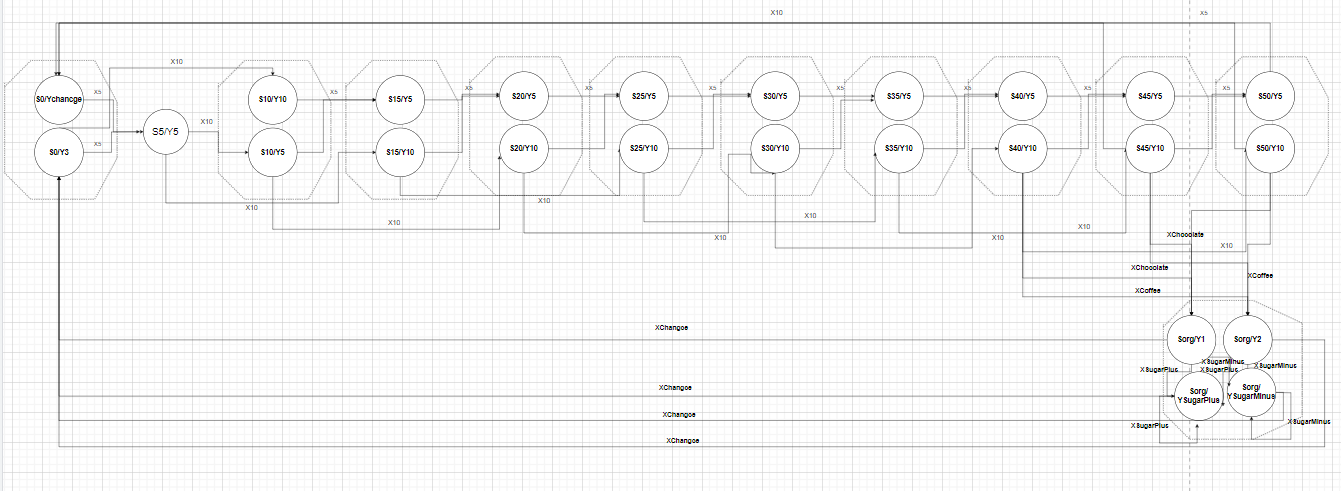
Граф автомата Мили



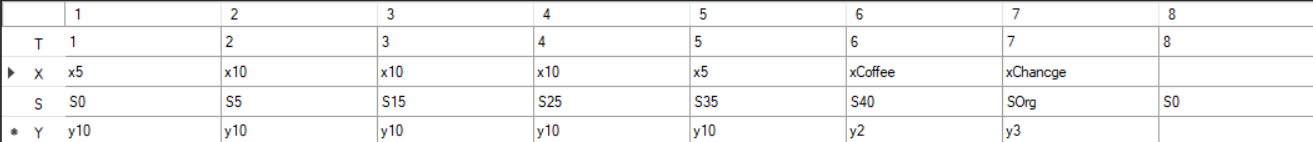
Протокол работы



Граф автомата Мура



Протокол работы



5. Программная модель

Реализация с помощью программного кода:

#pragma once

#include "State.cpp"

namespace CoffeeMachine {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для MainWindow

/// </summary>

public ref class MainWindow : public System::Windows::Forms::Form

{

public:

MainWindow(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MainWindow()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Button^ stepButton;

private: System::Windows::Forms::Button^ autoStep;

private: System::Windows::Forms::DataGridView^ exit;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column2;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column3;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column4;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column5;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column6;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column7;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column8;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column9;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Column10;

private: System::Windows::Forms::PictureBox^ Xin;

private: System::Windows::Forms::PictureBox^ Yin;

private: System::Windows::Forms::PictureBox^ Sin;

private: System::Windows::Forms::PictureBox^ imgAuto;

private: System::Windows::Forms::TextBox^ XInput1;

private: System::Windows::Forms::TextBox^ XInput2;

private: System::Windows::Forms::TextBox^ XInput3;

private: System::Windows::Forms::TextBox^ XInput4;

private: System::Windows::Forms::TextBox^ XInput5;

private: System::Windows::Forms::TextBox^ XInput6;

private: System::Windows::Forms::TextBox^ XInput7;

private: System::Windows::Forms::TextBox^ XInput8;

private: System::Windows::Forms::TextBox^ XInput9;

private: System::Windows::Forms::TextBox^ XInput10;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::Label^ label7;

private: System::Windows::Forms::Label^ label8;

private: System::Windows::Forms::Label^ label9;

private: System::Windows::Forms::Label^ label10;

private: System::Windows::Forms::Button^ clearInputs;

protected:

protected:

protected:

