

ДЕПАРТАМЕНТ ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
ТОМСКОЙ ОБЛАСТИ
ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
«ТОМСКИЙ ТЕХНИКУМ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

Специальность 09.02.07 «Информационные системы и программирование»

Разработка и тестирование приложения «Шифратор сообщений»

Пояснительная записка
к курсовому проекту
КП.22.09.02.07.603.12.ПЗ

Студент	_____	П. Д. Левицкий
«__»_____ 2022 г.		
Руководитель	_____	Д.А. Антипов
«__»_____ 2022 г.		

Томск 2022

Содержание

Введение	3
1 ОБЩАЯ ЧАСТЬ	5
1.1 Анализ предметной области	5
1.2 Выбор средств тестирования	8
2 СПЕЦИАЛЬНАЯ ЧАСТЬ	12
2.1 Описание требований к информационной системе	12
2.2 Диаграмма вариантов использования	16
2.3 Диаграмма состояний	17
2.4 Словари данных	18
2.5 Тестовые сценарии	29
ЗАКЛЮЧЕНИЕ	36
Перечень используемых источников	37
Приложение А. Результаты тестирования	38
Приложение Б. Листинг кода приложения	52

					ОП.22.090207.603.12.ПЗ							
Изм	Лист	№ докум.	Подпись	Дата								
Разраб.		Левицкий П. Д.			Пояснительная записка				Лит.		Лист	Листов
Пров.									Т		2	36
									ТТИТ 603 гр.			
Н. контр.												
Уте.												

Введение

Проблема защиты конфиденциальной информации всегда стояла ребром. Пользователи предпочитают отправлять логины, пароли и секретные фразы, используя социальные сети только потому, что это просто и быстро.

Но это крайне небезопасно - злоумышленник может получить доступ к аккаунтам, что может привести к массе нежелательных последствий - угон аккаунтов, рассылка спам-сообщений, кража средств, шантаж и вымогательство, список можно продолжать очень долго.

Основное правило безопасной передачи информации - канал передачи пароля должен быть иным относительно основного. Например, если адрес электронной почты передается через социальную сеть, то пароль от почтового аккаунта стоит передать через что угодно, но не через ту же социальную сеть - вероятность отслеживания третьими лицами сразу двух каналов передачи информации крайне низка.

Но разделение пары логин-пароль на два разных канала передачи не поможет в том случае, если устройство отправителя и\или получателя скомпрометировано.

Возможный выход - передача двух зашифрованных архивов. Но этот способ имеет свои недостатки - например, архив может быть поврежден или архиватор не будет поддерживать архивы конкретного типа.

Самый простой вариант - отправка зашифрованной информации напрямую через мессенджер\социальную сеть. Главное условие - знание способа шифровки для последующей дешифровки полученного сообщения и наличие программы, способной дешифровать сообщение, используя соответствующий алгоритм.

Целью данного курсового проекта является проведение тестирования функций приложения шифровки сообщений. С помощью приложения есть возможность зашифровывать и расшифровывать сообщения четырьмя способами.

Из соображений безопасности в приложении не предусмотрено наличие базы данных - все необходимые наборы данных хранятся непосредственно в массивах, словарях и списках (в оперативной памяти).

В процессе тестирования у тестировщика будет полный доступ к коду приложения. Предполагается тестирование с применением методики "белого ящика", так как для полноценного тестирования необходимо понимание логики шифровки и дешифровки.

В пояснительной записке к курсовому проекту представлены требования к тестируемому приложению, UML-диаграммы, словари данных, тестовые случаи, наборы данных для тестирования и непосредственно результаты проведения тестов.

Задачи:

1. Проведение анализа конкурентов;
2. Ознакомление с требованиями к продукту;
3. Просмотр кода тестируемых функций;
4. Описание тестовых случаев;
5. Определение входных данных для тестируемых функций;
6. Проведение тестирования реализованных функций.

					КП.22.09.02.07.603.12.ПЗ	Лист
						4
Изм		№ докум.	Подпись	Дата		

1. ОБЩАЯ ЧАСТЬ

1.1 Анализ предметной области

1.1.1 Анализ конкурентов

Шифратор текста – это приложение для обеспечения безопасности. Такие приложения довольно популярны, так как не каждый пользователь станет заниматься распаковкой зашифрованного архива или на его устройстве просто не хватает ресурсов для установки соответствующего приложения.

«Text Encrypt» (1.1.1.0)

Положительные стороны:

- Малый размер установленного приложения;
- Реализован процесс шифровки и дешифровки с помощью алгоритма AES (заявлено);
- Встроенный генератор приватного ключа.

Отрицательные стороны:

- Реализован лишь один способ шифровки и дешифровки;
- Отсутствие русской локализации;
- Нелогичное размещение элементов. Пользователю придется разбираться перед использованием;
- Реклама "продвинутого" генератора ключей во вкладке генератора приватного ключа;
- Нет кнопки очистки полей ввода;
- Возможность "расшифровать" расшифрованное сообщение, что приводит к логической ошибке.

Приложение «Text Encrypt for WhatsApp» (1.125.11.0)

Положительные стороны:

					КП.22.09.02.07.603.12.ПЗ	Лист
Изм.		№ докум.	Подпись	Дата		5

- Малый размер установленного приложения;
- Простой пользовательский интерфейс;
- Имеется возможность поделиться зашифрованным сообщением;
- Есть кнопка очистки полей ввода.

Отрицательные стороны:

- Реализован лишь один неназванный способ шифровки и дешифровки;
- Отсутствие русской локализации;
- Реклама платной версии на главной странице приложения;
- Возможность "расшифровать" расшифрованное сообщение;
- Название приложение может ввести потенциального пользователя в заблуждение.

Приложение «Шифрование текста» (1.6)

Положительные стороны:

- Малый размер установленного приложения;
- Простой пользовательский интерфейс;
- Поддержка «одноключевого» и «двухключевого» шифрования;
- Есть кнопка очистки полей ввода.
- Логичный и понятный интерфейс, начать работу можно сразу после установки;
- Наличие русской локализации;
- Наличие подсказок для ввода;
- Наличие кнопок «копировать» и «вставить».

Отрицательные стороны:

- Реализовано лишь два неназванных способа шифровки и дешифровки.

Приложение «Зашифровать Расшифровать (Encrypter)» (1.0.27) – [линк]

Положительные стороны:

					КП.22.09.02.07.603.12.ПЗ	Лист
Изм.		№ докум.	Подпись	Дата		6

- Малый размер установленного приложения;
- Большое количество способов шифровки, в общей сумме — 10 штук;
- Наличие подсказок для ввода;
- Наличие кнопки «поделиться»;
- Наличие кнопок «копировать» и «вставить»;

Отрицательные стороны:

- Нет кнопки очистки полей ввода;
- Отсутствие русской локализации;
- Сложный и запутанный интерфейс;
- Название приложения не описывает его сути
- Нет встроенной справки, при выборе способа шифрования указаны лишь их названия.

					КП.22.09.02.07.603.12.ПЗ	Лист
Изм		№ докум.	Подпись	Дата		7

1.2 Выбор средств тестирования

Тестирование - один из важнейших этапов жизненного цикла любого продукта. Оно позволяет выявить ошибки путем проверки соответствия реального поведения программы и ее ожидаемым поведением путем создания набора тестовых случаев.

Данный курсовой проект требует проведения функционального тестирования с целью доказательства соответствия итогового результата с исходными требованиями.

Функциональное тестирование имеет следующие преимущества:

- Полностью имитирует использование ПО за счет тестирования всех функций;
- Позволяет сократить время разработки за счет выявления ошибок на ранних стадиях разработки;
- Повышает общее качество продукта.

Тестирование будет проводиться методом "белого ящика", поскольку доступен исходный код программы. Этот метод будет использоваться на уровнях модульного и регрессионного тестирования.

Выбранная техника тестирования обладает такими преимуществами, как:

- Тестирование может проводиться на самых ранних этапах разработки;
- Повышение тщательности тестирования и попутное исправление ошибок при условии их обнаружения;
- Повышается максимально возможный процент покрытия кода тестами.
- Модульные тесты можно начинать писать сразу после реализации первого модуля, подлежащего тестированию при условии, если функционал в ходе разработки не изменяется.

					КП.22.09.02.07.603.12.ПЗ	Лист
						8
Изм		№ докум.	Подпись	Дата		

Также этот метод имеет недостатки:

- Требуются специальные навыки написания модульных тестов;
- Требуется понимание структуры исходного кода программы;
- Желательно располагать документацией (техническое задание, диаграммы и т. д.).

Было принято решение использовать именно эту технику тестирования, так как ее недостатки данном случае несущественны.

В первую очередь необходимо составить тестовые сценарии. Так как техника тестирования - "белый ящик", то тестовые сценарии будут основываться преимущественно на коде функций приложения.

Следующий этап - определение входных данных для тестов.

Модульное тестирование имеет следующие преимущества:

- Тестирование отдельно взятых методов программы

Для создания теста берется конкретный метод, определяются входные и ожидаемые параметры. В случае соответствия ожидаемого и фактического результата вызова метода тест считается пройденным.

- Тестирование происходит изолированно от остального кода

Отдельное тестирование методов позволяет довольно легко и быстро определить, логика какого метода некорректна или имеет ошибки и требует исправления.

- Автоматизация выполнения тестов

Покрытие максимального процента кода ведет к резкому увеличению количества тестов. Автоматизация позволяет не отвлекаться на изучение результатов - достаточно увидеть, был ли пройден тот или иной тест.

					КП.22.09.02.07.603.12.ПЗ	Лист
						9
Изм.		№ докум.	Подпись	Дата		

- Тестирование только публичных конечных точек

Малейшие изменения в классе могут привести к провалу тестов, так как реализация используемого класса изменилась. По этой причине при написании модульных тестов ограничиваются только общедоступными конечными точками, что позволяет изолировать модульные тесты от многих деталей внутренней реализации компонента. Как итог - уменьшение вероятности того, что изменения в классах могут привести к провалу юнит-тестов.

- Простая поддержка

При изменении первоначальных требований к продукту, как правило, изменяются и тестируемые методы. Для корректного прохождения тестирования нужно актуализировать эти тесты. Это можно сделать максимально быстро.

- Хорошо составленные тесты могут служить сопровождающей документацией разработчикам, поддерживающим продукт, но не какие функциональные возможности предоставляет модуль и как его использовать, могут взглянуть на модульные тесты, чтобы получить общее представление об API модуля.

Далее представлены некоторые из классов Assert, верифицирующие результаты теста:

- `AreEqual(object expected, object actual)`: проверяет равенство двух объектов;
- `AreEqual<T>(T expected, T actual)`: проверяет равенство двух коллекций;
- `AreNotEqual(object expected, object actual)`: проверяет неравенство двух объектов;
- `AreNotEqual<T>(T expected, T actual)`: проверяет неравенство двух коллекций;
- `AreSame(object expected, object actual)`: проверяет, указывают ли оба объекта на один и тот же объект в памяти;
- `AreNotSame(object expected, object actual)`: проверяет, указывают ли оба объекта на разные объекты в памяти;

					КП.22.09.02.07.603.12.ПЗ	Лист
						10
Изм.		№ докум.	Подпись	Дата		

- Equals(object objA, object objB): проверяет на равенство оба объекта;
- .IsFalse(bool condition): проверяет, равно ли условие condition значению false;
- .IsTrue(bool condition): проверяет, равно ли условие condition значению true;
- .IsNull(object value): проверяет, имеет ли объект value значение null;
- .InstanceOfType(object value, Type expectedType): проверяет представление объектом value типа expectedType.

