

**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет Радиотехнический
Кафедра “Системы обработки информации и управления”

Курс «Парадигмы и конструкции языков программирования»

Отчет по домашнему заданию

Выполнил:
студент группы РТ5-31Б:
Длютров Тимофей
Олегович
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий
Евгеньевич
Подпись и дата:

Москва, 2025

Постановка задачи

Разработать приложение windows forms на языке программирования C# с использованием принципа IOC/DI. Программа должна уметь зашифровывать и расшифровывать текстовый файл тремя разными способами и предоставлять отчёты о работе в форме журнала и в форме отдельных отчётов. Способы шифрования и предоставления отчётов могут выбираться пользователем в окне исполняемой программы.

Текст программы

Файл Program.cs:

```
1  using System;
2  using System.Collections.Generic;
3  using System.IO;
4  using System.Linq;
5  using System.Runtime.InteropServices.WindowsRuntime;
6  using System.Threading.Tasks;
7  using System.Windows.Forms;
8
9  namespace Decoder
10 {
11     //Объявление интерфейсов
12     public interface IDecoder
13     {
14         void BeginWork(string filename);
15         char DecodeNext();
16         void EndWork();
17     }
18     public interface IEncoder
19     {
20         void BeginWork(string filename);
21         void SendNext(char smb);
22         void EndWork();
23     }
24     public interface ILogger
25     {
26         void SendReport(string filename, string message);
27     }
}
```

```
28 //Создание классов
29     public class ClassicDecoder : IDecoder
30     {
31         FileStream f;
32         public void BeginWork(string filename)
33         {
34             f = File.OpenRead(filename);
35         }
36
37         public char DecodeNext()
38         {
39             if (f.Position <= f.Length - 1)
40             {
41                 char to_return = Convert.ToChar(f.ReadByte());
42                 return to_return;
43             }
44             else
45             {
46                 return '\0';
47             }
48         }
49
50         public void EndWork()
51         {
52             f.Close();
53         }
54     }
55
56     public class Crypt1Decoder : IDecoder
57     {
58         FileStream f;
59         public void BeginWork(string filename)
60         {
61             f = File.OpenRead(filename);
62         }
63
64         public char DecodeNext()
65         {
66             if (f.Position <= f.Length - 1)
67             {
68                 char to_return = Convert.ToChar(f.ReadByte() - 4);
69                 return to_return;
70             }
71             else
72             {
73                 return '\0';
74             }
75         }
76
77         public void EndWork()
78         {
79             f.Close();
80         }
81     }
82
83     public class Crypt2Decoder : IDecoder
84     {
85         FileStream f;
86         public void BeginWork(string filename)
87         {
88             f = File.OpenRead(filename);
89         }
90     }
```

```
91     ↓    | Ссылка: 3
92     ↓    | public char DecodeNext()
93     ↓    | {
94         |     if (f.Position <= f.Length - 1)
95         |     {
96             |         char to_return = Convert.ToChar(f.ReadByte() - ((f.Position - 1) % 4));
97             |         return to_return;
98         |     }
99         |     else
100         |     {
101             |         return '\0';
102         |     }
103     |
104     ↓    | Ссылка: 2
105     ↓    | public void EndWork()
106     ↓    | {
107         |     f.Close();
108     }
109
110    ↓    | Ссылка: 2
111    ↓    | public class ClassicEncoder : IEncoder