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->stepButton = (gcnew System::Windows::Forms::Button());

this->autoStep = (gcnew System::Windows::Forms::Button());

this->exit = (gcnew System::Windows::Forms::DataGridView());

this->Column1 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column2 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column3 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column4 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column5 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column6 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column7 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column8 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column9 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Column10 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Xin = (gcnew System::Windows::Forms::PictureBox());

this->Yin = (gcnew System::Windows::Forms::PictureBox());

this->Sin = (gcnew System::Windows::Forms::PictureBox());

this->imgAuto = (gcnew System::Windows::Forms::PictureBox());

this->XInput1 = (gcnew System::Windows::Forms::TextBox());

this->XInput2 = (gcnew System::Windows::Forms::TextBox());

this->XInput3 = (gcnew System::Windows::Forms::TextBox());

this->XInput4 = (gcnew System::Windows::Forms::TextBox());

this->XInput5 = (gcnew System::Windows::Forms::TextBox());

this->XInput6 = (gcnew System::Windows::Forms::TextBox());

this->XInput7 = (gcnew System::Windows::Forms::TextBox());

this->XInput8 = (gcnew System::Windows::Forms::TextBox());

this->XInput9 = (gcnew System::Windows::Forms::TextBox());

this->XInput10 = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label6 = (gcnew System::Windows::Forms::Label());

this->label7 = (gcnew System::Windows::Forms::Label());

this->label8 = (gcnew System::Windows::Forms::Label());

this->label9 = (gcnew System::Windows::Forms::Label());

this->label10 = (gcnew System::Windows::Forms::Label());

this->clearInputs = (gcnew System::Windows::Forms::Button());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->exit))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->Xin))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->Yin))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->Sin))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->imgAuto))->BeginInit();

this->SuspendLayout();

//

// stepButton

//

this->stepButton->Location = System::Drawing::Point(797, 490);

this->stepButton->Name = L"stepButton";

this->stepButton->Size = System::Drawing::Size(75, 23);

this->stepButton->TabIndex = 0;

this->stepButton->Text = L"Шаг";

this->stepButton->UseVisualStyleBackColor = true;

this->stepButton->Click += gcnew System::EventHandler(this, &MainWindow::stepButton\_Click);

//

// autoStep

//

this->autoStep->Location = System::Drawing::Point(896, 490);

this->autoStep->Name = L"autoStep";

this->autoStep->Size = System::Drawing::Size(127, 23);

this->autoStep->TabIndex = 1;

this->autoStep->Text = L"Автоматически";

this->autoStep->UseVisualStyleBackColor = true;

this->autoStep->Click += gcnew System::EventHandler(this, &MainWindow::autoStep\_Click);

//

// exit

//

this->exit->AccessibleDescription = L"4";

this->exit->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->exit->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(10) {

this->Column1, this->Column2,

this->Column3, this->Column4, this->Column5, this->Column6, this->Column7, this->Column8, this->Column9, this->Column10

});

this->exit->Cursor = System::Windows::Forms::Cursors::Default;

this->exit->Location = System::Drawing::Point(2, 531);

this->exit->Name = L"exit";

this->exit->RowHeadersBorderStyle = System::Windows::Forms::DataGridViewHeaderBorderStyle::None;

this->exit->RowHeadersWidth = 51;

this->exit->RowTemplate->Height = 24;

this->exit->Size = System::Drawing::Size(1053, 177);

this->exit->TabIndex = 2;

//

// Column1

//

this->Column1->HeaderText = L"1";

this->Column1->MinimumWidth = 6;

this->Column1->Name = L"Column1";

this->Column1->Width = 125;

//

// Column2

//

this->Column2->HeaderText = L"2";

this->Column2->MinimumWidth = 6;

this->Column2->Name = L"Column2";

this->Column2->Width = 125;

//

// Column3

//

this->Column3->HeaderText = L"3";

this->Column3->MinimumWidth = 6;

this->Column3->Name = L"Column3";

this->Column3->Width = 125;

