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)

ОТЧЕТ ПО ПРАКТИЧЕСКИМ РАБОТАМ

по дисциплине

«Проектирование клиент-серверных систем»

Практические задания №1-8

Выполнил студент группы ИКБО-10-19 Дараган Ф.А.

Принял ассистент Мельников Д.А.

Практические работы выполнены «__» _____ 2022 г.

(подпись студента)

«Зачтено» «__» _____ 2022 г.

(подпись преподавателя)

Москва 2022

Оглавление

Практическая работа № 1: «Проектирование архитектуры и дизайна клиент-серверных систем»

Цель работы

Практически освоить основные правила проектирования архитектуры и дизайна систем. Практически потренироваться в составлении интерфейсов клиентской части.

Задания

1. Даны действительные числа x,y. Вычислить значение функции z=log(x-y)- x/y
2. Вывести на печать переменные A,B,C в порядке их возрастания
3. Выяснить, существует ли треугольник с координатами вершин A(x1,y1), B(x2,y2),C(x3,y3).
4. Даны площадь квадрата S1 и круга S2. Определить поместится ли круг в квадрат и наоборот
5. Даны действительные числа A,B,C,D. Выяснить, можно ли уместить прямоугольник со сторонами A,B внутри прямоугольника со сторонами C,D.

Ход работы

Для каждой задачи была создана своя форма с пользовательским интерфейсом и обработкой вводимых данных согласно задаче.

Задача 1

На листинге 1 представлен код для решения задачи 1.

Листинг 1 — Класс Task7Form

```
public partial class Task7Form : Form
{
    public double xValue = 0;
    public double yValue = 0;
    public Task7Form() {}
    private void xInput_TextChanged(object sender, EventArgs e)
    {
```

```
        xValue = double.Parse(xInput.Text);
    }
    private void buttonCalculate_Click(object sender, EventArgs e)
    {
        var z = Math.Log(xValue - yValue) - xValue / yValue;
        resultLabel.Text = z.ToString();
    }
    private void yInput_TextChanged(object sender, EventArgs e)
    {
        yValue = double.Parse(yInput.Text);
    }
}
```

На рисунке 1 показан результат запуска данной формы.

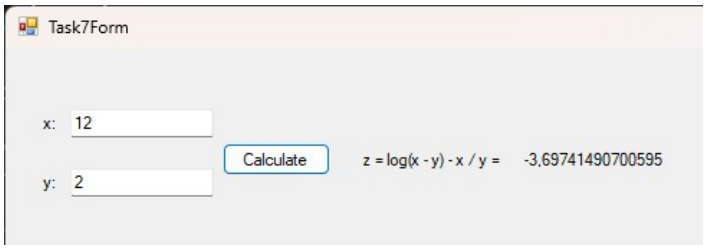

Рисунок 1. Скриншот формы для 7 задания

Задача 2

На листинге 2 представлен код для решения задачи 2.

Листинг 2 — Класс Task10Form

```
public partial class Task10Form : Form
{
    public double aValue;
    public double bValue;
    public double cValue;
    public Task10Form() {}
    private string getSortedValues()
    {
        double[] arr = { aValue, bValue, cValue };
        Array.Sort(arr);
        return String.Join(", ", arr);
    }
    private void sortButton_Click(object sender, EventArgs e)
```

```
    {
        MessageBox.Show(
            getSortedValues(),
            "Переменные в порядке возрастания",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information,
            MessageBoxDefaultButton.Button1,
            MessageBoxOptions.DefaultDesktopOnly
        );
    }
    private void aInput_TextChanged(object sender, EventArgs e)
    {
        aValue = double.Parse(aInput.Text);
    }
    private void bInput_TextChanged(object sender, EventArgs e)
    {
        bValue= double.Parse(bInput.Text);
    }
    private void cInput_TextChanged(object sender, EventArgs e)
    {
        cValue= double.Parse(cInput.Text);
    }
}
```

На рисунке 2 показан результат запуска данной формы.

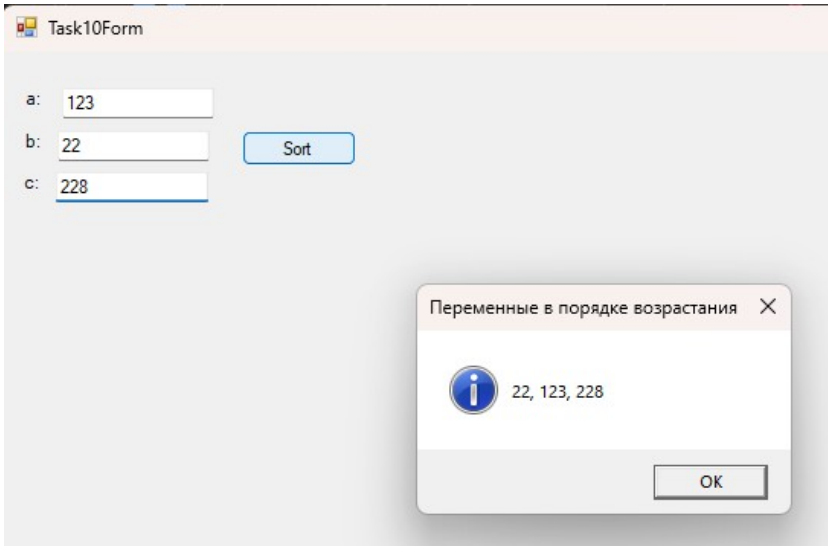

Рисунок 2. Скриншот формы для 10 задания

Задача 3

На листинге 3 представлен код для решения задачи 3.

Листинг 3 — Класс Task13Form

```
public partial class Task13Form : Form
{
    private double x1 = 0;
    private double y1 = 0;
    private double x2 = 0;
    private double y2 = 0;
    private double x3 = 0;
    private double y3 = 0;

    public Task13Form() {}

    private bool isExists()
    {
        double a = Math.Sqrt(Math.Pow((x2 - x3), 2) + Math.Pow((y2 - y3),
2));
        double b = Math.Sqrt(Math.Pow((x1 - x3), 2) + Math.Pow((y1 - y3),
2));
        double c = Math.Sqrt(Math.Pow((x1 - x2), 2) + Math.Pow((y1 - y2),
2));
        return (a + b) > c && (a + c) > b && (b + c) > a;
    }
    private void x1Input_TextChanged(object sender, EventArgs e)
    {
        x1 = double.Parse(x1Input.Text);
    }
    private void x2Input_TextChanged(object sender, EventArgs e)
    {
        x2 = double.Parse(x2Input.Text);
    }
    private void x3Input_TextChanged(object sender, EventArgs e)
    {
        x3 = double.Parse(x3Input.Text);
    }
    private void y1Input_TextChanged(object sender, EventArgs e)
    {
```

```
        y1 = double.Parse(y1Input.Text);
    }
    private void y2Input_TextChanged(object sender, EventArgs e)
    {
        y2 = double.Parse(y2Input.Text);
    }
    private void y3Input_TextChanged(object sender, EventArgs e)
    {
        y3 = double.Parse(y3Input.Text);
    }
    private void existsButton_Click(object sender, EventArgs e)
    {
        resultLabel.Text = isExists() ? "Yes" : "No";
    }
}
```

На рисунке 3 показан результат запуска данной формы.

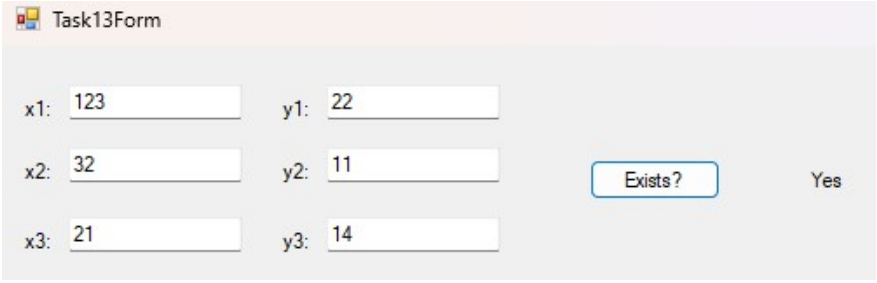

Рисунок 3. Скриншот формы для 13 задания

Задача 4

На листинге 4 представлен код для решения задачи 4.