112    ↓    | {
113        |     FileStream f;
114        |     Ссылка: 2
115        |     public void BeginWork(string filename)
116        |     {
117            |         f = File.OpenWrite(filename);
118        |
119        |     public void SendNext(char smb)
120        |     {
121            |         f.WriteByte(Convert.ToByte(smb));
122        |
123        |     Ссылка: 2
124        |     public void EndWork()
125        |     {
126            |         f.Close();
127        }
128
129    ↓    | Ссылка: 1
130    ↓    | public class Crypt1Encoder : IEncoder
131    ↓    | {
132        |     FileStream f;
133        |     Ссылка: 2
134        |     public void BeginWork(string filename)
135        |     {
136            |         f = File.OpenWrite(filename);
137        |
138        |     Ссылка: 2
139        |     public void SendNext(char smb)
140        |     {
141            |         f.WriteByte(Convert.ToByte(Convert.ToByte(smb) + 4));
142        |
143        |     Ссылка: 2
144        |     public void EndWork()
145        |     {
146            |         f.Close();
147        }
148
149    ↓    | public class Crypt2Encoder : IEncoder
150    ↓    | {
151        |     FileStream f;
152        |     int current_pos = 0;
153        |     Ссылка: 2
154        |     public void BeginWork(string filename)
155        |     {
```

```
154     f = File.OpenWrite(filename);
155 }
156
157     Ссылок: 2
158     public void SendNext(char smb)
159     {
160         f.WriteByte(Convert.ToByte(Convert.ToByte(smb) + (current_pos % 4)));
161         current_pos++;
162     }
163
164     Ссылок: 2
165     public void EndWork()
166     {
167         f.Close();           ↗ void Crypt2Encoder.EndWork()
168     }
169
170     Ссылок: 2
171     public class LogLogger : ILogger
172     {
173         Ссылок: 5
174         public void SendReport(string path, string message)
175         {
176             string filename = path + "/MainLog.txt";
177             if (!File.Exists(filename))
178             {
179                 File.Create(filename).Close();
180                 File.AppendAllText(filename, "Some programm log");
181             }
182             File.AppendAllText(filename, "\n" + message + " --- " + DateTime.Now.ToString());
183         }
184
185         Ссылок: 1
186         public class ReportLogger : ILogger
187         {
188             Ссылок: 5
189             public void SendReport(string path, string message)
190             {
191                 string filename = path + "/" + "Report " + DateTime.Now.ToString().Replace(':', '.') + ".txt";
192                 File.Create(filename).Close();
193                 File.AppendAllText(filename, "Programm execution report\n");
194                 File.AppendAllText(filename, DateTime.Now.ToString() + "\n");
195                 File.AppendAllText(filename, "\nMessage:\n");
196                 File.AppendAllText(filename, message);
197             }
198             //Основная программа
199             Ссылок: 0
200             internal static class Program
201             {
202                 /// <summary>
203                 /// Главная точка входа для приложения.
204                 /// </summary>
205                 [STAThread]
206                 Ссылок: 0
207                 static void Main()
208                 {
209                     Application.EnableVisualStyles();
210                     Application.SetCompatibleTextRenderingDefault(false);
211                     Application.Run(new Form1());
212                 }
213             }
214         }
215     }
216 }
```