//

// Column4

//

this->Column4->HeaderText = L"4";

this->Column4->MinimumWidth = 6;

this->Column4->Name = L"Column4";

this->Column4->Width = 125;

//

// Column5

//

this->Column5->HeaderText = L"5";

this->Column5->MinimumWidth = 6;

this->Column5->Name = L"Column5";

this->Column5->Width = 125;

//

// Column6

//

this->Column6->HeaderText = L"6";

this->Column6->MinimumWidth = 6;

this->Column6->Name = L"Column6";

this->Column6->Width = 125;

//

// Column7

//

this->Column7->HeaderText = L"7";

this->Column7->MinimumWidth = 6;

this->Column7->Name = L"Column7";

this->Column7->Width = 125;

//

// Column8

//

this->Column8->HeaderText = L"8";

this->Column8->MinimumWidth = 6;

this->Column8->Name = L"Column8";

this->Column8->Width = 125;

//

// Column9

//

this->Column9->HeaderText = L"9";

this->Column9->MinimumWidth = 6;

this->Column9->Name = L"Column9";

this->Column9->Width = 125;

//

// Column10

//

this->Column10->HeaderText = L"10";

this->Column10->MinimumWidth = 6;

this->Column10->Name = L"Column10";

this->Column10->Width = 125;

//

// Xin

//

this->Xin->Location = System::Drawing::Point(1124, -1);

this->Xin->Name = L"Xin";

this->Xin->Size = System::Drawing::Size(360, 304);

this->Xin->SizeMode = System::Windows::Forms::PictureBoxSizeMode::StretchImage;

this->Xin->TabIndex = 3;

this->Xin->TabStop = false;

//

// Yin

//

this->Yin->Location = System::Drawing::Point(1124, 309);

this->Yin->Name = L"Yin";

this->Yin->Size = System::Drawing::Size(368, 345);

this->Yin->SizeMode = System::Windows::Forms::PictureBoxSizeMode::StretchImage;

this->Yin->TabIndex = 4;

this->Yin->TabStop = false;

//

// Sin

//

this->Sin->Location = System::Drawing::Point(858, -1);

this->Sin->Name = L"Sin";

this->Sin->Size = System::Drawing::Size(269, 363);

this->Sin->SizeMode = System::Windows::Forms::PictureBoxSizeMode::StretchImage;

this->Sin->TabIndex = 5;

this->Sin->TabStop = false;

//

// imgAuto

//

this->imgAuto->Location = System::Drawing::Point(2, -1);

this->imgAuto->Name = L"imgAuto";

this->imgAuto->Size = System::Drawing::Size(782, 376);

this->imgAuto->SizeMode = System::Windows::Forms::PictureBoxSizeMode::StretchImage;

this->imgAuto->TabIndex = 6;

this->imgAuto->TabStop = false;

//

// XInput1

//

this->XInput1->Location = System::Drawing::Point(12, 444);

this->XInput1->Name = L"XInput1";

this->XInput1->Size = System::Drawing::Size(100, 22);

this->XInput1->TabIndex = 7;

//

// XInput2

//

this->XInput2->Location = System::Drawing::Point(118, 444);

this->XInput2->Name = L"XInput2";

this->XInput2->Size = System::Drawing::Size(100, 22);

this->XInput2->TabIndex = 8;

//

// XInput3

//

this->XInput3->Location = System::Drawing::Point(224, 444);

this->XInput3->Name = L"XInput3";

this->XInput3->Size = System::Drawing::Size(100, 22);

this->XInput3->TabIndex = 9;

//

// XInput4

//

this->XInput4->Location = System::Drawing::Point(330, 444);

this->XInput4->Name = L"XInput4";

this->XInput4->Size = System::Drawing::Size(100, 22);

this->XInput4->TabIndex = 10;

//

// XInput5

//

this->XInput5->Location = System::Drawing::Point(436, 444);

this->XInput5->Name = L"XInput5";

this->XInput5->Size = System::Drawing::Size(100, 22);

this->XInput5->TabIndex = 11;

//

// XInput6

//

this->XInput6->Location = System::Drawing::Point(542, 444);

this->XInput6->Name = L"XInput6";

this->XInput6->Size = System::Drawing::Size(100, 22);

this->XInput6->TabIndex = 12;