Листинг 4 — Класс Task16Form

```
public partial class Task16Form : Form
{
    private double circleRadius = 0;
    private double squareLenght = 0;
    public Task16Form() {}
    private bool isFit()
    {
        return circleRadius * 2 <= squareLenght;
    }
```

```
    private void circleInput_TextChanged(object sender, EventArgs e)
    {
        double S = Double.Parse(circleInput.Text);
        circleRadius = Math.Sqrt(S / Math.PI);
    }
    private void squareInput_TextChanged(object sender, EventArgs e)
    {
        double S = Double.Parse(squareInput.Text);
        squareLenght = Math.Sqrt(S);
    }
    private void fitButton_Click(object sender, EventArgs e)
    {
        answerLabel.Text = isFit() ? "Yes" : "No";
    }
}
```

На рисунке 4 показан результат запуска данной формы.

Рисунок 4. Скриншот формы для 16 задания

Задача 5

На листинге 5 представлен код для решения задачи 5.

Листинг 5 — Класс Task19Form

```
public partial class Task19Form : Form
{
    private double aValue;
    private double bValue;
    private double cValue;
    private double dValue;
    public Task19Form() {}
    private bool isFit()
    {
```

```
        return (aValue <= cValue && bValue <= dValue) || (aValue <= dValue &&
bValue <= cValue);
    }
    private void aInput_TextChanged(object sender, EventArgs e)
    {
        aValue = double.Parse(aInput.Text);
    }
    private void bInput_TextChanged(object sender, EventArgs e)
    {
        bValue= double.Parse(bInput.Text);
    }
    private void cInput_TextChanged(object sender, EventArgs e)
    {
        cValue= double.Parse(cInput.Text);
    }
    private void dInput_TextChanged(object sender, EventArgs e)
    {
        dValue= double.Parse(dInput.Text);
    }
    private void fitButton_Click(object sender, EventArgs e)
    {
        resultLabel.Text = isFit() ? "Yes" : "No";
    }
}
```

На рисунке 5 показан результат запуска данной формы.

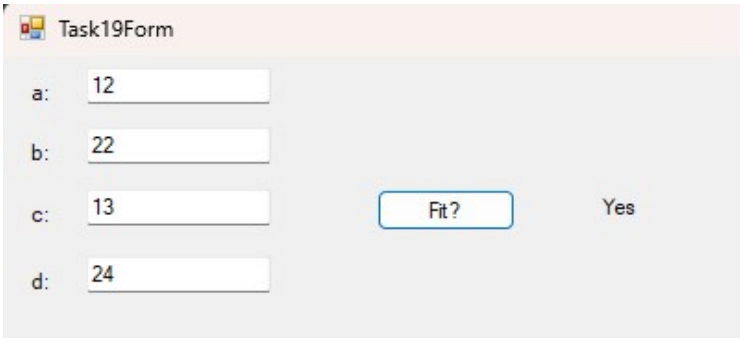

Рисунок 5. Скриншот формы для 19 задания

Вывод

В результате работы была создана программа, имеющая базовый интерфейс и решающая простые задачи.

Практическая работа № 2: «Простые двухуровневые клиент-серверные системы в Java»

Цель работы

Выработать умения и навыки по составлению программ простых одноуровневых клиент-серверных систем на основе технологии сокетов.

Задания

1. Найти периметр десятиугольника, координаты вершин которого заданы. Определить процедуру вычисления расстояния между двумя точками, заданными своими координатами, которые передаются функции в качестве параметров из основной программы.
2. Составить программу вывода на экран всех натуральных чисел, не превосходящих N и делящихся на каждую из своих цифр. Описать соответствующую функцию, получающую из основной программы в качестве параметра натуральное число и возвращающую TRUE, если оно удовлетворяет указанному условию.
3. Используя подпрограмму - функцию, составить программу для возведения чисел в целую положительную степень. Число передаются функции в качестве параметра из основной программы. Расчет вести для чисел, пока не будет введено число, равное 0.
4. Задав функцию, рассчитать и вывести на печать максимальные значения в трех парах чисел, вводимых с клавиатуры. Пара чисел передается функции в качестве параметра.
5. Даны числа A, B, C. Получить с использованием функции пользователя наименьшее значение. Числа передаются функции из основной программы в качестве параметров.

Ход работы

В ходе выполнения работы были созданы 2 основных класса для клиента и сервера, которые общаются между собой при помощи сокетов по принципу «сообщение-ответ». На серверной части были созданы дополнительные классы для обработки логики задач. К сожалению, описание логики программы вышло громоздким, но целью данной работы было

ознакомление с технологией сокетов, а не архитектурными решениями для сокетов.

Создание клиент-серверной системы на сокетах

На листинге 6 показан код клиента, который ждет сообщение с сервера, затем выводит его в консоль, ждет ввода в консоль, отправляет введенный текст на сервер и так по кругу.

Листинг 6 — Класс Client

```
public class Client {

    public static final int PORT = 22812;

    private Socket socket;

    public Client() throws IOException, UnknownHostException {
        socket = new Socket("localhost", PORT);
    }

    public void run() {
        try (
            var reader = new BufferedReader(new
InputStreamReader(System.in));
            var in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            var out = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));
        ) {
            System.out.println("Подключен к серверу...");
            while (true) {
                var responseStr = in.readLine();
                System.out.println(responseStr);
                var requestStr = reader.readLine();
                out.write(requestStr + "\n");
                out.flush();
            }
        } catch (IOException e) {
            System.out.println(e);
        }
```

```
    }

    public static void main(String[] args) {
        try {
            var client = new Client();
            client.run();
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

На листинге 7 показан код сервера, большая часть логики опущена в виду ее громоздкости.

Листинг 7 — Класс Server

```
public class Server {

    public static final int PORT = 22812;
    private static ServerSocket server;

    public Server() throws IOException {
        server = new ServerSocket(PORT);
    }

    private void print(BufferedWriter out, String message) throws
IOException{
        out.write(message + "\n");
        out.flush();
    }

    public void run() {
        try (
            var socket = server.accept();
            var in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            var out = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));
```

```
        ) {
            System.out.println("Подключение к серверу...");
            while (true) {
                try {
                    print(out, "Введите номер задания: [7, 10, 13, 16, 19]");
                    var numberOfTask = readNum(in);
                    switch (numberOfTask) {
                        // …
                    }
                    in.readLine();
                } catch (IOException e) {
                    // ...
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        try {
            var server = new Server();
            server.run();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Задача 7

На листинге 8 показан вспомогательный класс для хранения точек.

Листинг 8 — Класс Dot

```
public  class Dot {
    public int x;
    public int y;

    public Dot(int x, int y) {
```

```
        this.x = x;
        this.y = y;
    }
}
```

На листинге 9 показан код для решения задачи 7, для нахождения площади была использована формула площади Гаусса.

Листинг 9 — Класс Task7

```
public class Task7 {

    private List<Dot> dots;

    public Task7(List<Dot> dots) {
        this.dots = dots;
    }

    public double calcSquare() {
        int n = dots.size();
        double sum = 0;
        for (int i = 0; i < n - 1; i++) {
            sum += dots.get(i).x * dots.get(i + 1).y;
        }
        sum += dots.get(n - 1).x * dots.get(0).y;
        for (int i = 0; i < n - 1; i++) {
            sum += dots.get(i + 1).x * dots.get(i).y;
        }
        sum -= dots.get(0).x * dots.get(n - 1).y;
        return 1/2 * Math.abs(sum);
    }

    public double distanceBetweenDots(int i1, int i2) {
        Dot dot1 = dots.get(i1);
        Dot dot2 = dots.get(i2);
        return Math.sqrt(Math.pow(dot1.x - dot2.x, 2) + Math.pow(dot1.y -
dot2.y, 2));
    }
}
```

Задача 2

На листинге 10 показан код для решения задачи 10.

Листинг 10 — Класс Task10

```
public class Task10 {

    private int N;

    public Task10(int N) {
        this.N = N;
    }

    public List<Integer> naturalList(int n) {
        return IntStream.range(1, n)
            .filter(x -> !String.valueOf(x).contains("0"))
            .filter(x -> String.valueOf(x)
                    .chars()
                    .map(i -> i - 48)
                    .distinct()
                    .filter(i -> x % i != 0)
                    .count() == 0
                )
            .boxed()
            .toList();
    }

    public boolean checkNumber(int i) {
        return naturalList(i + 1).stream().filter(I -> i ==
I).findFirst().isPresent();
    }

    public List<Integer> getNumbers() {
        return naturalList(N);
    }
}
```

Задача 3

На листинге 11 показан код для решения задачи 13.

Листинг 11 — Класс Task13

```
public class Task13 {
    public double pow(int number, int power) {
        return Math.pow(number, power);
    }
}
```

Задача 4

На листинге 12 показан код для решения задачи 16.

Листинг 12 — Класс Task16

```
public class Task16 {
    public List<Integer> pairsMax(List<List<Integer>> pairs) {
        return pairs.stream().map(
            (List<Integer> l) -> l.stream().max(Integer::compare).get()
        ).toList();
    }
}
```

Задача 5

На листинге 13 показан код для решения задачи 19.