Файл Form1.cs:

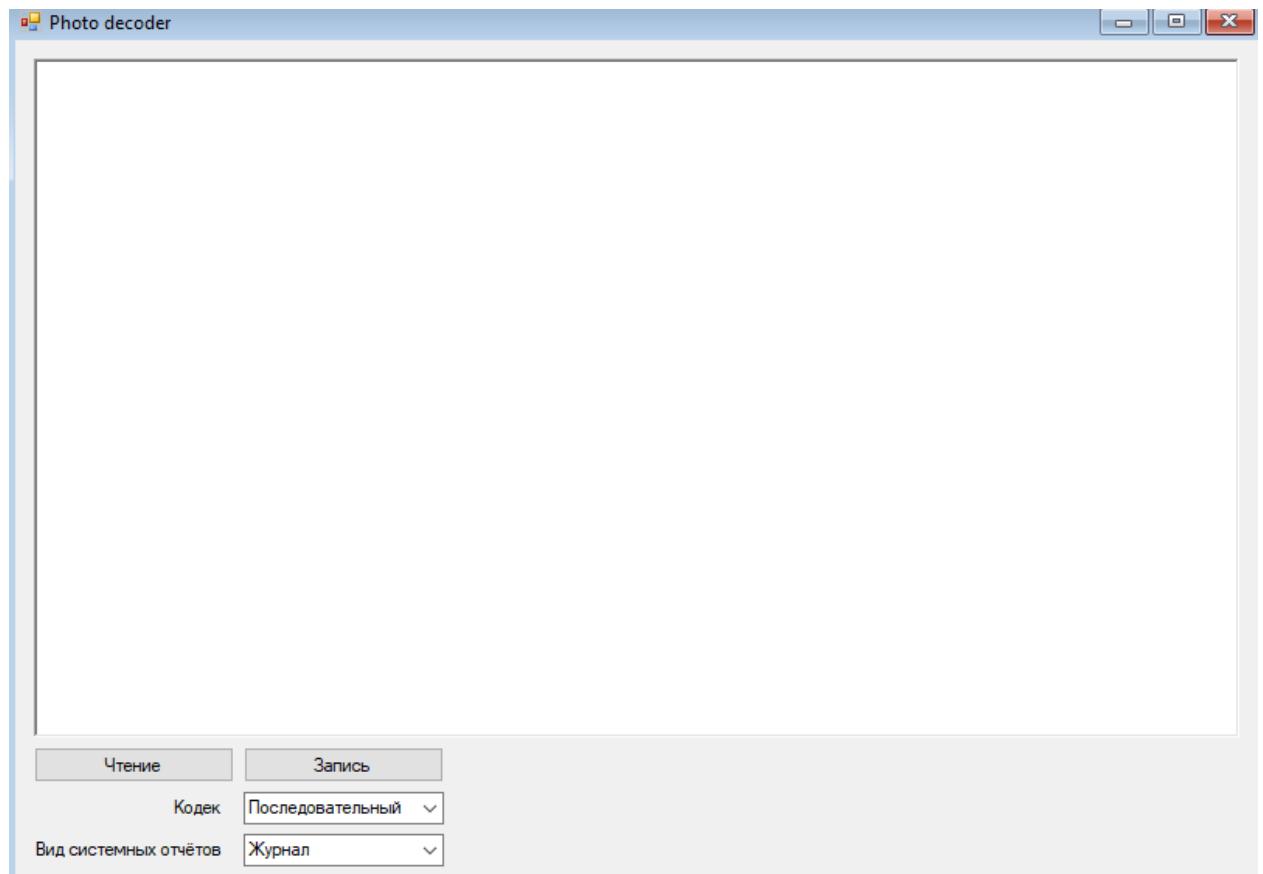
```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.IO;
7  using System.Linq;
8  using System.Text;
9  using System.Threading.Tasks;
10 using System.Windows.Forms;
11
12 namespace Decoder
13 {
14     Ссылок: 3
15     public partial class Form1 : Form
16     {
17         ILogger logger = new LogLogger();
18         IDecoder decoder = new ClassicDecoder();
19         IEncoder encoder = new ClassicEncoder();
20         Ссылок: 1
21         public Form1()
22         {
23             InitializeComponent();
24         }
25         Ссылок: 1
26         private void button1_Click(object sender, EventArgs e)
27         {
28             if (openFileDialog1.ShowDialog() == DialogResult.OK)
29             {
30                 decoder.BeginWork(openFileDialog1.FileName);
31                 char nexts = decoder.DecodeNext();
32                 richTextBox1.Clear();
33                 while (nexts != '\0')
34                 {
35                     richTextBox1.Text += nexts;
36                     nexts = decoder.DecodeNext();
37                 }
38                 richTextBox1.Update();
39                 decoder.EndWork();
40                 logger.SendReport(Application.StartupPath, "Чтение файла завершено\n");
41             }
42             else
43             {
44                 logger.SendReport(Application.StartupPath, "Файл не найден\n");
45             }
46         }
47         Ссылок: 1
48         private void button1_Click_1(object sender, EventArgs e)
49         {
50             if (saveFileDialog1.ShowDialog() == DialogResult.OK)
51             {