//

// XInput7

//

this->XInput7->Location = System::Drawing::Point(648, 444);

this->XInput7->Name = L"XInput7";

this->XInput7->Size = System::Drawing::Size(100, 22);

this->XInput7->TabIndex = 13;

//

// XInput8

//

this->XInput8->Location = System::Drawing::Point(754, 444);

this->XInput8->Name = L"XInput8";

this->XInput8->Size = System::Drawing::Size(100, 22);

this->XInput8->TabIndex = 14;

//

// XInput9

//

this->XInput9->Location = System::Drawing::Point(860, 444);

this->XInput9->Name = L"XInput9";

this->XInput9->Size = System::Drawing::Size(100, 22);

this->XInput9->TabIndex = 15;

//

// XInput10

//

this->XInput10->Location = System::Drawing::Point(966, 444);

this->XInput10->Name = L"XInput10";

this->XInput10->Size = System::Drawing::Size(100, 22);

this->XInput10->TabIndex = 16;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(60, 422);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(14, 16);

this->label1->TabIndex = 17;

this->label1->Text = L"1";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(166, 422);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(14, 16);

this->label2->TabIndex = 18;

this->label2->Text = L"2";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(268, 422);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(14, 16);

this->label3->TabIndex = 19;

this->label3->Text = L"3";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(379, 422);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(14, 16);

this->label4->TabIndex = 20;

this->label4->Text = L"4";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(476, 422);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(14, 16);

this->label5->TabIndex = 21;

this->label5->Text = L"5";

//

// label6

//

this->label6->AutoSize = true;

this->label6->Location = System::Drawing::Point(583, 422);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(14, 16);

this->label6->TabIndex = 22;

this->label6->Text = L"6";

//

// label7

//

this->label7->AutoSize = true;

this->label7->Location = System::Drawing::Point(689, 422);

this->label7->Name = L"label7";

this->label7->Size = System::Drawing::Size(14, 16);

this->label7->TabIndex = 23;

this->label7->Text = L"7";

//

// label8

//

this->label8->AutoSize = true;

this->label8->Location = System::Drawing::Point(794, 422);

this->label8->Name = L"label8";

this->label8->Size = System::Drawing::Size(14, 16);

this->label8->TabIndex = 24;

this->label8->Text = L"8";

//

// label9

//

this->label9->AutoSize = true;

this->label9->Location = System::Drawing::Point(903, 422);

this->label9->Name = L"label9";

this->label9->Size = System::Drawing::Size(14, 16);

this->label9->TabIndex = 25;

this->label9->Text = L"9";

//

// label10

//

this->label10->AutoSize = true;

this->label10->Location = System::Drawing::Point(997, 422);

this->label10->Name = L"label10";

this->label10->Size = System::Drawing::Size(21, 16);

this->label10->TabIndex = 26;

this->label10->Text = L"10";

//

// clearInputs

//

this->clearInputs->Location = System::Drawing::Point(987, 369);

this->clearInputs->Name = L"clearInputs";

this->clearInputs->Size = System::Drawing::Size(93, 23);

this->clearInputs->TabIndex = 27;

this->clearInputs->Text = L"Очистить";

this->clearInputs->UseVisualStyleBackColor = true;

this->clearInputs->Click += gcnew System::EventHandler(this, &MainWindow::clearInputs\_Click);

//

// MainWindow

//

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::SystemColors::WindowFrame;

this->ClientSize = System::Drawing::Size(1485, 720);

this->Controls->Add(this->clearInputs);

this->Controls->Add(this->label10);

this->Controls->Add(this->label9);

this->Controls->Add(this->label8);

this->Controls->Add(this->label7);

this->Controls->Add(this->label6);

this->Controls->Add(this->label5);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->XInput10);

this->Controls->Add(this->XInput9);

this->Controls->Add(this->XInput8);

this->Controls->Add(this->XInput7);

this->Controls->Add(this->XInput6);

this->Controls->Add(this->XInput5);

this->Controls->Add(this->XInput4);

this->Controls->Add(this->XInput3);

this->Controls->Add(this->XInput2);

this->Controls->Add(this->XInput1);

this->Controls->Add(this->imgAuto);

this->Controls->Add(this->Yin);

this->Controls->Add(this->Xin);

this->Controls->Add(this->exit);

this->Controls->Add(this->autoStep);

this->Controls->Add(this->stepButton);

this->Controls->Add(this->Sin);

this->Name = L"MainWindow";

this->Text = L"MainWindow";

this->Load += gcnew System::EventHandler(this, &MainWindow::MainWindow\_Load);