Листинг 13 — Класс Task19

```
public class Task19 {
    public int listMin(List<Integer> pairs) {
        return pairs.stream().min(Integer::compare).get();
    }
}
```

Тестирование разработанной системы и задач

Для краткости взаимодействие между клиентом и сервером представлено на листинге 14.

Листинг 14 — Листинг консоли клиента

```
Подключен к серверу...
Введите номер задания: [7, 10, 13, 16, 19]
```

```
7
Введите номер подзадачи: [1 - ввести точки, 2 - вычислить расстояние]
1
Введите X координату 0 точки
1
Введите Y координату 0 точки
2
Введите X координату 1 точки
3
Введите Y координату 1 точки
4
Введите X координату 2 точки
2
Введите Y координату 2 точки
32
Введите X координату 3 точки
12
Введите Y координату 3 точки
332
Введите X координату 4 точки
21
Введите Y координату 4 точки
54
Введите X координату 5 точки
2312
Введите Y координату 5 точки
565
Введите X координату 6 точки
343
Введите Y координату 6 точки
76
Введите X координату 7 точки
12
Введите Y координату 7 точки
57
Введите X координату 8 точки
32
```

```
Введите Y координату 8 точки
55
Введите X координату 9 точки
767
Введите Y координату 9 точки
323
Площадь фигуры = 0.0

Введите номер задания: [7, 10, 13, 16, 19]
7
Введите номер подзадачи: [1 - ввести точки, 2 - вычислить расстояние]
2
Введите номер первой точки
7
Введите номер второй точки
8
Расстояние между точками = 20.09975124224178

Введите номер задания: [7, 10, 13, 16, 19]
10
Введите номер подзадачи: [1 - узнать числа, 2 - проверить число]
1
Введите лимит
223
1,2,3,4,5,6,7,8,9,11,12,15,22,24,33,36,44,48,55,66,77,88,99,111,112,115,122,1
24,126,128,132,135,144,155,162,168,175,184,212,216,222

Введите номер задания: [7, 10, 13, 16, 19]
10
Введите номер подзадачи: [1 - узнать числа, 2 - проверить число]
2
Введите число
222
Удовлетворяет

Введите номер задания: [7, 10, 13, 16, 19]
13
Введите число
```

```
12
Введите степень
3
Результат = 1728.0

Введите номер задания: [7, 10, 13, 16, 19]
16
Введите 1 число 0 пары
2
Введите 2 число 0 пары
4
Введите 1 число 1 пары
12
Введите 2 число 1 пары
23
Введите 1 число 2 пары
44
Введите 2 число 2 пары
32
Результат = 4, 23, 44

Введите номер задания: [7, 10, 13, 16, 19]
19
Введите число 0
23
Введите число 1
45
Введите число 2
7534
Результат = 23
```

Вывод

В результате работы была создана программа, реализующая клиент-серверную архитектуру и решающая поставленные задачи. Общая структура такова, что пользователь вводит данные в клиентской части, они отправляются на сервер, где производятся вычисления, и сервер возвращает новые данные клиенту.

Практическая работа № 3: «Простые двухуровневые клиент-серверные приложения в C#»

Цель работы

Выработать умения и навыки составлять типовые программы решения задач на выбранном языке программирования, снабженные элементами графического интерфейса пользователя в виде клиент-серверных систем.

Задачи

1. Напечатать значения функции y=ln(x-1/x),где значения x вводятся с клавиатуры. При вводе числа, не входящего в область определения функции, вычисления прекратить.
2. Дано натуральное число N. Получить наименьшее число вида 4k , большее N.
3. Определить, является ли натуральное число N степенью числа 3 или нет.
4. Вывести на печать отрицательные значения функции z=tg(x)+5cos(x-2) для x изменяющегося на отрезке [12, 1] с шагом 1,2.
5. Найти первый отрицательный член последовательности sin(tg(n/2)) для n изменяющегося на следующим образом: n=1,2,3... .

Ход работы

В ходе работы была использована архитектура, аналогичная практической работе 2, поменялся лишь язык программирования и задания.

Создание клиент-серверной системы на сокетах

На листинге 15 показан код для клиентской части.

Листинг 15 — Класс Client

```
public class Client
{
    private IPEndPoint remoteEndPoint;
    private Socket sender;
    public Client(int localPort)
    {
```

```
        try
        {
            sender = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
            IPHostEntry ipHostInfo = Dns.GetHostEntry(Dns.GetHostName());
            IPAddress ipAddress = ipHostInfo.AddressList.First(a =>
a.AddressFamily == AddressFamily.InterNetwork);
            remoteEndPoint = new IPEndPoint(ipAddress, localPort);
        }
        catch (SocketException e)
        {
            Console.WriteLine(e);
        }
    }
    public void send(string enteredText)
    {
        byte[] msg = Encoding.Unicode.GetBytes(enteredText);
        int bytesSent = sender.Send(msg);
        Console.WriteLine("Sent {0} bytes over the network", bytesSent);
    }
    public string getFromConsole(string helloMessage, bool newLine = false)
    {
        if (helloMessage != null)
        {
            if (newLine)
            {
                Console.WriteLine(helloMessage);
            }
            else
            {
                Console.Write(helloMessage);
            }
        }
        return Console.ReadLine() ?? "";
    }
    public string recieve()
    {
        byte[] bytes = new byte[1024];
```

```
        int bytesReceived = sender.Receive(bytes);
        return Encoding.Unicode.GetString(bytes, 0, bytesReceived);
    }
    public void run()
    {
        sender.Connect(remoteEndPoint);
        Console.WriteLine("Socket connected to {0}", sender.RemoteEndPoint);
        while (true)
        {
            var recievedText = recieve();
            Console.WriteLine(recievedText);
            var enteredText = getFromConsole("> ", false);
            send(enteredText);
        }
        sender.Shutdown(SocketShutdown.Both);
        sender.Close();
    }
    static int Main(string[] args)
    {
        var client = new Client(12312);
        client.run();
        return 0;
    }
}
```

На листинге 16 показан код для серверной части, часть логики опять пропущена для краткости.

Листинг 16 — Класс Server

```
internal class Server
{
    private IPEndPoint localEndPoint;
    private Socket listener;
    private Socket handler;
    public Server(int localPort)
    {
        try
        {
```

```
            listener = new Socket(IPAddress.Any.AddressFamily,
SocketType.Stream, ProtocolType.Tcp);
            IPHostEntry ipHostInfo = Dns.GetHostEntry(Dns.GetHostName());
            IPAddress ipAddress = ipHostInfo.AddressList.First(a =>
a.AddressFamily == AddressFamily.InterNetwork);
            localEndPoint = new IPEndPoint(ipAddress, localPort);
        }
        catch (SocketException e)
        {
            Console.WriteLine(e);
        }
    }
    public void send(string enteredText, bool withRecieve =  false)
    {
        if (withRecieve)
        {
            enteredText += (withRecieve ? "\n Введите что угодно, чтобы
продолжить" : "");
        }
        byte[] msg = Encoding.Unicode.GetBytes(enteredText);
        int bytesSent = handler.Send(msg);
        if (withRecieve)
        {
            recieve();
        }
    }
    public string recieve()
    {
        byte[] bytes = new byte[1024];
        int bytesReceived = handler.Receive(bytes);
        return Encoding.Unicode.GetString(bytes, 0, bytesReceived);
    }
    public void run()
    {
        listener.Bind(localEndPoint);
        listener.Listen(10);
        while (true)
        {
            handler = listener.Accept();
```

```
            Console.WriteLine("Client connected");
            while (handler.Connected)
            {
                try
                {
                    trashMethod();
                }
                catch (Exception e)
                {
                    Console.WriteLine(e);
                    send("Произошла ошибка", true);
                }
            }
        }
    }
    public void trashMethod()
    {
        send("Выберите номер задачи [7, 10, 13, 16, 19]");
        var numberOfTask = recieveInt();

        switch (numberOfTask)
        {
            //...
        }
    }
    static int Main(string[] args)
    {
        var server = new Server(12312);
        server.run();
        return 0;
    }
}
```

Задание 1

На листинге 17 показан код для решения 1 задачи.

Листинг 17 — Класс Task7

```
internal class Task7
{
    public static double calculate(double x)
    {
        return Math.Log((x - 1) / x);
    }
}
```

Задание 2

На листинге 18 показан код для решения 2 задачи.