52                 encoder.BeginWork(saveFileDialog1.FileName);
53                 for (int i = 0; i < richTextBox1.Text.Length; ++i)
54                 {
55                     encoder.SendNext(richTextBox1.Text[i]);
56                 }
57                 encoder.EndWork();
58                 logger.SendReport(Application.StartupPath, "Запись файла завершена\n");
59             }
60             else
61             {
62                 logger.SendReport(Application.StartupPath, "Файл не найден\n");
63             }
64         }
65         Ссылок: 1
66         private void Form1_Load(object sender, EventArgs e)
67         {
68             //Добавление кодеков
69             codeccb.Items.Add("Последовательный");
70             codeccb.Items.Add("Шифр1");
71             codeccb.Items.Add("Шифр2");
72         }
73     }
74 }
```

```

70 //Добавление логгеров
71 loggercb.Items.Add("Журнал");
72 loggercb.Items.Add("Отчёт");
73 }
74
75 Ссылка: 1
76 private void codeccb_SelectedIndexChanged(object sender, EventArgs e)
77 {
78     switch (codeccb.SelectedIndex)
79     {
80         case 0: { decoder = new ClassicDecoder(); encoder = new ClassicEncoder(); break; }
81         case 1: { decoder = new Crypt1Decoder(); encoder = new Crypt1Encoder(); break; }
82         case 2: { decoder = new Crypt2Decoder(); encoder = new Crypt2Encoder(); break; }
83     }
84
85 Ссылка: 1
86 private void loggercb_SelectedIndexChanged(object sender, EventArgs e)
87 {
88     switch (loggercb.SelectedIndex)
89     {
90         case 0: { logger = new LogLogger(); break; }
91         case 1: { logger = new ReportLogger(); break; }
92     }
93 }
94 }
95

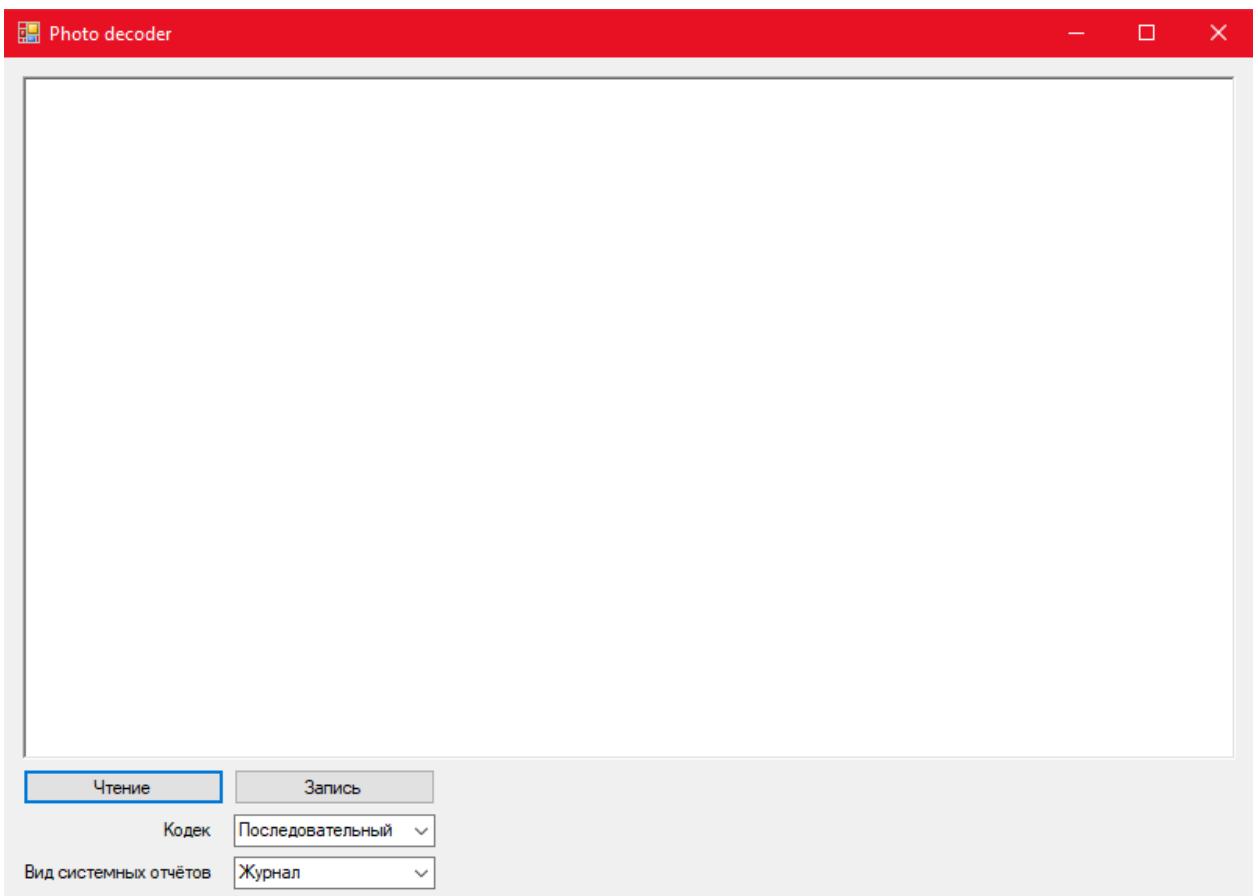
```