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->exit))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->Xin))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->Yin))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->Sin))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->imgAuto))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: static System::Drawing::Image^ firstPicture = System::Drawing::Image::FromFile("C:\\Users\\Влад\\Desktop\\Университет\\Теория автоматов\\Project1\\x.png");

private: static System::Drawing::Image^ secondPicture = System::Drawing::Image::FromFile("C:\\Users\\Влад\\Desktop\\Университет\\Теория автоматов\\Project1\\y.png");

private: static System::Drawing::Image^ thirdPicture = System::Drawing::Image::FromFile("C:\\Users\\Влад\\Desktop\\Университет\\Теория автоматов\\Project1\\s.png");

private: static System::Drawing::Image^ fourthPicture = System::Drawing::Image::FromFile("C:\\Users\\Влад\\Desktop\\Университет\\Теория автоматов\\Project1\\automat.png");

private: static int step = 0;

private: static State currentState = State::S0;

private: System::Void MainWindow\_Load(System::Object^ sender, System::EventArgs^ e) {

initPictures();

initExit();

}

private: System::Void stepButton\_Click(System::Object^ sender, System::EventArgs^ e) {

bool result = false;

switch (step)

{

case 0:

result = changeState(XInput1->Text);

break;

case 1:

result = changeState(XInput2->Text);

break;

case 2:

result = changeState(XInput3->Text);

break;

case 3:

result = changeState(XInput4->Text);

break;

case 4:

result = changeState(XInput5->Text);

break;

case 5:

result = changeState(XInput6->Text);

break;

case 6:

result = changeState(XInput7->Text);

break;

case 7:

result = changeState(XInput8->Text);

break;

case 8:

result = changeState(XInput9->Text);

break;

case 9:

result = changeState(XInput10->Text);

break;

default:

break;

}

if(result)

step++;

}

private: void initExit() {

this->exit->RowCount = 4;

this->exit->Rows[0]->HeaderCell->Value = "T";

this->exit->Rows[1]->HeaderCell->Value = "X";

this->exit->Rows[2]->HeaderCell->Value = "S";

this->exit->Rows[3]->HeaderCell->Value = "Y";

array<String^>^ rows = { "1", "2", "3", "4", "5", "6", "7", "8", "9", "10" };

for (int i = 0; i < this->exit->Rows[0]->Cells->Count; i++) {

this->exit->Rows[0]->Cells[i]->Value = rows[i];

}

}

private: void initPictures() {

this->Xin->Image = firstPicture;

this->Yin->Image = secondPicture;

this->Sin->Image = thirdPicture;

this->imgAuto->Image = fourthPicture;

}

private: void writeStateInTable(String^ input, String^ oldState, String^ newState, String^ output) {

this->exit->Rows[1]->Cells[step]->Value = input;

this->exit->Rows[2]->Cells[step]->Value = oldState;

if (step + 1 != 10)

this->exit->Rows[2]->Cells[step+1]->Value = newState;

this->exit->Rows[3]->Cells[step]->Value = output;

}

private: bool changeState(String^ xInput) {

switch (currentState)

{

case State::S0:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S0", "S10", "y10");

currentState = State::S10;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S0", "S5", "y10");

currentState = State::S5;

return true;

}

return false;

break;

case State::S5:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S5", "S15", "y10");

currentState = State::S15;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S5", "S10", "y10");

currentState = State::S10;

return true;

}

return false;

break;

case State::S10:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S10", "S20", "y10");

currentState = State::S20;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S10", "S15", "y10");

currentState = State::S15;

return true;

}

return false;

break;

case State::S15:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S15", "S25", "y10");

currentState = State::S25;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S15", "S20", "y10");

currentState = State::S20;

return true;

}

return false;

break;

case State::S20:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S20", "S30", "y10");

currentState = State::S30;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S20", "S25", "y10");

currentState = State::S25;

return true;