Выше перечислены далеко не все утверждения для проведения тестирования. Но уже из этого можно сделать вывод, что использование фреймворка MSTest будет необходимым и более чем достаточным.

					КП.22.09.02.07.603.12.ПЗ	Лист
						11
Изм		№ докум.	Подпись	Дата		

2. СПЕЦИАЛЬНАЯ ЧАСТЬ

2.1 Описание требований к информационной системе

2.1.1 Общее описание

Имеется приложение-шифратор, позволяющее шифровать и расшифровывать введенные пользователем сообщения. Реализована корректная обработка данных, также реализована проверка корректности входных данных.

Функционал пользователя:

- шифровка сообщений первым способом;
- шифровка сообщений вторым способом;
- шифровка сообщений третьим способом;
- шифровка сообщений четвертым способом.
- дешифровка сообщений, зашифрованных первым способом;
- дешифровка сообщений, зашифрованных вторым способом;
- дешифровка сообщений, зашифрованных третьим способом;
- дешифровка сообщений, зашифрованных четвертым способом;
- включение\отключение очистки полей ввода при шифровке;
- включение\отключение очистки полей ввода при шифровке;

В приложении также доступен просмотр краткой справки. Это необходимо для получения пользователем общего понимания процесса шифрования.

2.1.2 Требования к интерфейсу приложения

Продукт имеет графический user-friendly интерфейс, в дизайне применена стандартная палитра, так как она хорошо сочетается со стандартной темой целевой операционной системы.

Приложение не обладает множеством вкладок, настроек и опций, что делает его предельно понятным даже для пользователей, слабо знакомых с компьютером - во все разделы можно попасть по одному нажатию кнопки мыши.

					КП.22.09.02.07.603.12.ПЗ	Лист
						12
Изм		№ докум.	Подпись	Дата		

Из соображений безопасности в раздел настроек было добавлено две опции – очистка полей ввода во вкладках «Шифровка» и «Дешифровка». Во вкладке «Шифровка» по умолчанию введенный текст будет очищаться, в то время как для раздела «Дешифровка» флажок снят.

2.1.3 Требования к структуре приложения

Программа содержит следующие вкладки:

- 1) Шифровка
- 2) Дешифровка
- 3) Настройки
- 4) О программе

Разделы «Шифровка» и «Дешифровка» содержат в себе по четыре графически выделенных блока – по одному на каждый из четырех способов шифровки.

В разделе «Настройки» содержатся два флажка, определяющие, будут ли стерты вводимые данные в разделах шифровки и дешифровки.

Раздел «О программе» содержит в себе краткую справку, описывающую процессы шифровки.

2.1.4 Требования к хранению данных

Приложение предполагает ввод, хранение, обработку и вывод данных.

Входные данные вводятся в соответствующие поля на вкладке «Шифровка», по нажатию кнопки данные записываются в переменные.

Также входные данные генерируются и хранятся в массивах и\или листах внутри приложения, базы данных не используются из соображений безопасности.

Далее данные обрабатываются и выводятся в окно вывода на форме шифровки.

Все вышеперечисленное также справедливо и для раздела "Дешифровка".

2.1.5 Требования к языкам программирования

Интерфейс был разработан с помощью WPF (Windows Presentation Foundation). WPF - графическая подсистема, использующая расширяемый язык разметки XAML. Платформа WPF входит в состав .Net Framework и поддерживает необходимый для построения и компоновки набор элементов и инструментов.

Для написания бизнес-логики был применен язык C#, входящий в состав .Net Framework 6.0.

Главное преимущество использования C# при написании бизнес-логики проекта заключается в его статической типизации. По сравнению с динамической типизацией это позволяет избежать множества ошибок при написании логики приложения и, как следствие, избежать проблем при отладке.

2.1.6 Требования к программному обеспечению

Для корректного запуска приложения устройство, на котором развертывается приложение должно отвечать следующим требованиям:

- Операционная система: MS Windows 10;
- Установленный пакет .Net 6.0 Runtime.

2.1.7 Требования к аппаратному обеспечению

Для корректной работы приложения необходимо аппаратное обеспечение со следующими минимальными характеристиками:

- Процессор: не менее 1 ГГц или SoC;
- ОЗУ: 1 ГБ для 32-разрядной системы или 2 ГБ для 64-разрядной системы
- Место на жестком диске: 16 ГБ для 32-разрядной ОС или 20 ГБ для 64-разрядной ОС;

					КП.22.09.02.07.603.12.ПЗ	Лист
						14
Изм		№ докум.	Подпись	Дата		

- Видеоадаптер: DirectX 9 или более поздняя версия с драйвером WDDM 1.0;
- Экран: 800 x 600.

2.1.8 Требования к лингвистическому обеспечению

Взаимодействие пользователя с продуктом должно осуществляться полностью на русском языке (исключение могут составлять только системные сообщения, выдаваемые программными продуктами третьих компаний), графический интерфейс пользователя продукта должен быть создан на русском языке.

2.1.9 Требования к адаптивности приложения

Приложение должно обладать масштабируемым интерфейсом и быть адаптированным под большинство стандартных разрешений экранов.

					КП.22.09.02.07.603.12.ПЗ	Лист
						15
Изм		№ докум.	Подпись	Дата		

2.2 Диаграмма вариантов использования

Диаграмма вариантов использования отражает доступный пользователю функционал.

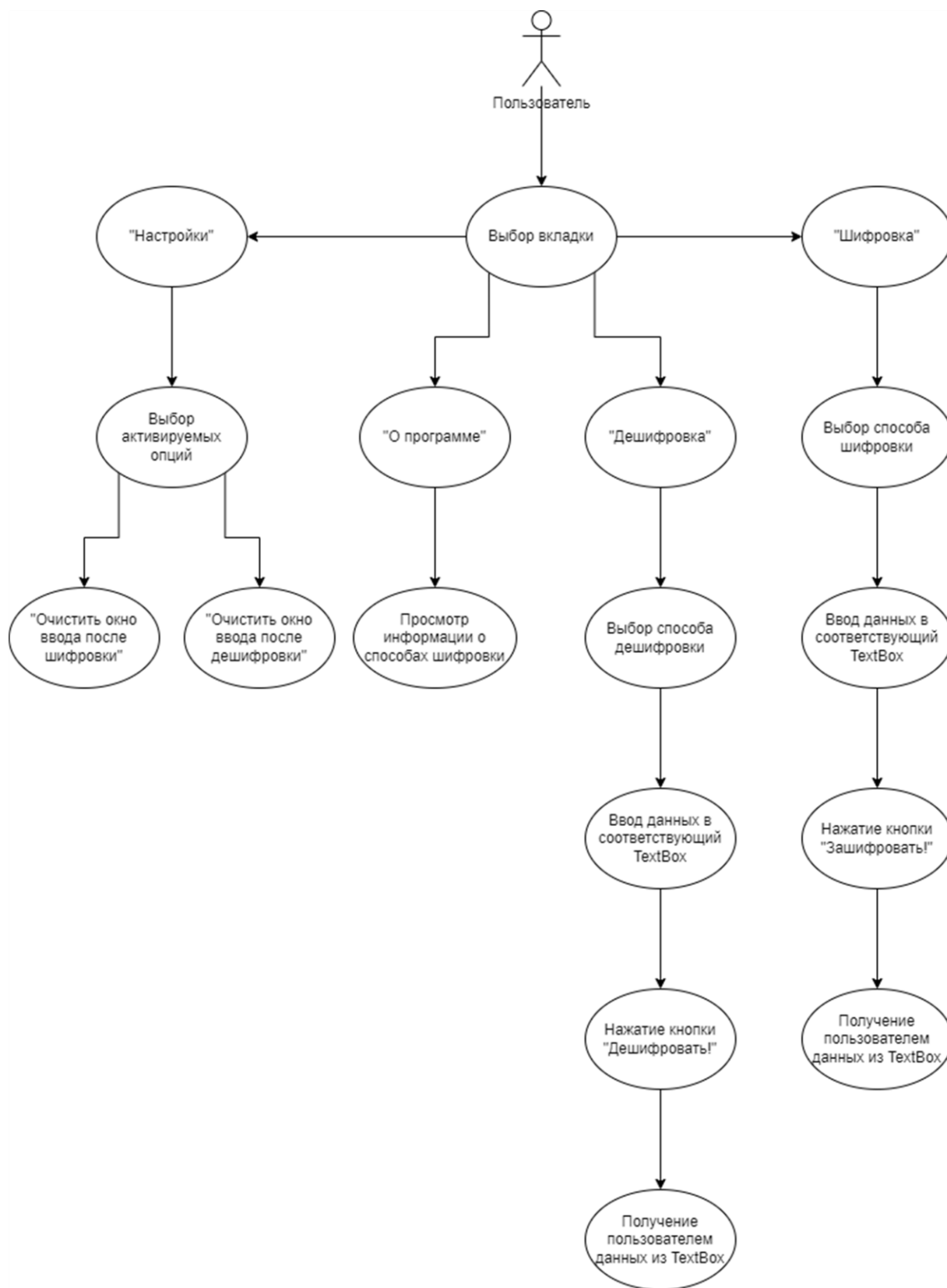


Рисунок 1 – диаграмма вариантов использования

Изм		№ докум.	Подпись	Дата

КП.22.09.02.07.603.12.ПЗ

Лист

16

2.3 Диаграммы состояний

Диаграмма состояний отражает переход информационной системы из одного состояния в другое.