Листинг 18 — Класс Task10

```
internal class Task10
{
    public static int getNearestNumber(int number)
    {
        int k = 0;
        while (true)
        {
            int a = (int)Math.Pow(4, k++);
            if (a > number)
                return a;
        }
    }
}
```

Задание 3

На листинге 19 показан код для решения 3 задачи.

Листинг 19 — Класс Task13

```
internal class Task13
{
    public static bool isThreePow(int number)
    {
```

```
        while (true)
        {
            if (number == 1)
            {
                return true;
            }

            if (number % 3 == 0)
            {
                number /= 3;
            }
            else
            {
                return false;
            }

        }
    }
}
```

Задание 4

На листинге 20 показан код для решения 4 задачи.

Листинг 20 — Класс Task16

```
internal class Task16
{
    public static List<double> values()
    {
        var list = new List<double>();
        for (double x = 12; x >= 1; x -= 1.2)
        {
            var z = Math.Tan(x) + 5 * Math.Cos(x - 2);
            list.Add(z);
        }
        return list;
    }
}
```

Задание 5

На листинге 21 показан код для решения 5 задачи.

Листинг 21 — Класс Task19

```
internal class Task19
{
    public static double firstNegative()
    {
        for (int n = 1; n < int.MaxValue; n++)
        {
            var z = Math.Sin(Math.Tan(n / 2));
            if (z < 0) {
                return z;
            }
        }
        return 0;
    }
}
```

Тестирование разработанной системы и задач

Для краткости взаимодействие между клиентом и сервером представлено на листинге 22.

Листинг 22 — Листинг консоли клиента

```
Socket connected to 172.18.160.1:12312
Выберите номер задачи [7, 10, 13, 16, 19]
> 7
Введите число(double):
> 22,12
Результат  = -0,04626171781607336
 Введите что угодно, чтобы продолжить
> 1
Выберите номер задачи [7, 10, 13, 16, 19]
> 10
Введите число(int):
> 154
Результат  = 256
 Введите что угодно, чтобы продолжить
```

```
> 1
Выберите номер задачи [7, 10, 13, 16, 19]
> 13
Введите число(int):
> 324
Результат  = Нет
 Введите что угодно, чтобы продолжить
> 1
Выберите номер задачи [7, 10, 13, 16, 19]
> 16
Результат  = -
4,831217574043842,0,992323239577785,1,4333368194961413,3,320213867563066,3,64
7204296557536,-3,5592242957028,-
16,09598235758598,0,34746911847845474,3,6892906803410144,6,05568516886216
 Введите что угодно, чтобы продолжить
> 1
Выберите номер задачи [7, 10, 13, 16, 19]
> 19
Результат  = -0,8172096612475641
 Введите что угодно, чтобы продолжить
>
```

Вывод

В результате работы была создана программа, реализующая клиент-серверную архитектуру и решающая поставленные задачи. Общая структура такова, что пользователь вводит данные в клиентской части, они отправляются на сервер, где производятся вычисления, и сервер возвращает новые данные клиенту.

Практическая работа № 4: «Построение модели деятельности предприятия. Нотация IDEF0»

Цель работы

1. Знакомство с графической нотацией формализации и описания бизнес-процессов IDEF0. Знакомство с понятием функциональной модели AS-IS («как есть»).
2. Описание и построение функциональной модели AS-IS выбранной предметной области с применением нотации IDEF0.

Задание

Для заданной предметной области разработать модель AS-IS. Вы можете выбрать один из вариантов процессов, описанных в приложении, или предложить свой вариант. Модель должна содержать:

1. не менее трёх уровней;
2. на диаграмме 1-го уровня (A0) должно быть не менее 4-х функциональных блоков;
3. на диаграмме 2-го и далее уровнях должно быть не менее 2-х функциональных блоков.

Ход работы

Модель AS-IS – это модель «как есть», т. е. модель уже существующего процесса/функции. Анализ процессов является обязательной частью любого проекта создания или развития системы. Построение функциональной модели AS-IS позволяет четко зафиксировать, какие процессы осуществляются на предприятии, какие информационные объекты используются при выполнении функций различного уровня детализации.

Модель AS-IS показывает зоны ответственности исполнителей процесса и ход самого процесса («кто что сделал», как взаимосвязаны этапы между собой и как каждый этап влияет на конечный результат). Функциональная модель AS-IS является отправной точкой для анализа потребностей предприятия, выявления проблем и «узких» мест, разработки проекта совершенствования деловых процессов. Анализ функциональной модели AS-IS позволяет понять, в чем заключается проблема, в чем будут

состоять преимущества новых процессов и каким изменениям подвергнется существующая структура организации процесса в результате оптимизации.

Была выбрана предметная область «Деятельность библиотеки».

На рисунке 6 показана диаграмма нулевого уровня.

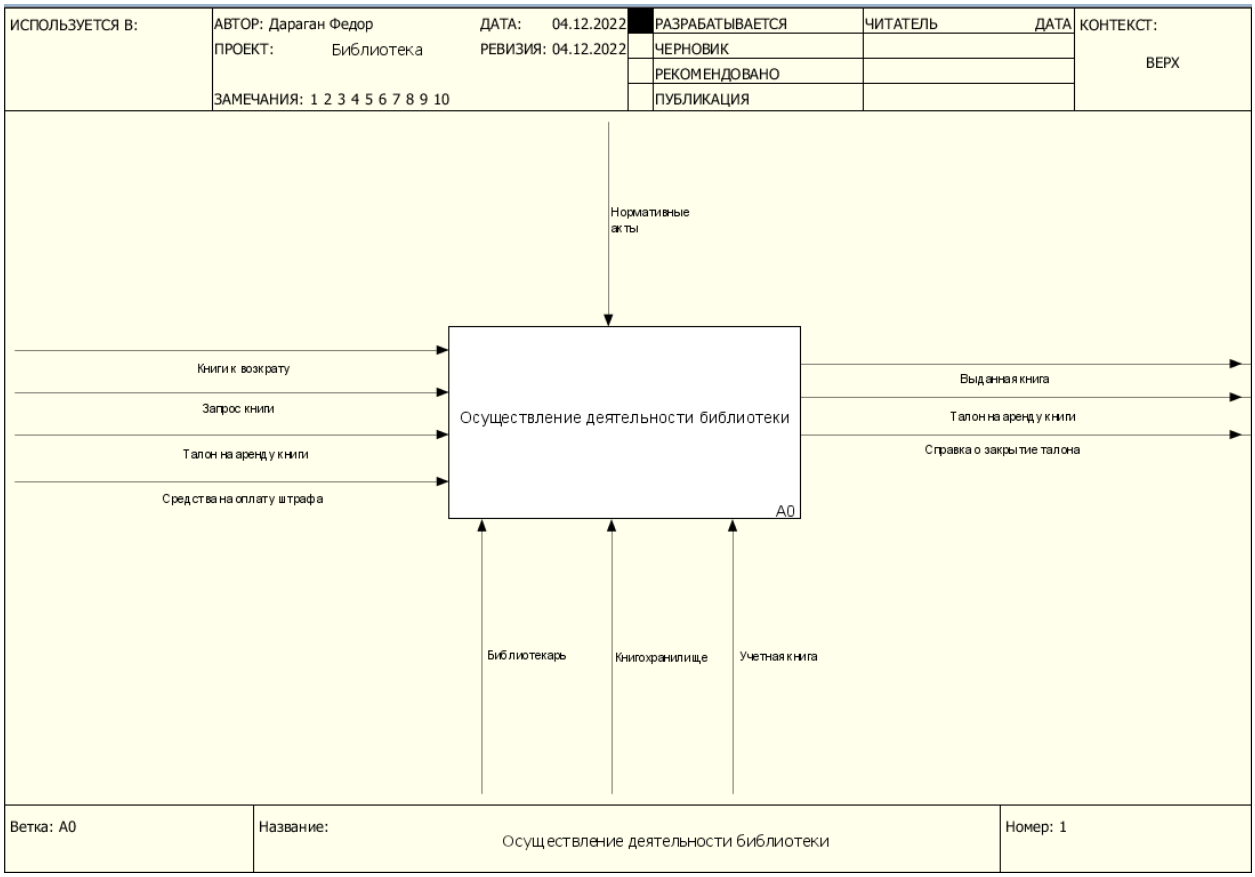

Рисунок 6. Уровень A0

На рисунке 7 показана декомпозиция уровня А0.