}

return false;

break;

case State::S25:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S25", "S35", "y10");

currentState = State::S35;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S25", "S30", "y10");

currentState = State::S30;

return true;

}

return false;

break;

case State::S30:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S30", "S40", "y10");

currentState = State::S40;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S30", "S35", "y10");

currentState = State::S35;

return true;

}

return false;

break;

case State::S35:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S35", "S45", "y10");

currentState = State::S45;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S35", "S40", "y10");

currentState = State::S40;

return true;

}

return false;

break;

case State::S40:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S40", "S50", "y10");

currentState = State::S50;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S40", "S45", "y10");

currentState = State::S45;

return true;

}

else if (xInput->ToLower() == "xcoffee") {

writeStateInTable("xCoffee", "S40", "SOrg", "y2");

currentState = State::Sorg;

return true;

}

else if (xInput->ToLower() == "xchocolate") {

writeStateInTable("xChocolate", "S40", "SOrg", "y1");

currentState = State::Sorg;

return true;

}

return false;

break;

case State::S45:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S45", "S0", "y10");

currentState = State::S0;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S45", "S50", "y10");

currentState = State::S50;

return true;

}

else if (xInput->ToLower() == "xcoffee") {

writeStateInTable("xCoffee", "S45", "SOrg", "y2");

currentState = State::Sorg;

return true;

}

else if (xInput->ToLower() == "xchocolate") {

writeStateInTable("xChocolate", "S45", "SOrg", "y1");

currentState = State::Sorg;

return true;

}

return false;

break;

case State::S50:

if (xInput->ToLower() == "x10") {

writeStateInTable("x10", "S50", "S0", "y10");

currentState = State::S0;

return true;

}

else if (xInput->ToLower() == "x5") {

writeStateInTable("x5", "S50", "S0", "y10");

currentState = State::S0;

return true;

}

else if (xInput->ToLower() == "xcoffee") {

writeStateInTable("xCoffee", "S50", "SOrg", "y2");

currentState = State::Sorg;

return true;

}

else if (xInput->ToLower() == "xchocolate") {

writeStateInTable("xChocolate", "S50", "SOrg", "y1");

currentState = State::Sorg;

return true;

}

return false;

break;

case State::Sorg:

if (xInput->ToLower() == "xsugarplus") {

writeStateInTable("xSugarPlus", "SOrg", "SOrg", "ySugarPlus");

currentState = State::Sorg;

return true;

}

else if (xInput->ToLower() == "xsugarminus") {

writeStateInTable("xSugarMinus", "SOrg", "SOrg", "ySugarMinus");

currentState = State::Sorg;

return true;

}

else if (xInput->ToLower() == "xchancge") {

writeStateInTable("xChancge", "SOrg", "S0", "y3");

currentState = State::S0;

return true;

}

return false;

break;

default:

break;

return false;

}

}

private: System::Void clearInputs\_Click(System::Object^ sender, System::EventArgs^ e) {

this->XInput1->Text = "";

this->XInput2->Text = "";

this->XInput3->Text = "";

this->XInput4->Text = "";

this->XInput5->Text = "";

this->XInput6->Text = "";

this->XInput7->Text = "";

this->XInput8->Text = "";

this->XInput9->Text = "";

this->XInput10->Text = "";

this->exit->Rows->Clear();

step = 0;

currentState = State::S0;

initExit();

}

private: System::Void autoStep\_Click(System::Object^ sender, System::EventArgs^ e) {

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

stepButton\_Click(sender, e);