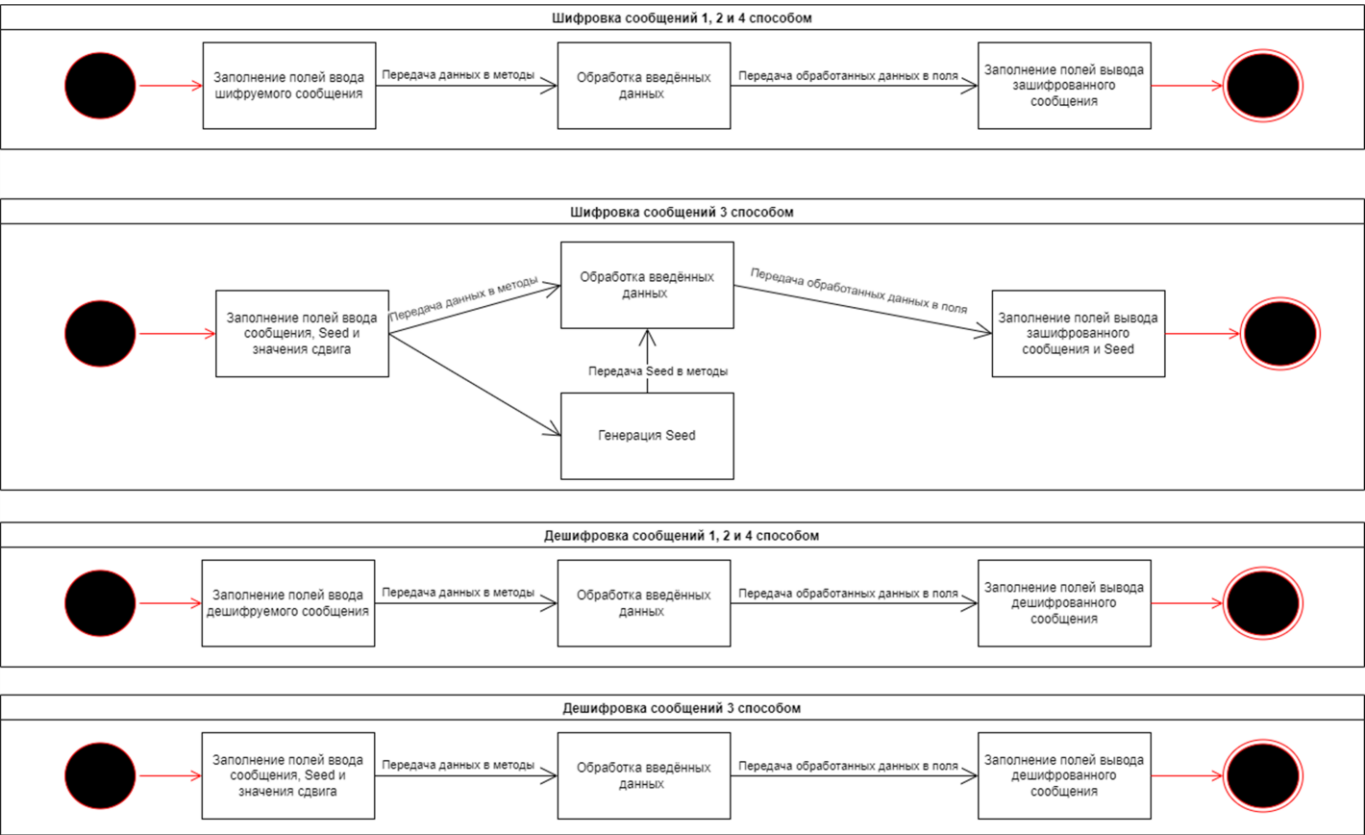


Рис. 2 – диаграмма состояний

2.4 Словари данных

Словарь данных для Dictionary<char, int> freqdict:

Индекс	Ключ	Тип данных	Значение	Тип данных
0		char	1	int
1	!	char	2	int
2	"	char	3	int
3	#	char	4	int
4	\$	char	5	int
5	%	char	6	int
6	&	char	7	int
7	'	char	8	int
8	(char	9	int
9)	char	10	int
10	*	char	11	int
11	+	char	12	int
12	,	char	13	int
13	-	char	14	int
14	.	char	15	int
15	/	char	16	int
16	0	char	17	int
17	1	char	18	int
18	2	char	19	int
19	3	char	20	int
20	4	char	21	int
21	5	char	22	int
22	6	char	23	int
23	7	char	24	int
24	8	char	25	int
25	9	char	26	int
26	:	char	27	int
27	;	char	28	int
28	<	char	29	int
29	=	char	30	int

30	>	char	31	int
31	?	char	32	int
32	@	char	33	int
33	A	char	34	int
34	B	char	35	int
35	C	char	36	int
36	D	char	37	int
37	E	char	38	int
38	F	char	39	int
39	G	char	40	int
40	H	char	41	int
41	I	char	42	int
42	J	char	43	int
43	K	char	44	int
44	L	char	45	int
45	M	char	46	int
46	N	char	47	int
47	O	char	48	int
48	P	char	49	int
49	Q	char	50	int
50	R	char	51	int
51	S	char	52	int
52	T	char	53	int
53	U	char	54	int
54	V	char	55	int
55	W	char	56	int
56	X	char	57	int
57	Y	char	58	int
58	Z	char	59	int
59	[char	60	int
60	\	char	61	int
61]	char	62	int
62	^	char	63	int
63	_	char	64	int
64	`	char	65	int
65	a	char	66	int
66	b	char	67	int
67	c	char	68	int
68	d	char	69	int
69	e	char	70	int
70	f	char	71	int
71	g	char	72	int
72	h	char	73	int
73	i	char	74	int
74	j	char	75	int
75	k	char	76	int
76	l	char	77	int
77	m	char	78	int
78	n	char	79	int
79	o	char	80	int

Изм		№ докум.	Подпись	Дата

КП.22.09.02.07.603.12.ПЗ

Лист

19

80	p	char	81	int
81	q	char	82	int
82	r	char	83	int
83	s	char	84	int
84	t	char	85	int
85	u	char	86	int
86	v	char	87	int
87	w	char	88	int
88	x	char	89	int
89	y	char	90	int
90	z	char	91	int
91	{	char	92	int
92		char	93	int
93	}	char	94	int
94	~	char	95	int
95	A	char	96	int
96	Б	char	97	int
97	B	char	98	int
98	Г	char	99	int
99	Д	char	100	int
100	Е	char	101	int
101	Ж	char	102	int
102	З	char	103	int
103	И	char	104	int
104	Й	char	105	int
105	К	char	106	int
106	Л	char	107	int
107	М	char	108	int
108	Н	char	109	int
109	О	char	110	int
110	П	char	111	int
111	Р	char	112	int
112	С	char	113	int
113	Т	char	114	int
114	У	char	115	int
115	Ф	char	116	int
116	Х	char	117	int
117	Ц	char	118	int
118	Ч	char	119	int
119	Ш	char	120	int
120	Щ	char	121	int
121	Ъ	char	122	int
122	Ы	char	123	int
123	Ь	char	124	int
124	Э	char	125	int
125	Ю	char	126	int
126	Я	char	127	int
127	a	char	128	int
128	б	char	129	int
129	в	char	130	int

130	г	char	131	int
131	д	char	132	int
132	е	char	133	int
133	ж	char	134	int
134	з	char	135	int
135	и	char	136	int
136	й	char	137	int
137	к	char	138	int
138	л	char	139	int
139	м	char	140	int
140	н	char	141	int
141	о	char	142	int
142	п	char	143	int
143	р	char	144	int
144	с	char	145	int
145	т	char	146	int
146	у	char	147	int
147	ф	char	148	int
148	х	char	149	int
149	ц	char	150	int
150	ч	char	151	int
151	ш	char	152	int
152	щ	char	153	int
153	ъ	char	154	int
154	ы	char	155	int
155	ь	char	156	int
156	э	char	157	int
157	ю	char	158	int
158	я	char	159	int

Словарь данных для List<char> positions_list

Индекс	Значение	Тип данных
0	t	char
1	h	char
2	e	char
3	q	char
4	u	char
5	i	char
6	c	char
7	k	char
8	b	char
9	r	char
10	o	char
11	w	char
12	n	char
13	f	char
14	x	char
15	j	char
16	m	char
17	p	char
18	s	char
19	v	char
20	l	char
21	a	char
22	z	char
23	y	char
24	d	char
25	g	char
26	T	char
27	H	char
28	E	char
29	Q	char
30	U	char
31	I	char
32	C	char
33	K	char
34	B	char
35	R	char
36	O	char
37	W	char
38	N	char
39	F	char
40	X	char
41	J	char
42	M	char
43	P	char
44	S	char
45	V	char

Изм		№ докум.	Подпись	Дата

КП.22.09.02.07.603.12.ПЗ

Лист

22

46	L	char
47	A	char
48	Z	char
49	Y	char
50	D	char
51	G	char
52	с	char
53	ъ	char
54	е	char
55	ш	char
56	ь	char
57	ж	char
58	щ	char
59	ё	char
60	э	char
61	т	char
62	и	char
63	х	char
64	м	char
65	я	char
66	г	char
67	к	char
68	ф	char
69	р	char
70	а	char
71	н	char
72	ц	char
73	у	char
74	з	char
75	б	char
76	л	char
77	о	char
78	д	char
79	в	char
80	ы	char
81	п	char
82	й	char
83	ч	char
84	ю	char
85	С	char
86	Ъ	char
87	Е	char
88	Ш	char
89	Ь	char
90	Ж	char
91	Щ	char
92	Ё	char
93	Э	char
94	Т	char
95	И	char
96	Х	char

					КП.22.09.02.07.603.12.ПЗ	Лист
						23
Изм		№ докум.	Подпись	Дата		

97	М	char
98	Я	char
99	Г	char
100	К	char
101	Ф	char
102	Р	char
103	А	char
104	Н	char
105	Ц	char
106	У	char
107	З	char
108	Б	char
109	Л	char
110	О	char
111	Д	char
112	В	char
113	Ы	char
114	П	char
115	Й	char
116	Ч	char
117	Ю	char
118		char
119	,	char
120	!	char
121	.	char
122	?	char
123	-	char
124	_	char
125	<	char
126	>	char
127	[char
128]	char
129	{	char
130	}	char
131	+	char
132	=	char
133	\$	char
134	@	char
135	%	char
136	:	char
137	(char
138)	char
139	"	char
140	1	char
141	2	char
142	3	char
143	4	char
144	5	char
145	6	char
146	7	char
147	8	char
148	9	char
149	0	char

					КП.22.09.02.07.603.12.ПЗ	Лист
						24
Изм		№ докум.	Подпись	Дата		

Словарь данных для List<char> positions_list_4

Индекс	Значение	Тип данных
0	a	char
1	b	char
2	c	char
3	d	char
4	e	char
5	f	char
6	g	char
7	h	char
8	i	char
9	j	char
10	k	char
11	l	char
12	m	char
13	n	char
14	o	char
15	p	char
16	q	char
17	r	char
18	s	char
19	t	char
20	u	char
21	v	char
22	w	char
23	x	char
24	y	char
25	z	char
26	A	char
27	B	char
28	C	char
29	D	char
30	E	char
31	F	char
32	G	char
33	H	char
34	I	char
35	J	char
36	K	char
37	L	char
38	M	char
39	N	char
40	O	char
41	P	char
42	Q	char
43	R	char

Изм		№ докум.	Подпись	Дата

КП.22.09.02.07.603.12.ПЗ

Лист

25

44	S	char
45	T	char
46	U	char
47	V	char
48	W	char
49	X	char
50	Y	char
51	Z	char
52	a	char
53	б	char
54	в	char
55	г	char
56	д	char
57	е	char
58	ё	char
59	ж	char
60	з	char
61	и	char
62	й	char
63	к	char
64	л	char
65	м	char
66	н	char
67	о	char
68	п	char
69	р	char
70	с	char
71	т	char
72	у	char
73	ф	char
74	х	char
75	ц	char
76	ч	char
77	ш	char
78	щ	char
79	ъ	char
80	ы	char
81	ь	char
82	э	char
83	ю	char
84	я	char
85	A	char
86	Б	char
87	В	char
88	Г	char
89	Д	char
90	Е	char
91	Ё	char
92	Ж	char
93	З	char