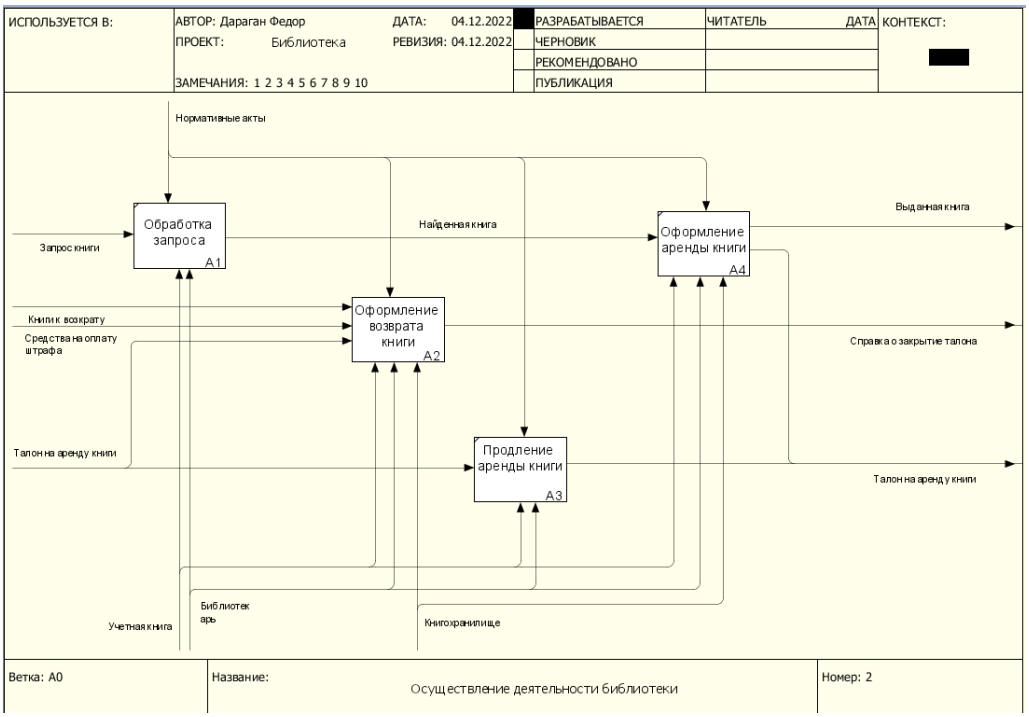

Рисунок 7. Декомпозиция уровня А0

На рисунке 8 показана декомпозиция уровня А2.

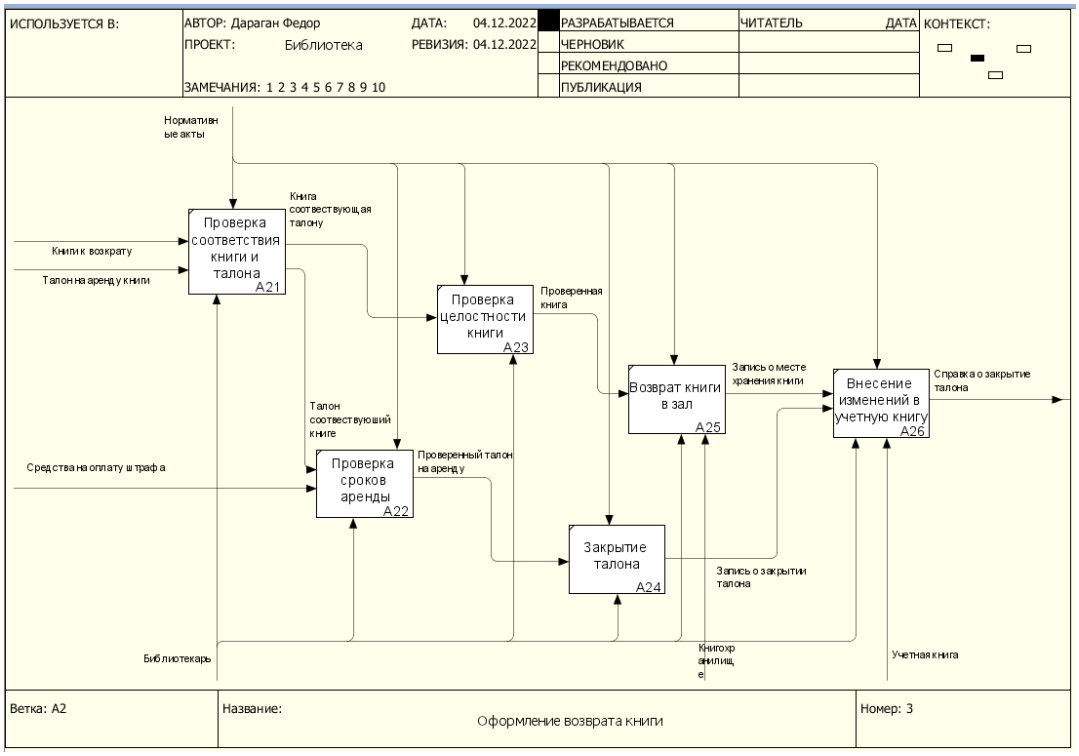

Рисунок 8. Декомпозиция уровня А2

Вывод

В результате работы была создана схема с 2-мя уровнями декомпозиции в нотации IDEF0. Она демонстрирует деятельность библиотеки. Рассмотрена деятельность по работе с книгами: выдача, продление аренды, возврат. Подробно рассмотрен процесс возврата книги. Схема реализует модель «AS-IS».

Практическая работа № 5: «Построение модели деятельности предприятия. Оптимизация бизнес-процессов»

Цель работы

1. Знакомство с понятием функциональной модели TO-BE («как будет»).
2. Доработка созданной модели AS-IS с учетом выявленных недостатков в организации бизнес-процессов.

Задание

Для заданной предметной области преобразовать созданную модель AS-IS в модель TO-BE. Внедрив информационную систему или клиент-серверную архитектуру. Модель должна содержать:

- не менее трёх уровней;
- на диаграмме 1-го уровня (A0) должно быть не менее 4-х функциональных блоков;
- на диаграмме 2-го и далее уровнях должно быть не менее 2-х функциональных блоков.

Ход работы

Функциональная модель TO-BE позволяет уже на стадии проектирования будущей ИС определить эти изменения. Применение функциональной модели TO-BE позволяет не только сократить сроки внедрения информационной системы, но также снизить риски, связанные с невосприимчивостью персонала к информационным технологиям. Модель TO-BE нужна для анализа альтернативных (лучших) путей выполнения функции и документирования того, как компания будет делать бизнес в будущем.

В ходе выполнения работы были автоматизированны процессы работы с данными. Учетная книга была заменена на ИС Система учета книг, что позволило производить первичную обработку запроса и продление аренды без участия библиотекаря. Также это позволило в процессе возврата книг автоматизировать проверку сроков аренды и шаблонное заполнение бумаг на закрытие талона и выписку справки.

На рисунке 9 показан уровень A0. Как можно увидеть, все бумажные талоны и справки были заменены на электронные.

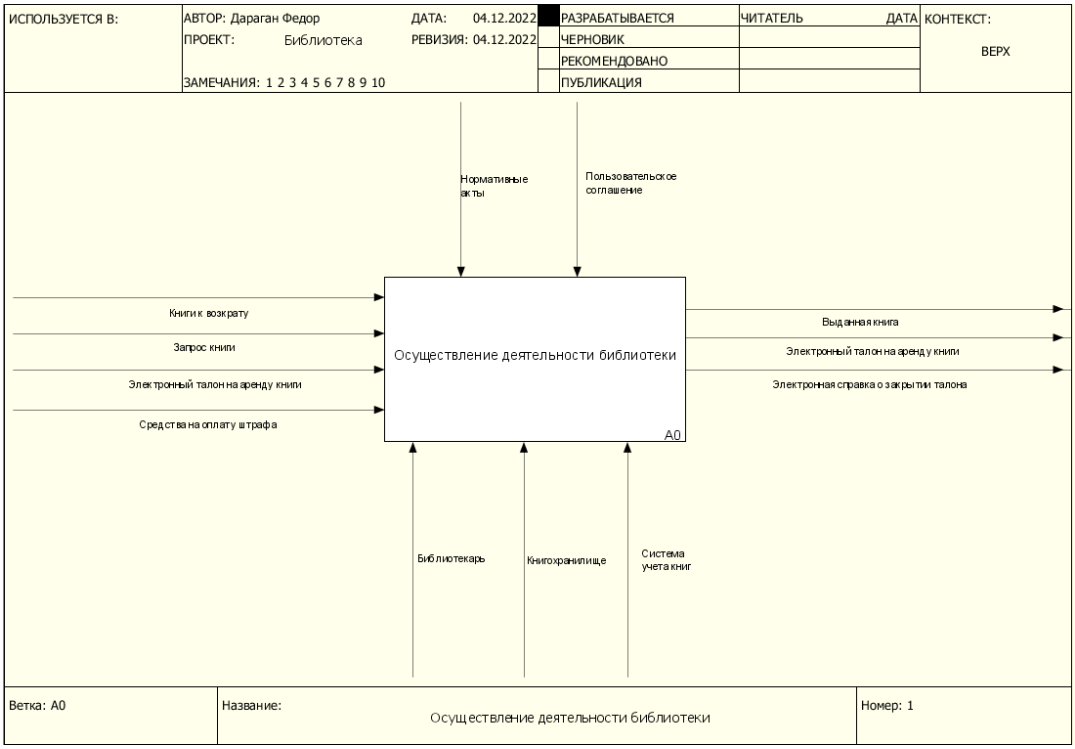

Рисунок 9. Уровень A0

На рисунке 10 показана декомпозиция уровня A0.