Форма Form1.cs (режим конструктора):



Проверка работы программы

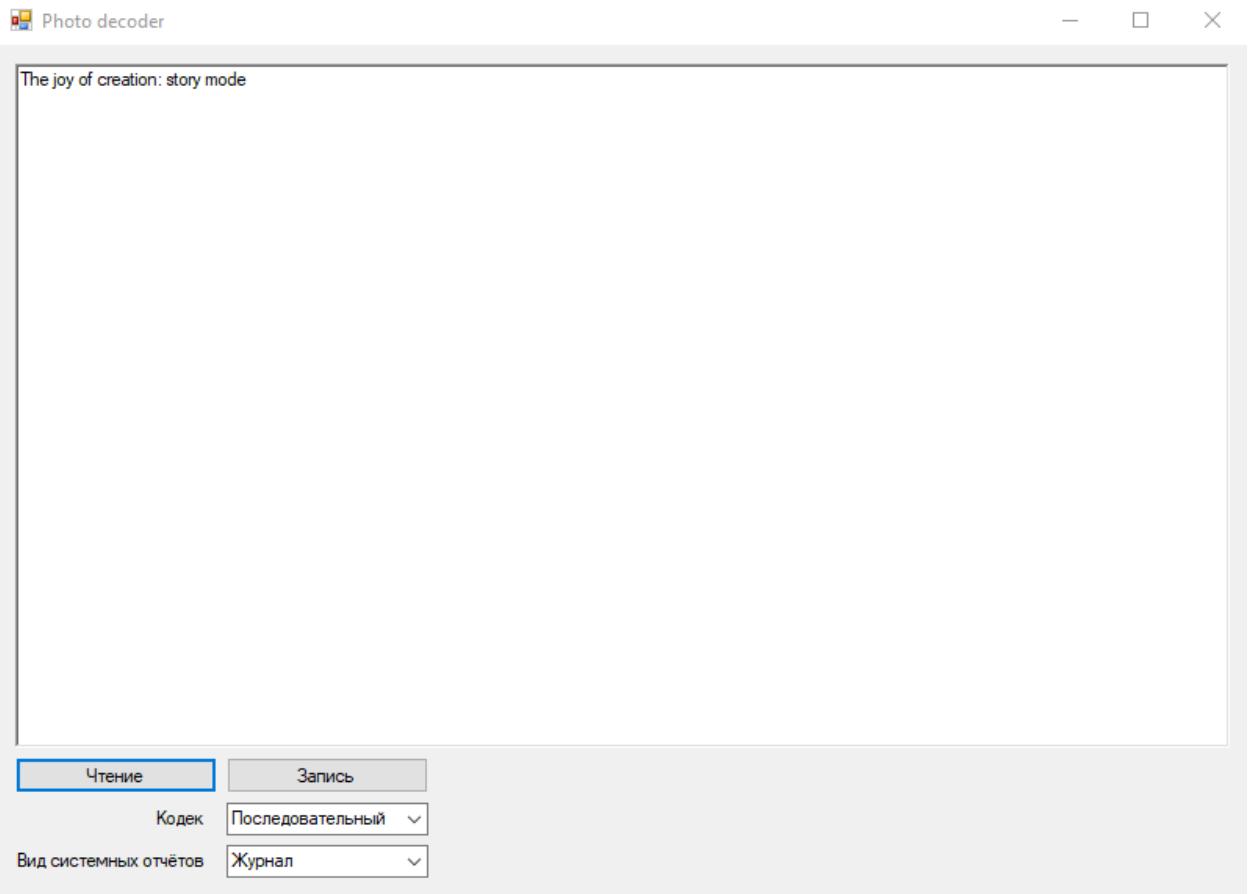
После запуска открывается форма:



Файл, зашифрованный последовательным кодеком:



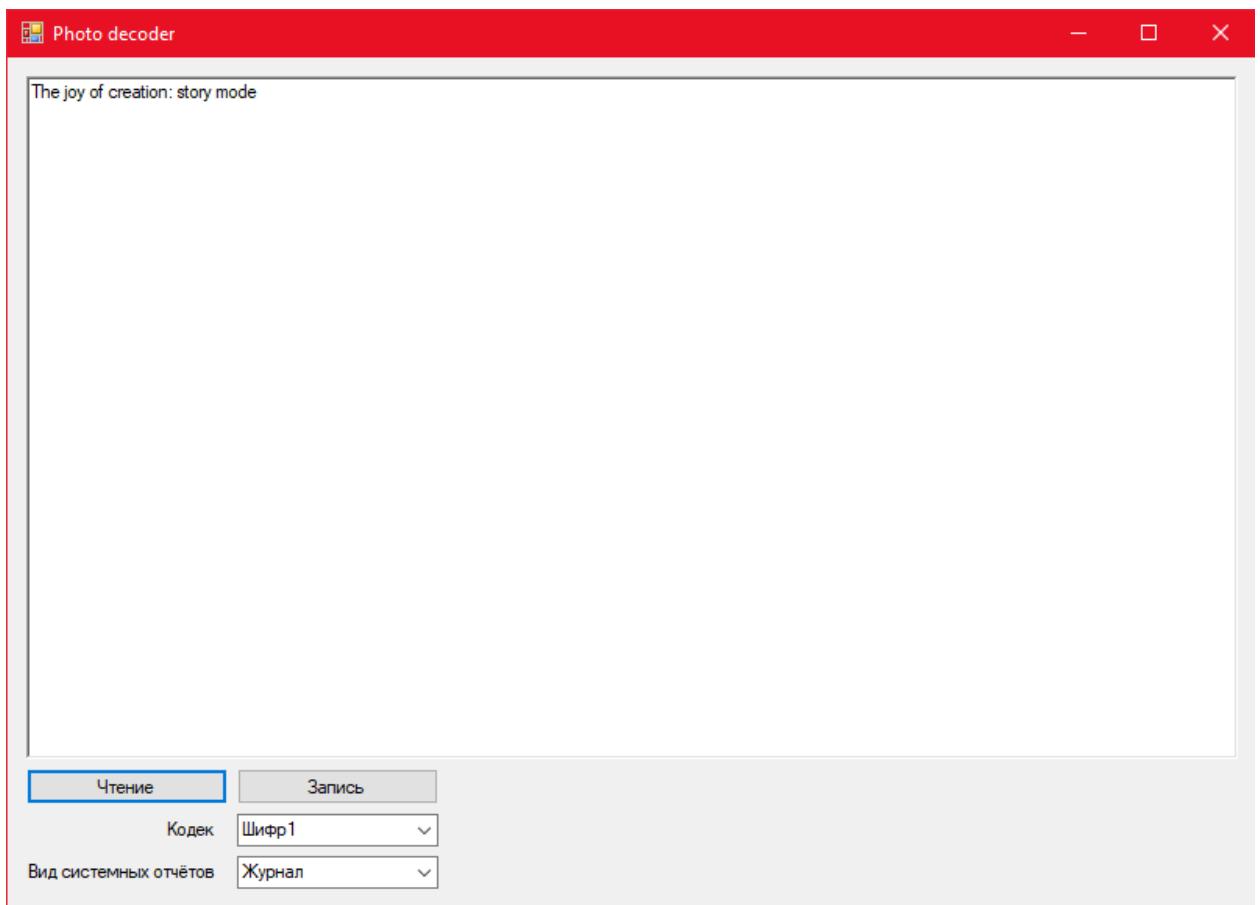
Файл, расшифрованный последовательным кодеком:



Файл, зашифрованный первым кодеком:



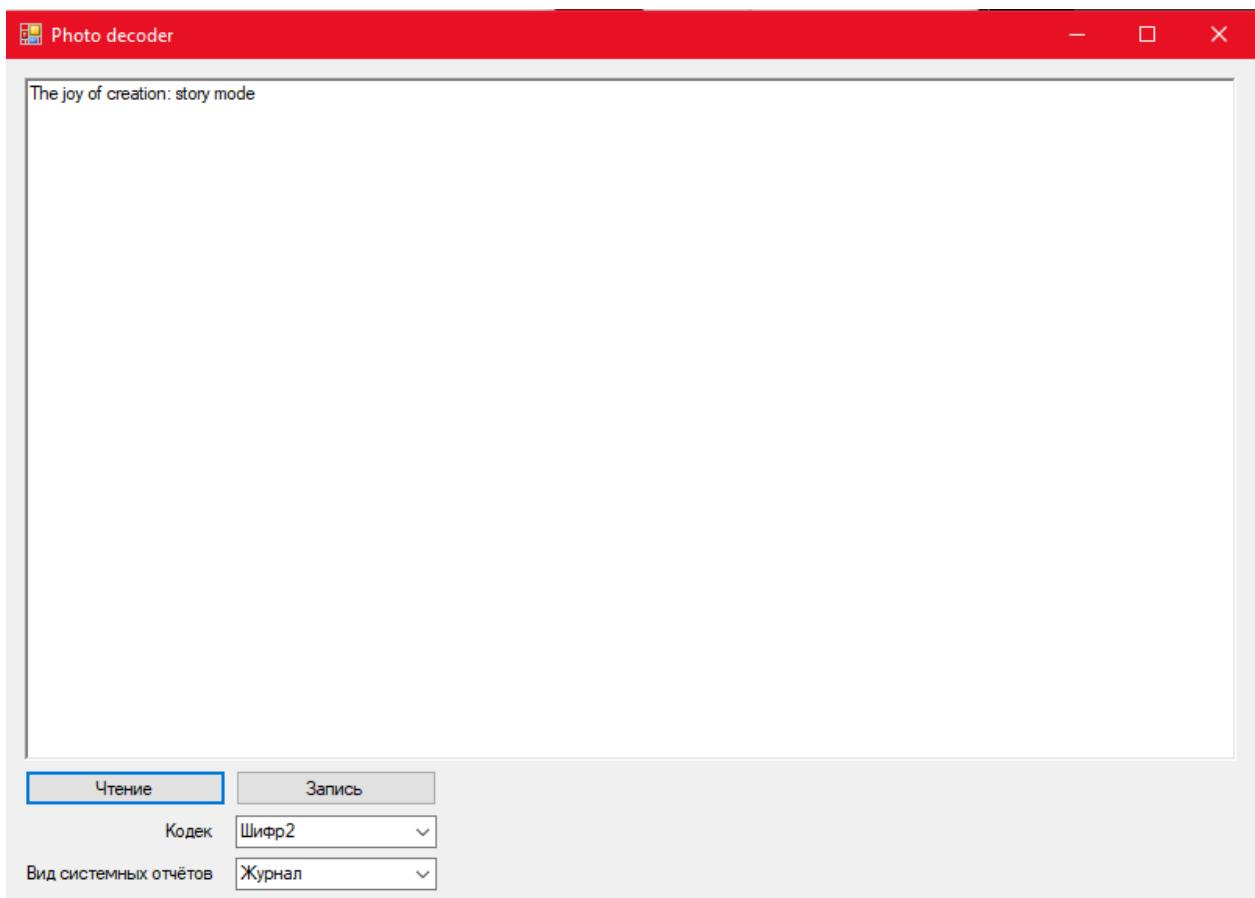
Файл, расшифрованный первым кодеком:



Файл, зашифрованный вторым кодеком:



Файл, расшифрованный вторым кодеком:



Журнал после всех действий:

```
1 Some programm log
2 Чтение файла завершено
3 --- 31.12.2025 17:37:37
4 Чтение файла завершено
5 --- 31.12.2025 17:40:30
6 Чтение файла завершено
7 --- 31.12.2025 17:41:27
```

Отчёт о проделанном действии:

```
1 Programm execution report
2 19.12.2025 15:33:35
3
4 Message:
5 Чтение файла завершено
6
```