					КП.22.09.02.07.603.12.ПЗ	Лист
						26
Изм		№ докум.	Подпись	Дата		

94	И	char
95	Й	char
96	К	char
97	Л	char
98	М	char
99	Н	char
100	О	char
101	П	char
102	Р	char
103	С	char
104	Т	char
105	У	char
106	Ф	char
107	Х	char
108	Ц	char
109	Ч	char
110	Ш	char
111	Щ	char
112	Ъ	char
113	Ы	char
114	Ь	char
115	Э	char
116	Ю	char
117	Я	char
118	1	char
119	2	char
120	3	char
121	4	char
122	5	char
123	6	char
124	7	char
125	8	char
126	9	char
127	0	char
128		char
129	,	char
130	!	char
131	.	char
132	?	char
133	-	char
134	_	char
135	<	char
136	>	char
137	[char
138]	char
139	{	char
140	}	char
141	+	char
142	=	char
143	\$	char

					КП.22.09.02.07.603.12.ПЗ	Лист
						27
Изм		№ докум.	Подпись	Дата		

144	@	char
145	%	char
146	:	char
147	(char
148)	char
149	"	char

					КП.22.09.02.07.603.12.ПЗ	Лист
						28
Изм		№ докум.	Подпись	Дата		

2.5 Тестовые сценарии

Название проекта	«Шифратор сообщений»
Номер версии	1.0
Имя тестирующего	Левицкий Павел Дмитриевич
Даты тестирования	7 июня 2022 г., 3:20:06... 20 августа 2022 г., 17:19:04

Тестовый сценарий №1:

Номер тестового случая	tc_1
Приоритет	Высокий
Название	Шифровка сообщения первым способом
Резюме испытания	Пользователь должен получить обработанное первым методом сообщение
Шаги тестирования	<ol style="list-style-type: none">1. Запуск программы2. Ввод пользователем незашифрованного сообщения в первое окно ввода шифруемого сообщения3. Нажатие пользователем кнопки «Зашифровать!»
Ожидаемый результат	Вывод в окно вывода последовательности чисел, разделенных пробелом
Фактический результат	Вывод в окно вывода последовательности чисел, разделенных пробелом

					КП.22.09.02.07.603.12.ПЗ	Лист
						29
Изм		№ докум.	Подпись	Дата		

Предпосылки	Поля не должны быть пустыми, должны соблюдаться условия для ввода пользовательских сообщений
Статус	Пройдено
Комментарии	<ol style="list-style-type: none"> 1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 2. Протестирован ввод пользователем некорректного сообщения, результат – вывод сообщения о некорректно заполненном поле ввода.

Номер тестового случая	tc_2
Приоритет	Высокий
Название	Дешифровка сообщения, зашифрованного первым способом
Резюме испытания	Пользователь должен получить дешифровку сообщения, ранее зашифрованного первым методом
Шаги тестирования	<ol style="list-style-type: none"> 1. Запуск программы 2. Ввод пользователем зашифрованного сообщения в первое окно ввода дешифруемого сообщения 3. Нажатие пользователем кнопки «Дешифровать!»
Ожидаемый результат	Вывод в окно вывода корректно дешифрованного сообщения
Фактический результат	Вывод в окно вывода корректно дешифрованного сообщения
Предпосылки	Поля не должны быть пустыми
Статус	Пройдено
Комментарии	<ol style="list-style-type: none"> 1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 2. Протестирован ввод пользователем некорректного сообщения, результат – вывод некорректного сообщения.

					КП.22.09.02.07.603.12.ПЗ	Лист
						30
Изм		№ докум.	Подпись	Дата		

Номер тестового случая	tc_3
Приоритет	Высокий
Название	Шифровка сообщения вторым способом
Резюме испытания	Пользователь должен получить обработанное вторым методом сообщение
Шаги тестирования	<ol style="list-style-type: none"> 1. Запуск программы 2. Ввод пользователем незашифрованного сообщения во второе окно ввода шифруемого сообщения 3. Нажатие пользователем кнопки «Зашифровать!»
Ожидаемый результат	Вывод в окно вывода последовательности чисел, разделенных пробелом
Фактический результат	Вывод в окно вывода последовательности чисел, разделенных пробелом
Предпосылки	Поля не должны быть пустыми, должны соблюдаться условия для ввода пользовательских сообщений
Статус	Пройдено
Комментарии	<ol style="list-style-type: none"> 1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 2. Протестирован ввод пользователем некорректного сообщения, результат – вывод сообщения о некорректно заполненном поле ввода.

Номер тестового случая	tc_4
Приоритет	Высокий
Название	Дешифровка сообщения, зашифрованного вторым способом
Резюме испытания	Пользователь должен получить дешифровку сообщения, ранее зашифрованного вторым методом
Шаги тестирования	<ol style="list-style-type: none"> 1. Запуск программы 2. Ввод пользователем зашифрованного сообщения во второе окно ввода дешифруемого сообщения 3. Нажатие пользователем кнопки «Дешифровать!»

Ожидаемый результат	Вывод в окно вывода корректно дешифрованного сообщения
Фактический результат	Вывод в окно вывода корректно дешифрованного сообщения
Предпосылки	Поля не должны быть пустыми
Статус	Пройдено
Комментарии	<ol style="list-style-type: none"> 1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 2. Протестирован ввод пользователем некорректного сообщения, результат – вывод некорректного сообщения.

Номер тестового случая	tc_5
Приоритет	Высокий
Название	Шифровка сообщения третьим способом
Резюме	Пользователь должен получить обработанное третьим методом сообщение
Шаги тестирования	<ol style="list-style-type: none"> 1. Запуск программы 2. Ввод пользователем незашифрованного сообщения в третье окно ввода шифруемого сообщения 3. Ввод значения сдвига в окне ввода сдвига, следуя указаниям 4. Нажатие пользователем кнопки «Зашифровать!»
Ожидаемый результат	Вывод в окно вывода последовательности чисел и символов
Фактический результат	Вывод в окно вывода последовательности чисел и символов
Предпосылки	Поля не должны быть пустыми, должны соблюдаться условия для ввода пользовательских сообщений
Статус	Пройдено
Комментарии	<ol style="list-style-type: none"> 1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 2. Протестирован ввод пользователем некорректного сообщения, результат – вывод сообщения о некорректно заполненном поле ввода.

					КП.22.09.02.07.603.12.ПЗ	Лист
Изм		№ докум.	Подпись	Дата		32

Номер тестового случая	tc_6
Приоритет	Высокий
Название	Дешифровка сообщения, зашифрованного третьим способом
Резюме	Пользователь должен получить дешифровку сообщения, ранее зашифрованного третьим методом
Шаги тестирования	<ol style="list-style-type: none"> 1. Запуск программы 2. Ввод пользователем зашифрованного сообщения в третье окно ввода дешифруемого сообщения 3. Ввод соответствующего значения в поле ввода Seed 4. Ввод соответствующего значения сдвига в поле ввода сдвига 5. Нажатие пользователем кнопки «Дешифровать!»
Ожидаемый результат	Вывод в окно вывода корректно дешифрованного сообщения
Фактический результат	Вывод в окно вывода корректно дешифрованного сообщения
Предпосылки	Поля не должны быть пустыми
Статус	Пройдено
Комментарии	<ol style="list-style-type: none"> 1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 2. Протестирован ввод пользователем некорректного сообщения, результат – вывод некорректного сообщения.

Номер тестового случая	tc_7
Приоритет	Высокий
Название	Шифровка сообщения четвертым способом
Резюме	Пользователь должен получить обработанное четвертым методом сообщение
Шаги тестирования	<ol style="list-style-type: none"> 1. Запуск программы 2. Ввод пользователем зашифрованного сообщения в четвертое окно ввода шифруемого сообщения 3. Нажатие пользователем кнопки «Дешифровать!»
Ожидаемый результат	Вывод в окно вывода корректно зашифрованного сообщения
Фактический результат	Вывод в окно вывода корректно зашифрованного сообщения
Предпосылки	Поля не должны быть пустыми
Статус	Пройдено
Комментарии	<ol style="list-style-type: none"> 1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 2. Протестирован ввод пользователем некорректного сообщения, результат – вывод сообщения о некорректно заполненном поле ввода.

Номер тестового случая	tc_8
Приоритет	Высокий
Название	Дешифровка сообщения, зашифрованного четвертым способом
Резюме	Пользователь должен получить дешифровку сообщения, ранее зашифрованного четвертым методом
Шаги тестирования	6. Запуск программы 7. Ввод пользователем зашифрованного сообщения в четвертое окно ввода дешифруемого сообщения 4. Нажатие пользователем кнопки «Дешифровать!»
Ожидаемый результат	Вывод в окно вывода корректно дешифрованного сообщения
Фактический результат	Вывод в окно вывода корректно дешифрованного сообщения
Предпосылки	Поля не должны быть пустыми
Статус	Пройдено
Комментарии	1. Протестирован ввод пользователем пустого сообщения, результат – вывод сообщения о некорректно заполненном поле ввода. 3. Протестирован ввод пользователем некорректного сообщения, результат – вывод сообщения о некорректно заполненном поле ввода.

					КП.22.09.02.07.603.12.ПЗ	Лист
						35
Изм		№ докум.	Подпись	Дата		

Заключение

В ходе написания данного курсового проекта был разработан продукт «Шифратор сообщений» и были протестированы такие его функции, как:

- шифровка сообщений первым способом;
- шифровка сообщений вторым способом;
- шифровка сообщений третьим способом;
- шифровка сообщений четвертым способом.
- дешифровка сообщений, зашифрованных первым способом;
- дешифровка сообщений, зашифрованных вторым способом;
- дешифровка сообщений, зашифрованных третьим способом;
- дешифровка сообщений, зашифрованных четвертым способом.

Отдельно были написаны модульные тесты для всех методов, возвращающих значения (см. приложение А).

Из вышеизложенного можно сделать вывод о достижении поставленных задач.