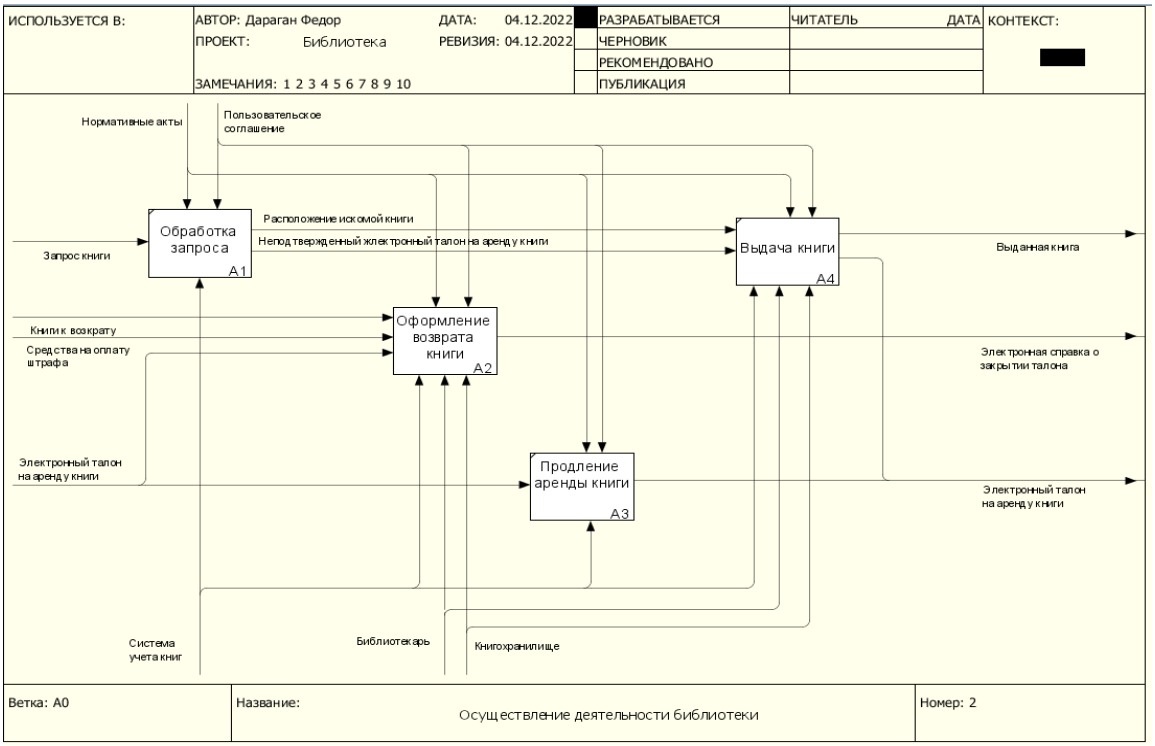

Рисунок 10. Декомпозиция уровня A0

На рисунке 11 показана декомпозиция уровня A1.

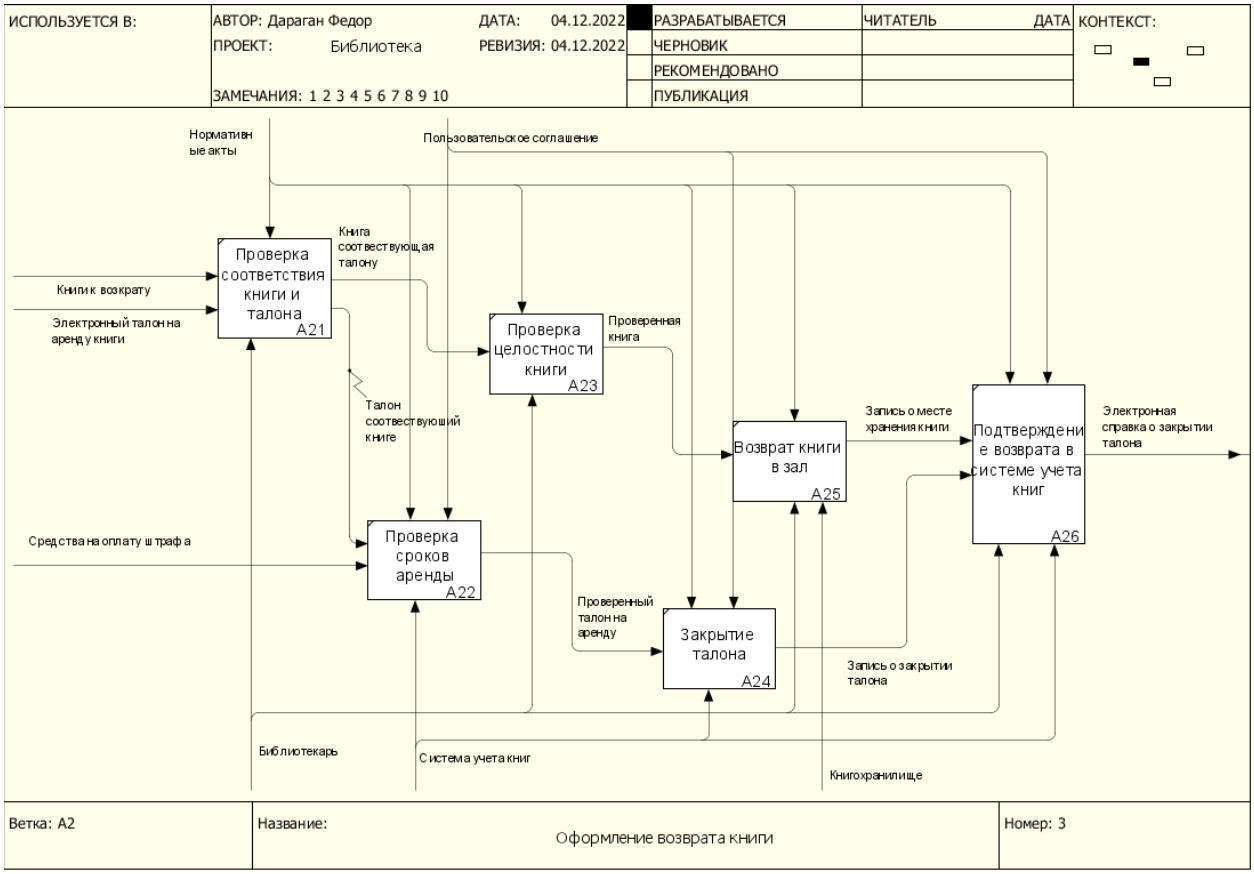

Рисунок 11. Декомпозиция уровня A1

Вывод

В результате работы была создана схема с 2-мя уровнями декомпозиции в нотации IDEF0. Она демонстрирует деятельность библиотеки. Рассмотрена деятельность по работе с книгами: выдача, продление аренды, возврат. Подробно рассмотрен процесс возврата книги. Схема реализует модель «TO-BE».

Практическая работа № 6: «Построение модели деятельности предприятия. Нотация IDEF3»

Цель работы

Получить практические навыки в построении IDEF3-модели бизнес-процесса.

Задание

С помощью методологии IDEF3 декомпозировать 1 из функциональных блоков модели окружения (A0), используя все типы перекрестков.

Модель должна содержать: Перекрестки типа AND, OR, XOR.

Ход работы

Была построена модель в формате IDEF3, декомпозирующая блок «Выдача книги» из практических работ 4 и 5. На рисунке 12 показана полученная модель.