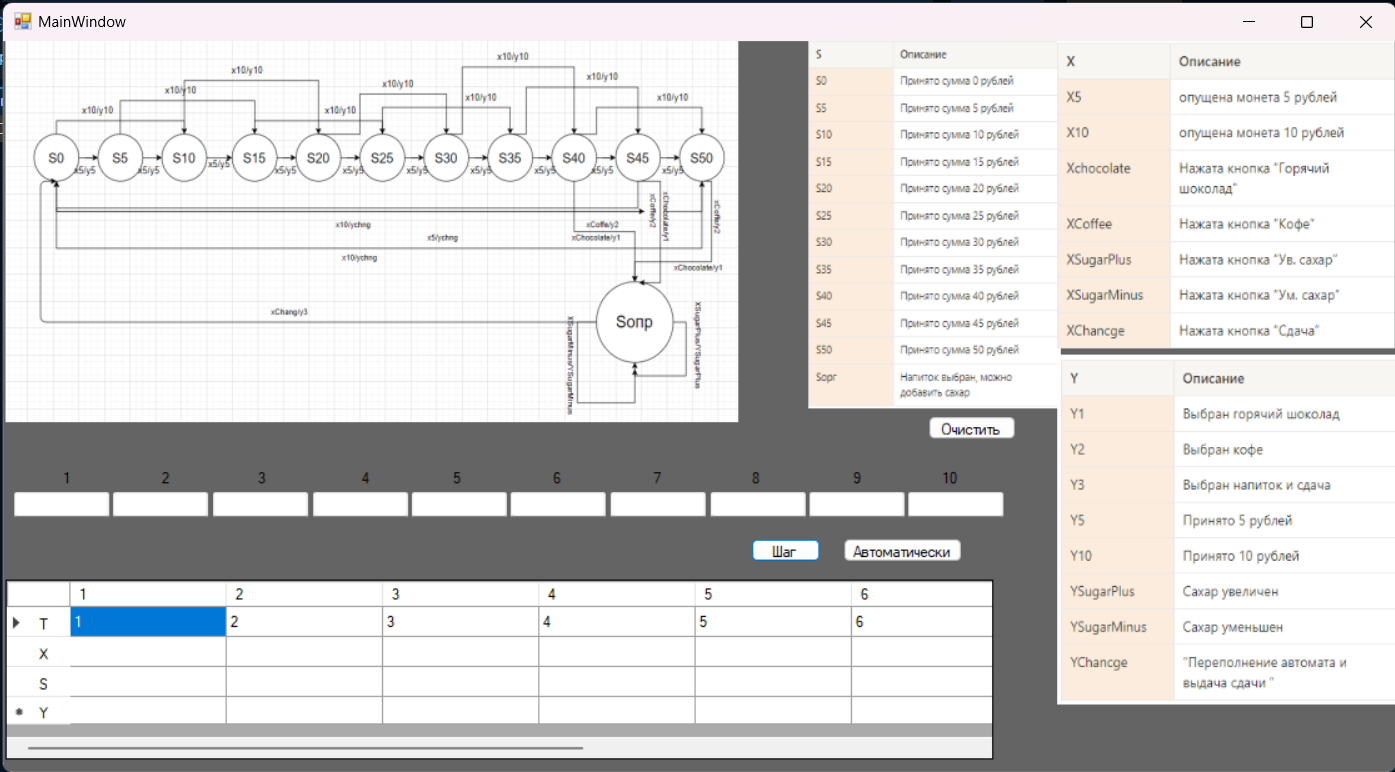
stepButton\_Click(sender, e);

}

};

}

Программная модель:



Вывод:

В рамках первой лабораторной работы по теории автоматов, мы изучили пример конечного детерминированного автомата Мили, который моделирует функционирование автомата выдачи напитков. В этой работе, мы разобрали, как данный автомат может быть описан с помощью состояний, событий и переходов между ними.

Начало работы автомата выдачи напитков происходит с момента включения, когда пользователь сталкивается с главным экраном (s0). Затем он взаимодействует с автоматом, вставляя монеты (x5, x10), выбирая напиток (Xchocolate, XCoffee) и выбирая различные операции, такие как выбор количества сахара или запрос сдачи. Автомат переходит между разными состояниями в зависимости от входных событий и выполняет соответствующие операции, такие как выбор напитка и выдача сдачи.

Мы также создали таблицы, описывающие переходы и выходы для автомата Мили, и провели проверку его работы с использованием конкретной последовательности событий. Это позволило нам более глубоко понять, как автомат реагирует на разнообразные входные данные и какие события он генерирует в ответ.

Данная лабораторная работа также подчеркнула важность использования формальных методов для моделирования и анализа сложных систем, подобных автомату. Теория автоматов предоставляет нам необходимые инструменты для понимания и оптимизации работы подобных систем.

В итоге, данная работа позволила нам лучше ознакомиться с функционированием автомата выдачи напитков с применением конечных автоматов Мили. Она также подтверждает важность формального описания систем для их более глубокого анализа и усовершенствования.