					КП.22.09.02.07.603.12.ПЗ	Лист
						36
Изм		№ докум.	Подпись	Дата		

Список использованных источников

Ниже перечислены источники, использовавшиеся в ходе разработки документации и приложения для данного курсового проекта:

- Построение UML-диаграмм:
<https://habr.com/ru/post/566218/>
<https://nationalteam.worldskills.ru/skills/proektirovanie-diagrammy-sostoyaniy-uml-statechart-diagram/>
- Составление тестовых сценариев:
<https://habr.com/ru/post/587620/>
<https://habr.com/ru/post/549054/>
- Модульное тестирование:
<https://docs.microsoft.com/ru-ru/visualstudio/test/walkthrough-creating-and-running-unit-tests-for-managed-code?view=vs-2022>
<https://metanit.com/sharp/mvc5/18.3.php>
<https://habr.com/ru/post/191986/>
- Бизнес-логика:
<https://metanit.com/sharp/>
<https://habr.com/ru/post/232009/>
<https://metanit.com/sharp/tutorial/2.14.php>
- Работа с интерфейсом:
<https://docs.microsoft.com/ru-ru/dotnet/desktop/wpf/introduction-to-wpf?view=netframeworkdesktop-4.8>
<https://metanit.com/sharp/wpf/1.php>
- Основы и принципы криптографии:
<http://sumk.ulstu.ru/docs/mszki/www.college.ru/UDP/texts/zi04.html>
- Составление документации:
<https://intuit.ru/studies/courses/2195/55/lecture/15050?page=2>

1. AbbcccStringBuilderTest()

Код теста:

```
[TestMethod()]
public void AbbcccStringBuilderTest()
{
    //arrange
    string expected = " !!\\";"
```

Результат выполнения:

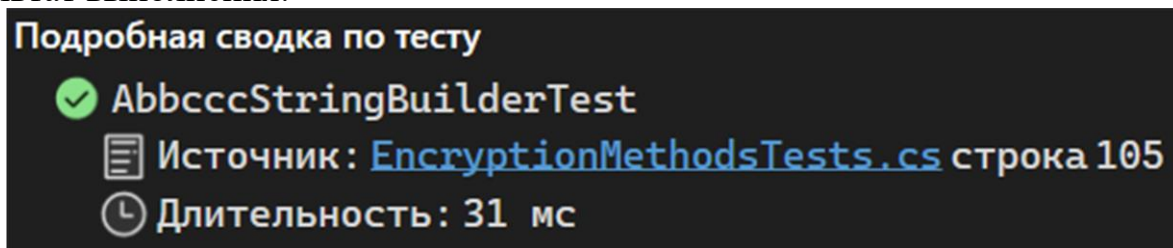


Рис. 3 – результат выполнения теста метода `AbbcccStringBuilder()`

2. DecryptAs1Test()

Код теста:

```
[TestMethod()]
public void DecryptAs1Test()
{
    //arrange
    string input_encrypted = "84 69 83 84 77 69 83 83 65 71 69";
    string expected = "testmessage";
    List<char> freqsequence_list = new()
    {
        ' ', '!', '"', '#', '$', '%', '&', '\'', '(', ')', '*', '+', ',', '-', '.', '/', '0', '1', '2', '3',
        '4', '5', '6', '7', '8', '9', ':', ';', '<', '=', '>', '?', '@', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',
        'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', '[', '\\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g',
        'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~', 'А', 'Б', 'В',
        'Г', 'Д', 'Е', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Ъ', 'Ы', 'Ь',
        'Э', 'Ю', 'Я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц',
        'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я'
    };

    //action
    string result = MainWindow.DecryptAs1(freqsequence_list, input_encrypted);

    //assert
    Assert.AreEqual(expected, result);
}
```

Результат выполнения:

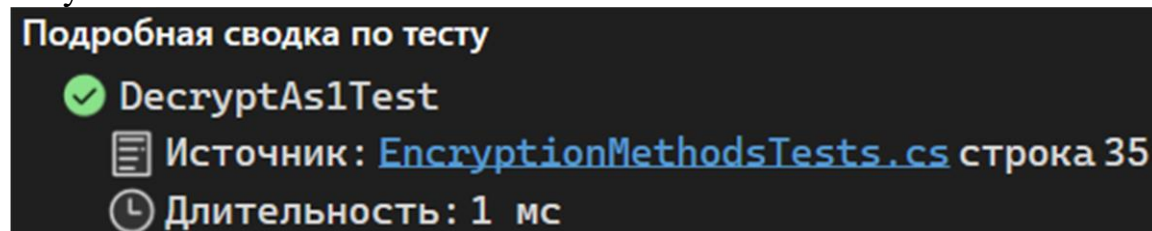


Рис. 4 – результат выполнения теста метода DecryptAs1()

3. DecryptAs2Test()

Код теста:

```
[TestMethod()]
public void DecryptAs2Test()
{
    //arrange
    string input_encrypted = "0 2 18 0 16 2 18 18 21 25 2";
    string expected = "testmessage";
    List<char> positions_list = new()
    {
        't','h','e','q','u','i','c','k','b','r','o','w','n','f','x','j','m','p','s','v',
        'l','a','z','y','d','g','t','h',
        'e','q','u','i','c','k','b','r','o','w','n','f','x','j','m','p','s','v','l','a',
        'z','y','d','g','c','b','e','s',
        'b','j','s','e','t','i','x','m','y','g','k','f','p','a','n','c','y','z','b',
        'l','o','d','v','y','p','y','c',
        'y','c','b','e','s','b','j','s','e','t','i','x','m','y','g','k','f','p','a',
        'n','c','y','z','b','l','o','d',
        'v','y','p','y','c','y','i','!','.', '?','_ ',
        ' _','<','>','[',']','{','}','+','=','','$','@','%',':','(',')','"',' ',
        '1','2','3','4','5','6','7','8','9','0'
    };

    //action
    string result = MainWindow.DecryptAs2(positions_list, input_encrypted);

    //assert
    Assert.AreEqual(expected, result);
}
```

Результат выполнения:

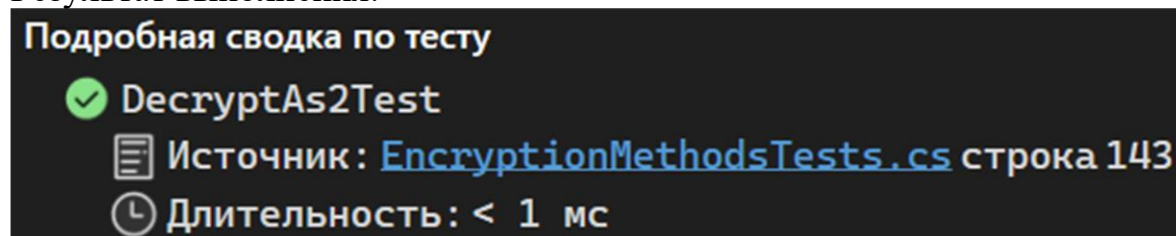


Рис. 5 – результат выполнения теста метода DecryptAs2()

4. DecryptAs3Test()

Код теста:


```
[TestMethod()]
public void DecryptAs3Test()
{
    //arrange
    string input_encrypted = "111!-63-!96!-63-!110!-63-!111!-63-!104!-63-!96!-63-!110!-63-!110!-63-!92!-63-!98!-63-!96!-63-!";
    int ascii_shift = 5;
    string seed_string = "!-63-!";
    string output_string = "";
    string expected = "testmessage";


    //action
    string result = MainWindow.DecryptAs3(seed_string, output_string, input_encrypted,
    ascii_shift);

    //assert
    Assert.AreEqual(expected, result);
}
```

Результат выполнения:

Подробная сводка по тесту


DecryptAs3Test


Источник: [EncryptionMethodsTests.cs](#) строка 203



Длительность: < 1 мс

Рис. 6 – результат выполнения теста метода DecryptAs3()

5. EncryptAs1Test()

Код теста:

```
[TestMethod()]
public void EncryptAs1Test()
{
    //arrange
    string input_string = "testmessage";
    string output_encrypted = "";
    string expected = "84 69 83 84 77 69 83 83 65 71 69";
    List<char> freqsequence_list = new()
    {
        ' ', '!', '"', '#', '$', '%', '&', '\'', '(', ')', '*', '+', ',', '-',
        '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
        ':', ';', '<', '=', '>', '?', '@', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',
        'N', 'O', 'P', 'Q', 'R', 'S',
        'T', 'U', 'V', 'W', 'X', 'Y', 'Z', '[', '\\', ']', '^', '_', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g',
        'h', 'i', 'j', 'k', 'l', 'm',
        'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', '{', '|', '}', '~', 'A', 'Б', 'В',
        'Г', 'Д', 'Е', 'Ж', 'З', 'И',
        'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Ъ', 'Ы', 'Ь',
        'Э', 'Ю', 'Я', 'а', 'б', 'в',
        'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц',
        'ч', 'ш', 'щ', 'ъ', 'ы', 'ь',
        'э', 'ю', 'я'
    };

    //action
    MainWindow.EncryptAs1(freqsequence_list, input_string, ref output_encrypted);
    string result = MainWindow.LastSpaceCutter(output_encrypted);

    //assert
    Assert.AreEqual(expected, result);
}
```

Результат выполнения:

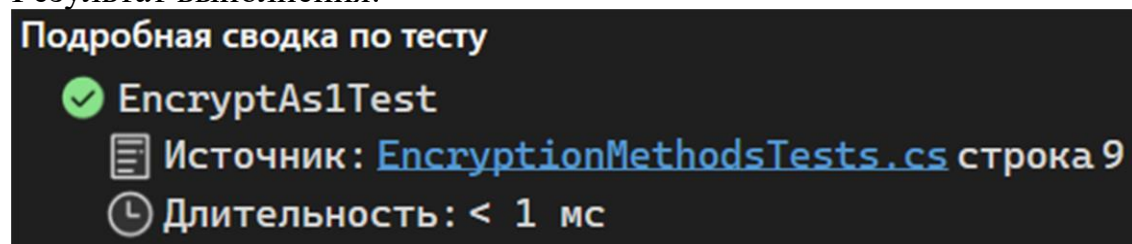


Рис. 7 – результат выполнения теста метода EncryptAs1()

					КП.22.09.02.07.603.12.ПЗ	Лист
						41
Изм.		№ докум.	Подпись	Дата		

6. EncryptAs2Test()

Код теста:


```
[TestMethod()]
public void EncryptAs2Test()
{
    //arrange
    string input_string = "testmessage";
    string output_encrypted = "";
    string expected = "0 2 18 0 16 2 18 18 21 25 2";
    List<char> positions_list = new()
    {
        't','h','e','q','u','i','c','k','b','r','o','w','n','f','x','j','m','p','s','v',
        'l','a','z','y','d','g','T','H',
        'E','Q','U','I','C','K','B','R','O','W','N','F','X','J','M','P','S','V','L','A',
        'Z','Y','D','G','c','ъ','e','ш',
        'ь','ж','щ','ё','э','т','и','х','м','я','г','к','ф','р','а','н','ц','у','з','б',
        'л','о','д','в','ы','п','й','ч',
        'ю','с','ъ','е','ш','ь','ж','щ','ё','э','т','и','х','м','я','г','к','ф','р','а',
        'н','ц','у','з','б','л','о','д',
        'в','ы','п','й','ч','ю',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',
        ' ','_','<','>','[',' ',']','{',' ','}','+','=','','$','@','%',':','(',' ',')',"'',
        '1','2','3','4','5','6','7','8','9','0'
    };


    //action
    MainWindow.EncryptAs2(positions_list, input_string, ref output_encrypted);
    string result = MainWindow.LastSpaceCutter(output_encrypted);

    //assert
    Assert.AreEqual(expected, result);
}
```

Результат выполнения:

Подробная сводка по тесту


EncryptAs2Test


Источник: [EncryptionMethodsTests.cs](#) строка 118



Длительность: < 1 мс

Рис. 8 – Результат выполнения теста метода EncryptAs2()

7. EncryptAs3Test()

Код теста:

```
[TestMethod()]
public void EncryptAs3Test()
{
    //arrange
    string input_string = "testmessage";
    int ascii_shift = 5;
    string seed_string = "!-63-!";
    string expected = "111!-63-!96!-63-!110!-63-!111!-63-!104!-63-!96!-63-!110!-63-!110!-63-!92!-63-!98!-63-!96!-63-!";

    //action
    string result = MainWindow.EncryptAs3(input_string, ascii_shift, seed_string);
    //string result = MainWindow.LastSpaceCutter(output_encrypted);

    //assert
    Assert.AreEqual(expected, result);
}
```