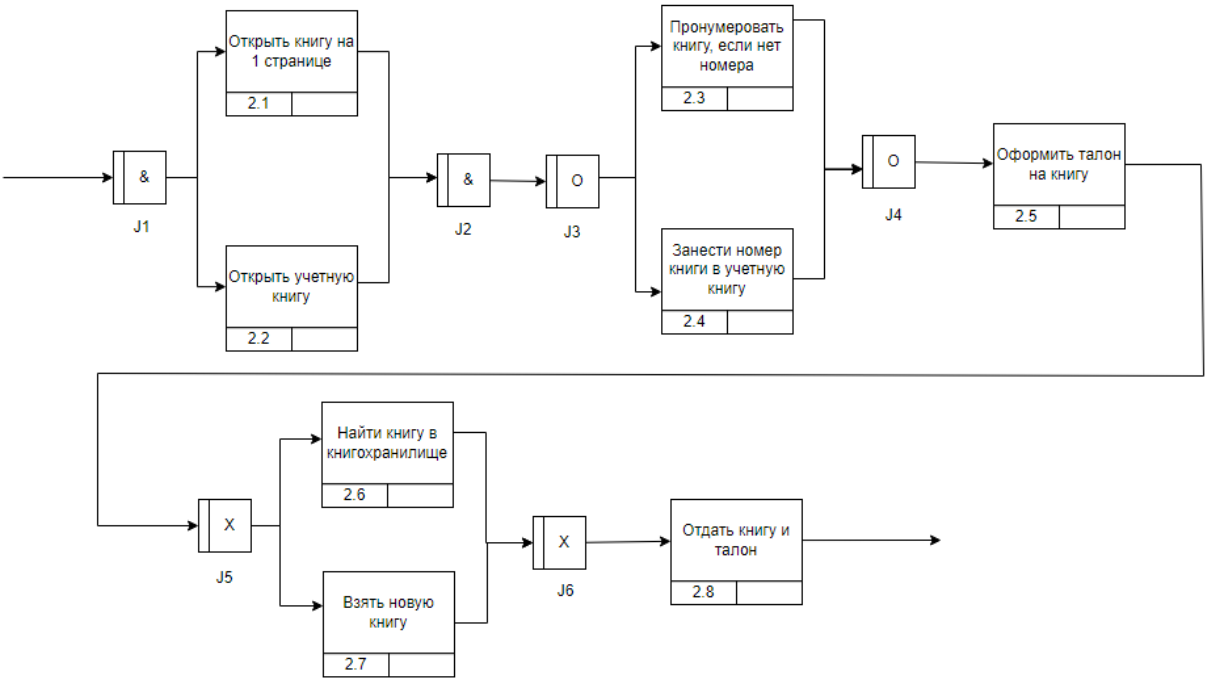

Рисунок 12. Декомпозиция блока «Выдача книги» в формате IDEF3

Вывод

В результате работы была создана декомпозиция блока «Выдача книги» в нотации IDEF3. Она показывает деятельность в процессе выдачи книги: нумерация книги, ее маркировка, оформление талона, поиск и выдача книги.

Практическая работа № 7: «Построение модели деятельности предприятия. Нотация DFD»

Цель работы

Получить практические навыки в построении DFD-модели бизнес-процесса.

Задание

С помощью методологии DFD декомпозировать 1 из функциональных блоков. Можно выбрать часть процесса, который моделировался на предыдущих лабораторных работах. При выборе учтите, что процесс обязательно должен предусматривать обработку информации, лучше, чтобы это была автоматизированная обработка с использованием одной или нескольких информационных систем.

Ход работы

Поскольку взаимодействие с информацией в рассмотренной системе просты было решено рассмотреть блоки «обработка запроса» и «выдача книги» вместе. Полученная модель показана на рисунке 13.

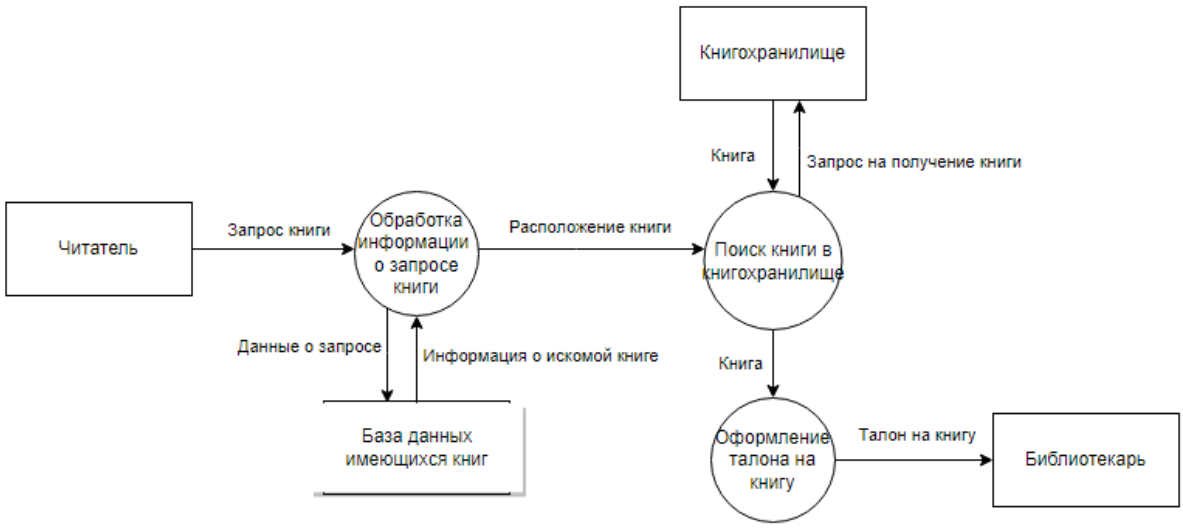

Рисунок 13. Модель обработки запроса и выдачи книги в нотации DFD

Вывод

В результате была создана модель, рассматривающая процесс обработки пользовательского запроса вплоть до оформления талона на книгу. В процессе происходит взаимодействие с базой имеющихся книг, а с внешней сущностью книгохранилища происходит обмен информацией.

Практическая работа № 8: «Построение модели деятельности предприятия. Нотация UML»

Цель работы

Получить практические навыки в построении прецедентной UMLмодели бизнес-процесса.

Задание

Выберите бизнес-процесс, для которого будете формировать модель. Вы можете выбрать один из вариантов процессов, описанных в приложении, или предложить свой вариант. Можно выбрать один из процессов, для которого на предыдущих лабораторных работах строилась модель по одной из структурных методологий. Желательно, чтобы процесс имел различные версии, т.е. альтернативные потоки событий.

Для заданной предметной области:

- построить диаграмму классов;
- построить диаграмму последовательности;
- построить диаграмму взаимодействий (диаграмму коммуникаций);
- построить диаграмму пакетов.

Ход работы

В качестве бизнес процесса была выбрана «деятельность ресторана».

На рисунке 14 показана диаграмма классов.

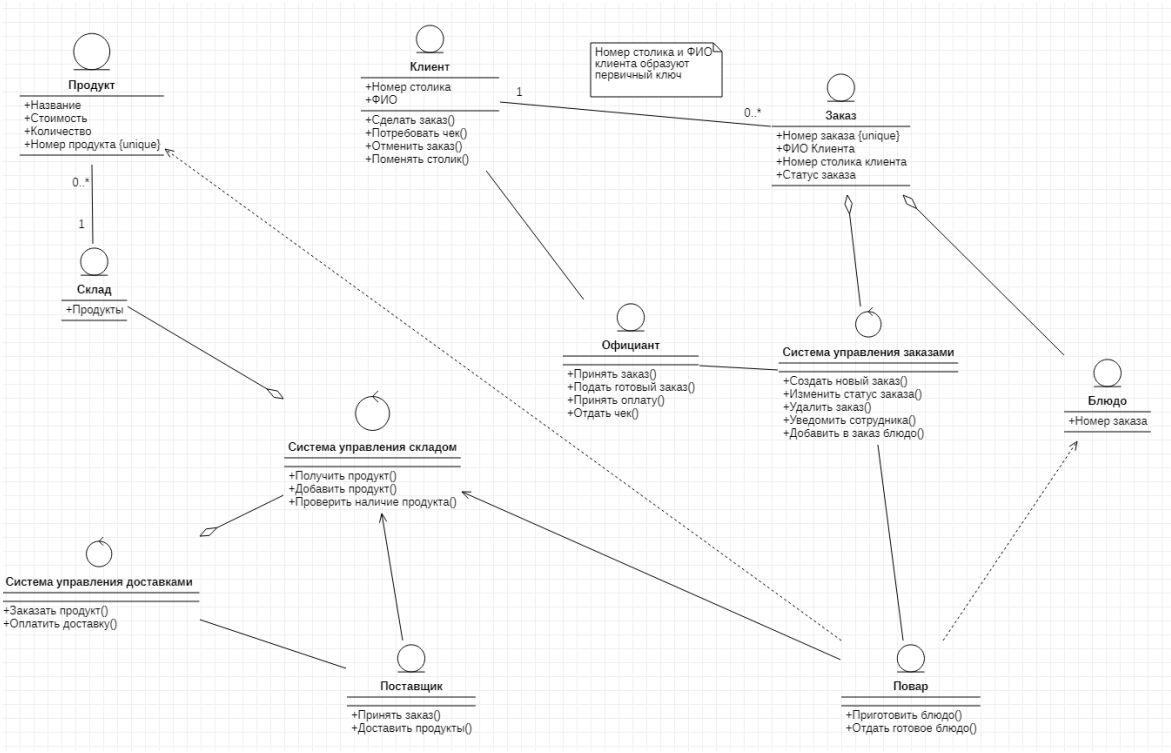

Рисунок 14. Диаграмма классов

На рисунке 15 показана диаграмма последовательности.