Результат выполнения:

Подробная сводка по тесту

✓ **EncryptAs3Test**

📄 Источник: [EncryptionMethodsTests.cs](#) строка 186

🕒 Длительность: < 1 мс

Рис. 9 – Результат выполнения теста метода EncryptAs3()

8. FreqSequenceListBuilderTest()

Код теста:

```
[TestMethod()]
public void FreqSequenceListBuilderTest()
{
    //arrange
    Dictionary<char, int> input = new()
    {
        {' ', 1},
        {'!', 2},
        {'"', 3},
        {'#', 4}
    };
    List<char> expected = new()
    {
        ' ',
        '!',
        '"',
        '#'
    };

    //action
    List<char> result = MainWindow.FreqSequenceListBuilder(input);

    //assert
    Assert.IsTrue(Enumerable.SequenceEqual(expected, result));
}
```

Результат выполнения:

Подробная сводка по тесту

✓ **FreqSequenceListBuilderTest**

📄 Источник: [EncryptionMethodsTests.cs](#) строка 59

🕒 Длительность: 1 мс

Рис. 10 – результат выполнения теста метода FreqSequenceListBuilder()

					КП.22.09.02.07.603.12.ПЗ	Лист
						44
Изм		№ докум.	Подпись	Дата		

9. InternalSeedBuilderTest()

Код теста:

```
[TestMethod()]
public void InternalSeedBuilderTest()
{
    //arrange
    char input = (char)85;
    string expected = "!--85-!";

    //action
    string actual = MainWindow.SeedStringBuilder(input);

    //assert
    Assert.AreEqual(expected, actual);
}
```

Результат выполнения:

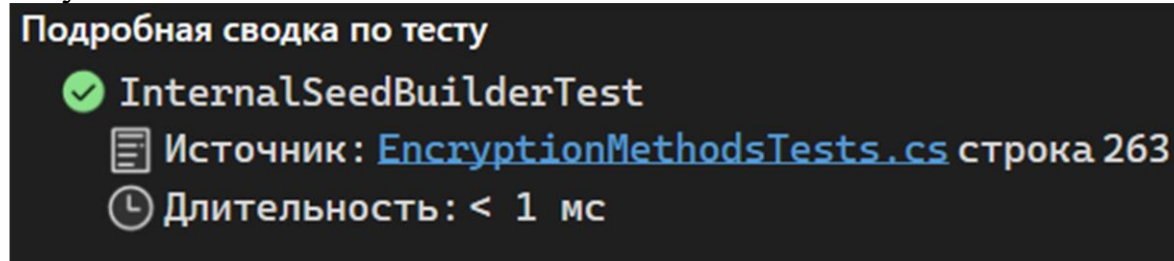


Рис. 11 – результат выполнения теста метода InternalSeedBuilder()

10. LastSpaceCutterTest()

Код теста:

```
[TestMethod()]
public void LastSpaceCutterTest()
{
    //arrange
    string input = "string ";
    string expected = "string";

    //action
    string output = MainWindow.LastSpaceCutter(input);

    //assert
    Assert.AreEqual(expected, output);
}
```

Результат выполнения:

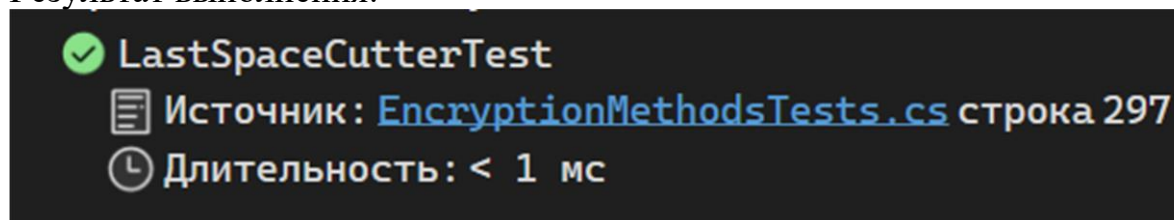


Рис. 12 – результат выполнения теста метода LastSpaceCutter()

					КП.22.09.02.07.603.12.ПЗ	Лист
Изм		№ докум.	Подпись	Дата		45

11. PositionsList2GeneratorTest()

Код теста:

```
[TestMethod()]
public void PositionsList2GeneratorTest()
{
    //arrange
    List<char> expected = new()
    {
        'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t',
        'u','v','w','x','y','z',
        'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T',
        'U','V','W','X','Y','Z',
        'а','б','в','г','д','е','ё','ж','з','и','й','к','л','м','н','о','п','р','с','т',
        'у','ф','х','ц','ч','ш','щ','ъ','ы','ь','э','ю','я',
        'А','Б','В','Г','Д','Е','Ё','Ж','З','И','Й','К','Л','М','Н','О','П','Р','С','Т',
        'У','Ф','Х','Ц','Ч','Ш','Щ','Ъ','Ы','Ь','Э','Ю','Я',
        '1','2','3','4','5','6','7','8','9','0',' ','!',' ','?','-',
        '._','<','>','[',']','{','}','+','=','','$','@','%',':','(',')','"','\''
    };

    //action
    List<char> actual = MainWindow.PositionsList2Generator();

    //assert
    Assert.IsTrue(expected.SequenceEqual(actual));
}
```

Результат выполнения:

Подобная сводка по тесту

✓ **PositionsList2GeneratorTest**

📄 Источник: [EncryptionMethodsTests.cs](#) строка 277

🕒 Длительность: < 1 мс

Рис. 13 – результат выполнения теста метода PositionsList2Generator()

12. PositionsListGeneratorTest()

Код теста:


```
[TestMethod()]
public void PositionsListGeneratorTest()
{
    //arrange
    List<char> expected = new()
    {
        't','h','e','q','u','i','c','k','b','r','o','w','n','f','x','j','m','p','s','v',
        'l','a','z','y','d','g',
        'T','H','E','Q','U','I','C','K','B','R','O','W','N','F','X','J','M','P','S','V',
        'L','A','Z','Y','D','G',
        'с','ъ','е','ш','ь','ж','щ','ё','э','т','и','х','м','я','г','к','ф','р','а','н',
        'ц','у','з','б','л','о','д','в','ы','п','й','ч','ю',
        'С','Ъ','Е','Ш','Ь','Ж','Щ','Ё','Э','Т','И','Х','М','Я','Г','К','Ф','Р','А','Н',
        'Ц','У','З','Б','Л','О','Д','В','Ы','П','Й','Ч','Ю',
        '?', '-',
        '<', '>', '[', ']', '{', '}', '+', '=', '$', '@', '%', ':', '(', ')', '"', '1', '2', '3', '4', '5',
        '6', '7', '8', '9', '0'
    };


    //action
    List<char> actual = MainWindow.PositionsListGenerator();

    //assert
    Assert.IsTrue(expected.SequenceEqual(actual));
}
```

Результат выполнения:

Подробная сводка по тесту

 **PositionsListGeneratorTest**

 **Источник:** [EncryptionMethodsTests.cs](#) строка 166


 **Длительность:** < 1 мс

Рис. 14 – результат выполнения теста метода PositionsListGenerator()

13. SeedGeneratorTest()

Код теста:

```
[TestMethod()]
public void SeedGeneratorTest()
{
    //arrange
    int expected1 = 33;
    int expected2 = 125;

    //action
    int actual = MainWindow.SeedGenerator();
    //assert

    //assert
    Assert.IsTrue((actual >= expected1) && (actual <= expected2));
}
```

Результат выполнения:

Подробная сводка по тесту

✓ **SeedGeneratorTest**

📄 Источник: [EncryptionMethodsTests.cs](#) строка 234

🕒 Длительность: < 1 мс

Рис. 15 – результат выполнения теста метода SeedGenerator()

					КП.22.09.02.07.603.12.ПЗ	Лист
Изм		№ докум.	Подпись	Дата		48

14. SeedStringBuilderTest()

Код теста:

```
[TestMethod()]
public void SeedStringBuilderTest()
{
    //assert
    char input = 'U';
    string expected = "!-85-!";

    //action
    string actual = MainWindow.SeedStringBuilder(input);

    //assert
    Assert.AreEqual(expected, actual);
}
```

Результат выполнения:

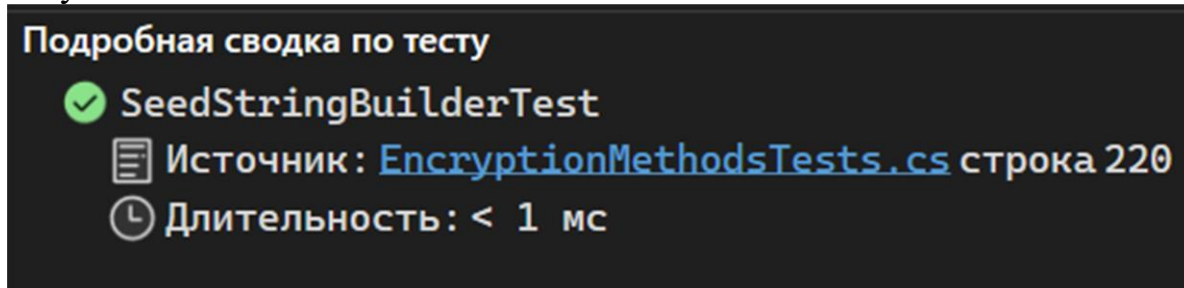


Рис. 16 – результат выполнения теста метода SeedStringBuilder()

15. SymbolRepeat_CounterTest()

Код теста:

```
[TestMethod()]
public void SymbolRepeat_CounterTest()
{
    //arrange
    string input = " !!\\";
    Dictionary<char, int> expected = new()
    {
        {' ', 1},
        {'!', 2},
        {'"', 3},
        {'#', 4}
    };

    //action
    Dictionary<char, int> result = MainWindow.SymbolRepeatCounter(input);

    //assert
    Assert.IsTrue(Enumerable.SequenceEqual(expected, result));
}
```

Результат выполнения:

Подробная сводка по тесту



SymbolRepeat_CounterTest



Источник: [EncryptionMethodsTests.cs](#) строка 85



Длительность: 1 мс

Рис. 17 – результат выполнения теста метода SymbolRepeat_Counter

					КП.22.09.02.07.603.12.ПЗ	Лист
						50
Изм		№ докум.	Подпись	Дата		

16. UserSeedOutTest()

Код теста:

```
[TestMethod()]
public void UserSeedOutTest()
{
    //arrange
    int input = 85;
    string expected = "U";

    //action
    string actual = MainWindow.UserSeedOut(input);

    //assert
    Assert.AreEqual(expected, actual);
}
```

Результат выполнения:

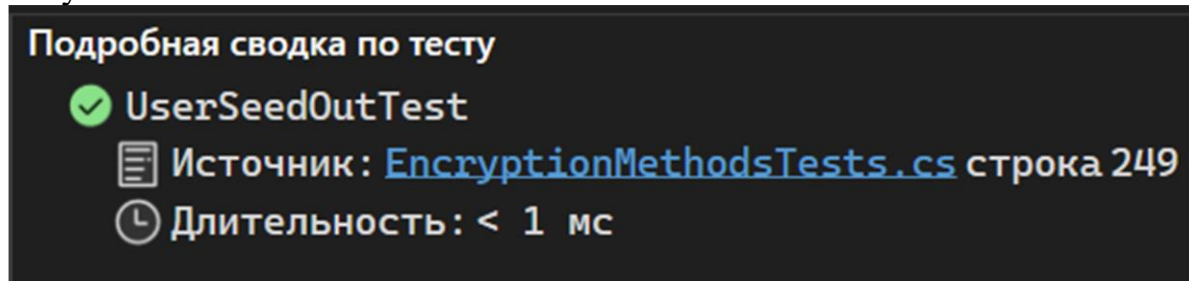


Рис. 18 – результат выполнения теста метода UserSeedOut()

					КП.22.09.02.07.603.12.ПЗ	Лист
						51
Изм		№ докум.	Подпись	Дата		

Содержимое файла MainWindow.xaml.cs:

```
using System;
using System.Collections.Generic;
using System.Windows;

namespace Encryptor_with_GUI
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
            userseed_output.IsReadOnly = true;
        }

        public void ClearInputAfterEnc_Checkbox_Checked(object sender, RoutedEventArgs e)
        {
            //
        }

        private void ClearInputAfterDec_Checkbox_Checked(object sender, RoutedEventArgs e)
        {
            //
        }

        //шифровка
        public void FirstEncBtnClick(object sender, RoutedEventArgs e)
        {
            FreqDictGenerator(out Dictionary<char, int> freqdict, out List<char> freqsequence_list);
            string input_string = first_enc_input.Text;
            string output_encrypted = "";
            EncryptAs1(freqsequence_list, input_string, ref output_encrypted);
            first_enc_output.Text = LastSpaceCutter(output_encrypted);

            if (ClearInputAfterEnc_Checkbox.IsChecked == true)
            {
                first_enc_input.Clear();
            }
        }

        public void SecondEncBtnClick(object sender, RoutedEventArgs e)
        {
            List<char> positions_list = PositionsListGenerator();
            string input_string = second_enc_input.Text;
            string output_encrypted = "";
            EncryptAs2(positions_list, input_string, ref output_encrypted);
            second_enc_output.Text = LastSpaceCutter(output_encrypted);

            if (ClearInputAfterEnc_Checkbox.IsChecked == true)
            {
                second_enc_input.Clear();
            }
        }
    }
}
```

```

public void ThirdEncBtnClick(object sender, RoutedEventArgs e)
{
    string input_string = third_enc_input.Text;
    //генерация межсимвольных значений
    int seed = SeedGenerator();
    //составление внутреннего seed с маркерами (!--n--
!) и вывод пользовательского seed
    try
    {
        int ascii_shift = Convert.ToInt32(ascii_shift_input.Text);
        string seed_string = InternalSeedBuilder(seed);
        userseed_output.Text = UserSeedOut(seed);
        string output_encrypted = EncryptAs3(input_string, ascii_shift, seed_string);
        third_enc_output.Text = output_encrypted;

        if (ClearInputAfterEnc_Checkbox.IsChecked == true)
        {
            third_enc_input.Clear();
        }
    }
    catch
    {
        ErrorMessage();
    }
}

public void FourthEncBtnClick(object sender, RoutedEventArgs e)
{
    string output_encrypted = "";
    List<char> positions_list_4 = PositionsList2Generator();
    string input_string = fourth_enc_input.Text;
    EncryptAs2(positions_list_4, input_string, ref output_encrypted);
    fourth_enc_output.Text = LastSpaceCutter(output_encrypted);

    if (ClearInputAfterEnc_Checkbox.IsChecked == true)
    {
        fourth_enc_input.Clear();
    }
}

//дешифровка
public void FirstDecBtnClick(object sender, RoutedEventArgs e)
{
    FreqDictGenerator(out Dictionary<char, int> freqdict, out List<char> freqsequence_
list);
    string input_encrypted = first_dec_input.Text;
    string output_string = DecryptAs1(freqsequence_list, input_encrypted);
    first_dec_output.Text = output_string;

    if (ClearInputAfterDec_Checkbox.IsChecked == true)
    {
        first_dec_input.Clear();
    }
}

```

```

public void SecondDecBtnClick(object sender, RoutedEventArgs e)
{
    List<char> positions_list = PositionsListGenerator();
    string input_encrypted = second_dec_input.Text;
    string output_string = DecryptAs2(positions_list, input_encrypted);
    second_dec_output.Text = output_string;

    if (ClearInputAfterDec_Checkbox.IsChecked == true)
    {
        second_dec_input.Clear();
    }
}

public void ThirdDecBtnClick(object sender, RoutedEventArgs e)
{
    try
    {
        char seed_char = Convert.ToChar(seedinput_dec.Text);    //введенный с
имвол конвертится из string в char
        string seed_string = SeedStringBuilder(seed_char);
        string output_string = "";
        string input_encrypted = third_dec_input.Text;
        int ascii_shift = Convert.ToInt32(ascii_shift_input.Text);
        output_string = DecryptAs3(seed_string, output_string, input_encrypted
, ascii_shift);
        third_dec_output.Text = output_string;

        if (ClearInputAfterDec_Checkbox.IsChecked == true)
        {
            third_dec_input.Clear();
        }
    }
    catch
    {
        ErrorMessage();
    }
}

public void FourthDecBtnClick(object sender, RoutedEventArgs e)
{
    List<char> positions_list_2 = PositionsList2Generator();
    string input_encrypted = fourth_dec_input.Text;
    string output_string = DecryptAs2(positions_list_2, input_encrypted);
    fourth_dec_output.Text = output_string;

    if (ClearInputAfterDec_Checkbox.IsChecked == true)
    {
        fourth_dec_input.Clear();
    }
}
}
}

```

Содержимое файла EncryptionMethods.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;

namespace Encryptor_with_GUI
{
    public partial class MainWindow : Window
    {
        #region First methods
        public static char[] EncryptAs1(List<char> freqsequence_list, string input_string, ref string output_encrypted)
        {
            char[] inputChar_arr = input_string.ToCharArray();
            for (int x = 0; x < input_string.Length; x++)
            {
                int index_of = freqsequence_list.IndexOf(inputChar_arr[x]);
                output_encrypted = output_encrypted + index_of + " ";
            }

            return inputChar_arr;
        }

        public static string DecryptAs1(List<char> freqsequence_list, string input_encrypted)
        {
            string input_index_int = input_encrypted.Replace(" ", string.Empty);
            int[] inputIndex_arr = new int[input_index_int.Length];

            try
            {
                inputIndex_arr = input_index_int.Split(' ').Select(int.Parse).ToArray();

                string output_string = "";
                for (int i = 0; i < inputIndex_arr.Length; i++)//повторяется, пока не закончатся символы в строке
                {
                    for (int j = 0; j < freqsequence_list.Count; j++)//перебирает все символы в словаре
                    {
                        if (inputIndex_arr[i] == j)
                        {
                            string output_str = Convert.ToString(freqsequence_list[j]);
                            output_string += output_str;
                        }
                    }
                }
                return output_string;
            }
            catch
            {
                return "Пожалуйста, заполните окно ввода";
            }
        }
    }
}

```

```

public static void FreqDictGenerator(out Dictionary<char, int> freqdict, out List<char>
> freqsequence_list)
{
    string freqsequence_string = AbbcccStringBuilder(); //построение частотной строки
    freqdict = SymbolRepeatCounter(freqsequence_string); //пересчет вхождений каждого с
имвола в частотную строку
    //перемещение ключей(символов) в порядке увеличения числа вхождений в массив(тепер
ь кол-во вхождений элемента == индекс этого же элемента в массиве)
    freqsequence_list = FreqSequenceListBuilder(freqdict);
}

public static List<char> FreqSequenceListBuilder(Dictionary<char, int> freqdict) //Dict
ionary<KEY, VALUE>
{
    List<char> freqsequence_list = new()
    {
        (char)10060 //заполнение нулевого индекса листа, подгонка индексации к частоте
вхождения
    };
    freqsequence_list = freqdict.Keys.ToList();
    return freqsequence_list;
}

public static Dictionary<char, int> SymbolRepeatCounter(string freqsequence_string)
{
    char alphabet_char;
    Dictionary<char, int> freqdict = new();
    foreach (char ch in freqsequence_string)
    {
        alphabet_char = ch;
        if (freqdict.ContainsKey(alphabet_char))
            freqdict[alphabet_char]++;
        else
            freqdict.Add(alphabet_char, 1);
    }
    return freqdict;
}

public static string AbbcccStringBuilder() //создание строки повторяющихся символов для
дальнейшего частотного анализа (способ "%1")
{
    string symbol_string = "";
    int repeat = 0;
    for (int symbol = 32; symbol <= 126; symbol++, repeat++) //с 32 по 126 символ(латин
ица+символы)
    {
        for (int i = repeat; i >= 0; i--)
        ) symbol_string = symbol_string.Insert(symbol_string.Length, Convert.ToString(Convert.
ToChar(symbol)));
    }
    //пропуск неподдерживаемых консолью символов
    for (int symbol = 1040; symbol <= 1103; symbol++, repeat++) //с 1040 по 1103 символ
(кириллица, оба регистра)
    {
        for (int i = repeat; i >= 0; i--)
        ) symbol_string = symbol_string.Insert(symbol_string.Length, Convert.ToString(Convert.
ToChar(symbol)));
    }

    return symbol_string;
}

```



```

#endregion

#region Second methods
public static char[] EncryptAs2(List<char> dictionary, string input_string, ref string
output_encrypted)
{
    char[] inputChar_arr = input_string.ToCharArray();

    for (int x = 0; x < input_string.Length; x++)
    {
        int index_of = dictionary.IndexOf(inputChar_arr[x]);
        output_encrypted = output_encrypted + index_of + " ";
    }
    return inputChar_arr;
}

public static string DecryptAs2(List<char> positions_list, string input_encrypted)
{
    input_encrypted = input_encrypted.Replace(" ", string.Empty);
    int[] inputIndex_arr = new int[input_encrypted.Length];
    try
    {
        inputIndex_arr = input_encrypted.Split(' ').Select(int.Parse).ToArray();
        string output_string = "";
        for (int i = 0; i < inputIndex_arr.Length; i++)//повторяется, пока не закончат
        ся символы в строке
        {
            for (int j = 0; j < positions_list.Count; j++)//перебирает все символы в с
            ловаре
            {
                if (inputIndex_arr[i] == j)
                {
                    string output_str = Convert.ToString(positions_list[j]);
                    output_string += output_str;
                }
            }
        }
        return output_string;
    }
    catch
    {
        return "Пожалуйста, заполните окно ввода";
    }
}

```

```

public static List<char> PositionsListGenerator()//создание листа символов для дальней
шего вывода индексов символов
{
    return new()
    {
        't','h','e','q','u','i','c','k','b','r','o','w','n','f','x','j','m','p','s','v',
        'l','a','z','y','d','g',
        'T','H','E','Q','U','I','C','K','B','R','O','W','N','F','X','J','M','P','S','V',
        'L','A','Z','Y','D','G',
        'с','ъ','е','ш','ь','ж','щ','ё','э','т','и','х','м','я','г','к','ф','р','а','н',
        'ц','у','з','б','л','о','д','в','ы','п','й','ч','ю',
        'С','Ъ','Е','Ш','Ь','Ж','Щ','Ё','Э','Т','И','Х','М','Я','Г','К','Ф','Р','А','Н',
        'Ц','У','З','Б','Л','О','Д','В','Ы','П','Й','Ч','Ю',
        ' ','!','?','_','<','>','[',']','{','}','+','=','','$','%',':','(',')','"','1','2','3','4','5',
        '6','7','8','9','0'
    };
}

#endregion

#region Third methods

public static string EncryptAs3(string input_string, int ascii_shift, string seed_stri
ng)
{
    byte[] ascii_bytes = Encoding.UTF8.GetBytes(input_string);//получение байтов симво
лов в кодировке ASCII (латиница)
    string output_encrypted = "";
    foreach (int ascii_bytes_of_element in ascii_bytes)
    {
        string encrypted_symbol = (ascii_bytes_of_element - ascii_shift + seed_string)
; //байтовое значение символа-сдвиг+добавление подстроки seed
        output_encrypted += encrypted_symbol;
    }

    return output_encrypted;
}

public static string DecryptAs3(string seed_string, string output_string, string input
_encrypted, int ascii_shift)
{
    string input_encrypted_without_seed = input_encrypted.Replace(seed_string, " ");
    //
    List<int> int_chars_list = new();
    string[] output_arr = input_encrypted_without_seed.Split(new char[] { ' ' }, Strin
gSplitOptions.RemoveEmptyEntries);
    foreach (string s in output_arr)
    {
        if (!int.TryParse(s, out int temp))
        {
            ErrorMessage();
        }

        else
        {
            int_chars_list.Add(temp);
        }
    }
}

```



```

public static string LastSpaceCutter(string output_encrypted)
{
    try
    {
        output_encrypted = output_encrypted.Remove(output_encrypted.LastIndexOf(
f(' '), 1));
        return output_encrypted;
    }
    catch
    {
        return "Пожалуйста, заполните окно ввода";
    }
}

public static void ErrorMessage()//вывод сообщения об ошибке для случая ввода
некорректного значения
{
    MessageBox.Show("Ошибка. Пожалуйста, заполните все поля");
}
}

```

					КП.22.09.02.07.603.12.ПЗ	Лист
						60
Изм		№ докум.	Подпись	Дата		

Интерфейс приложения:

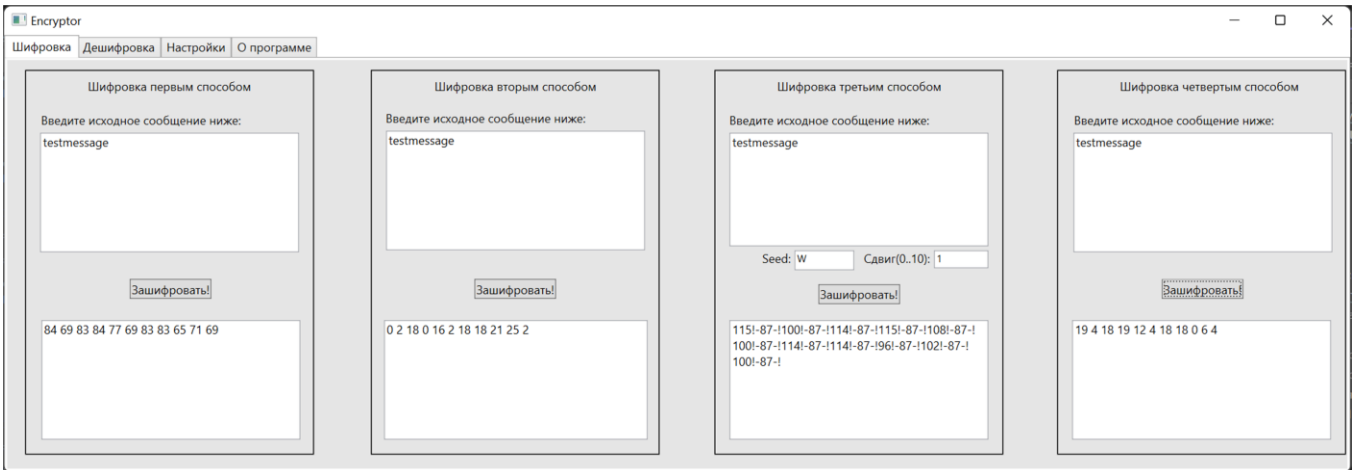


Рис. 19 – вкладка «Шифровка»

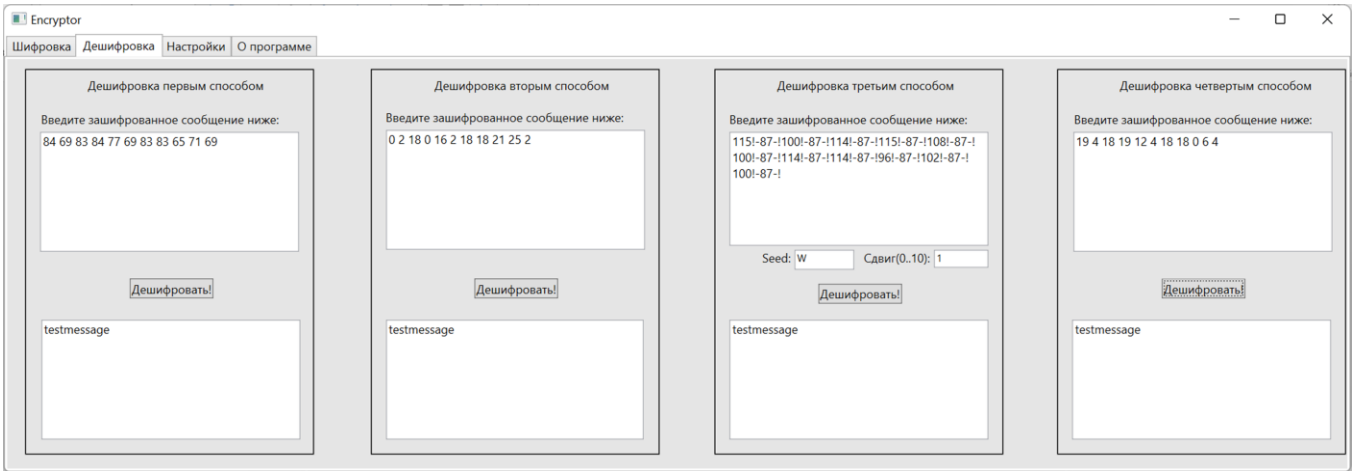


Рис. 20 – вкладка «Дешифровка»

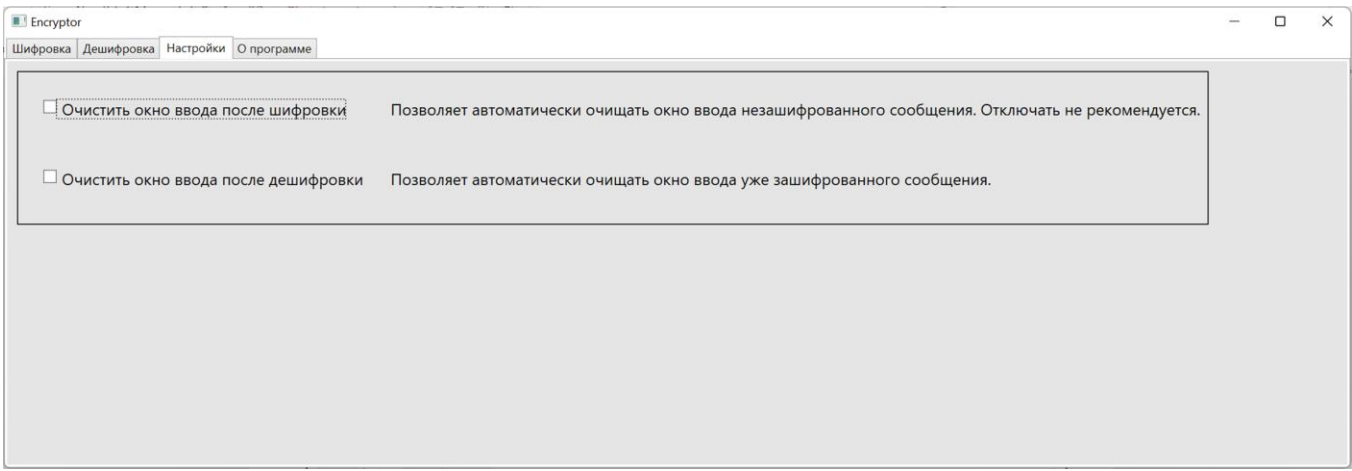


Рис. 21 – вкладка «Настройки»

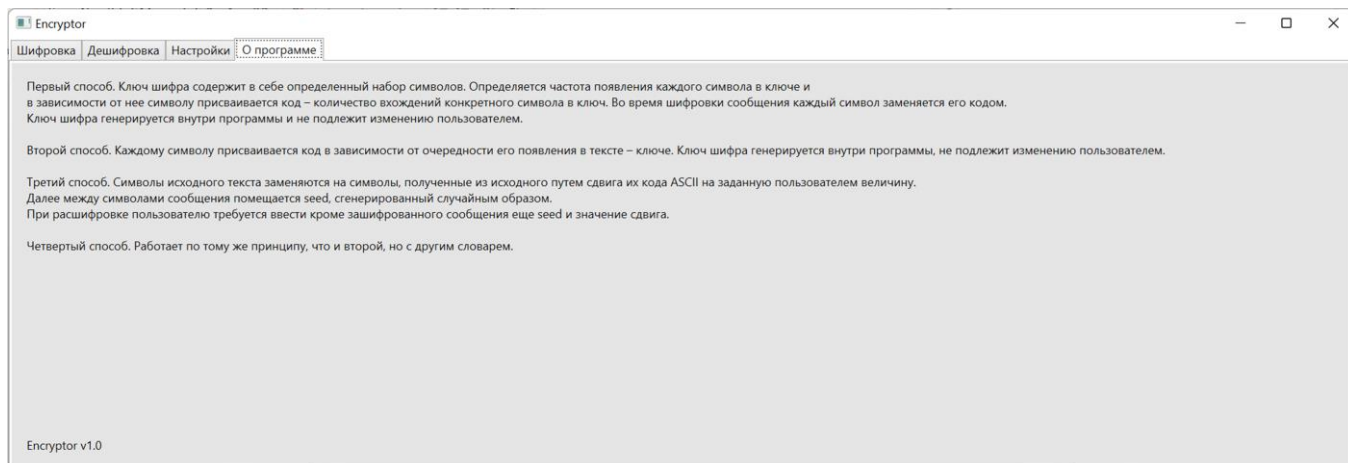


Рис. 22 – вкладка «О программе»