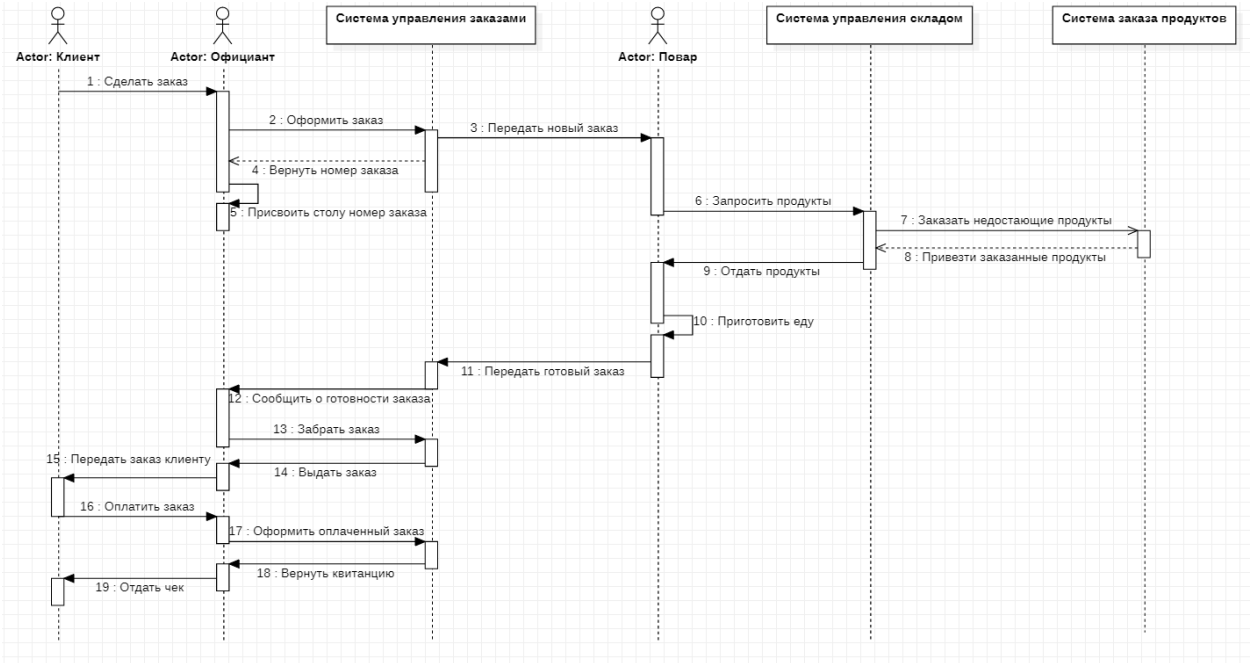

Рисунок 15. Диаграмма последовательности

На рисунке 16 показана диаграмма коммуникаций.

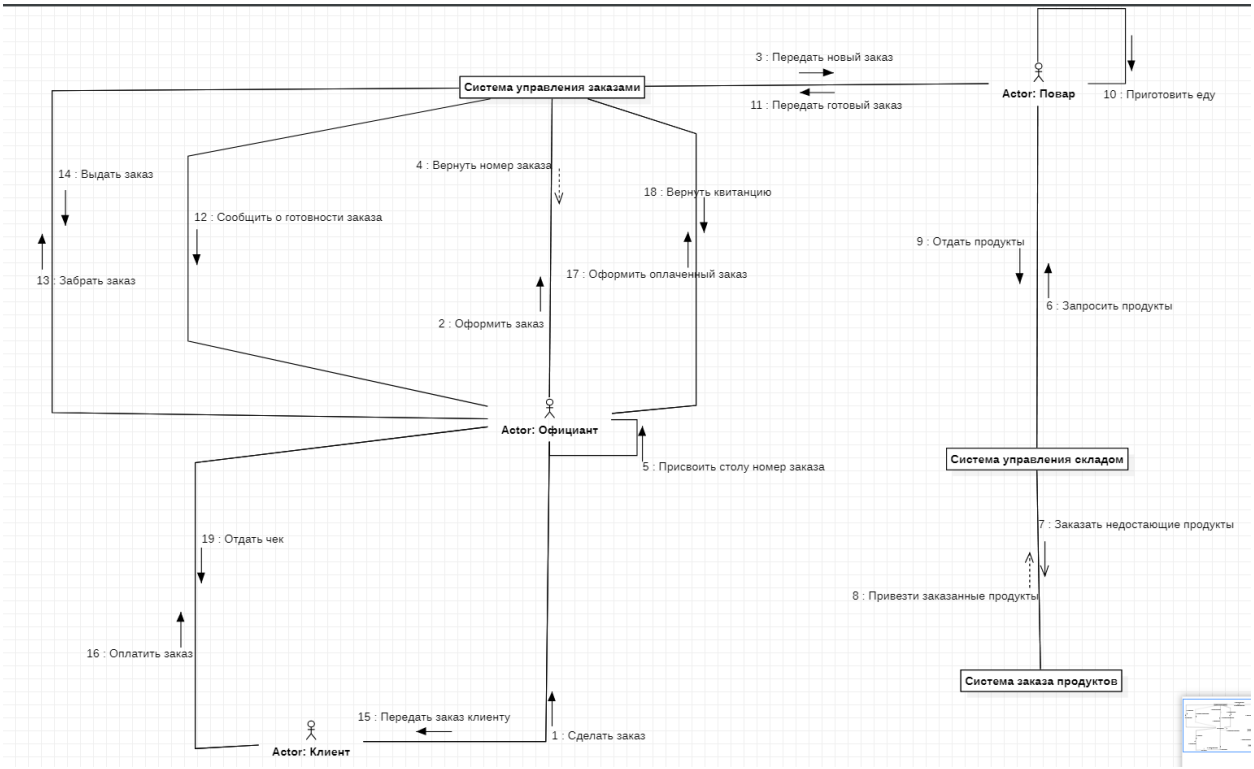

Рисунок 16. Диаграмма коммуникаций

На рисунке 17 показана диаграмма компонентов вместо диаграммы пакетов в виду отсутствия вложенности в проекте.

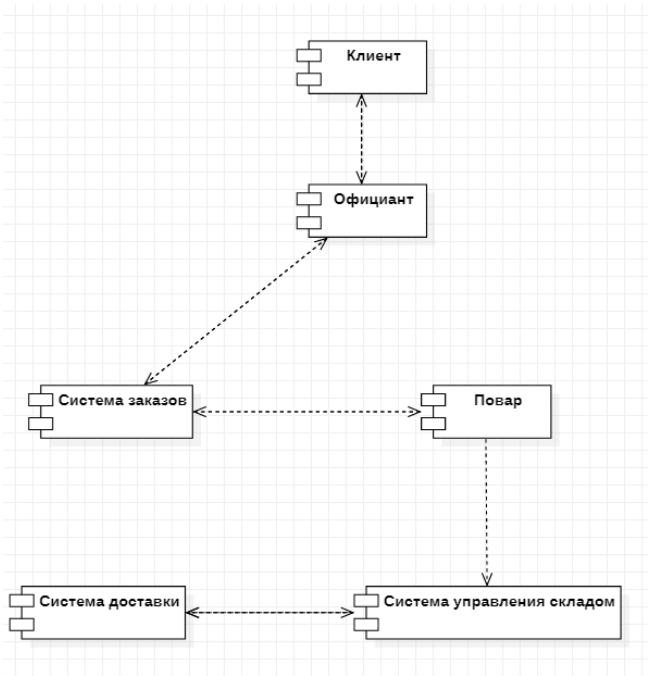

Рисунок 17. Диаграмма компонентов

Вывод

В результате работы была создана модель бизнес-процессов для ресторанного бизнеса. На основе данной модели в нотации UML были составлены диаграммы классов, последовательности, коммуникаций и компонентов. Созданные диаграммы представляют из себя визуализацию бизнес-системы и помогают в её документировании.