Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| *институт* |
| Кафедра Прикладной математики и компьютерной безопасности |
| *кафедра* |

**ЛАБОРАТОРНАЯ РАБОТА №4**

|  |
| --- |
| **Квадрат Полибия** |
| *тема* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Руководитель | | |  |  |  | В.И.Вайнштейн |
|  | |  |  | *подпись, дата* |  | *инициалы, фамилия* |
| Студент | КИ15-01 №031508683 | |  |  |  | М.С.Димаксян |
|  | *номер группы, зачетной книжки* | |  | *подпись, дата* |  | *инициалы, фамилия* |

Красноярск 2019

**Задание:**

Реализовать Квадрат Полибия на любом языке программирования.

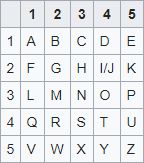
**Теория:**

**Шаг 1:**

К каждому языку отдельно составляется таблица шифрования с одинаковым (не обязательно) количеством пронумерованных строк и столбцов, параметры которой зависят от его мощности (количества букв в алфавите). Берутся два целых числа, произведение которых ближе всего к количеству букв в языке — получаем нужное число строк и столбцов. Затем вписываем в таблицу все буквы алфавита подряд — по одной в каждую клетку. При нехватке клеток можно вписать в одну две буквы (редко употребляющиеся или схожие по употреблению).

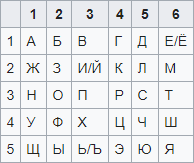
#### Латинский алфавит

В современном латинском алфавите 26 букв, следовательно таблица должна состоять из 5 строк и 5 столбцов, так как 25=5\*5 наиболее близкое к 26 число. При этом буквы I, J не различаются (J [отождествляется](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D0%B6%D0%B4%D0%B5%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) с буквой I), так как не хватает 1 ячейки:



#### Русский алфавит

Число букв в русском алфавите отличается от числа букв в греческом алфавите, поэтому размер таблицы выбирают другой (квадрат 6\*6=36, поскольку 36 наиболее близкое число к 33). Возможен другой вариант составления, предусматривающий объединение букв Е и Ё, И и Й, Ъ и Ь.



**Шаг 2:**

Существует несколько методов шифрования с помощью квадрата Полибия. Приведем два из них:

1. Для шифрования на квадрате находили букву текста и вставляли в шифровку нижнюю от неё в том же столбце. Если буква была в нижней строке, то брали верхнюю из того же столбца. В представленной ниже программе используется данный метод.
2. Сообщение преобразуется в координаты по квадрату Полибия, координаты записываются вертикально, затем их считывают по строкам. Далее координаты преобразуются в буквы по этому же квадрату.

В данной реализации используется второй метод.

**Исходный код:**

**Алгоритм зашифрования:**

private void f4\_ButtonEncrypt\_Click(object sender, EventArgs e)

{

string ALF\_RU = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ⁣⁤⁢"; //невидимые разделители

string ALF\_EN = "ABCDEFGHIKLMNOPQRSTUVWXYZ"; //J

string alf\_ru = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя­͏⁠"; //невидимые разделители

string alf\_en = "abcdefghiklmnopqrstuvwxyz"; //j

string Source = f4\_fieldOriginal.Text.Replace("J","I").Replace("j", "i");

//Создание матриц - квадратов Полибия

char[,] EN = new char[5, 5];

int k = 0;

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

EN[i, j] = ALF\_EN[k];

k++;

}

}

k = 0;

char[,] en = new char[5, 5];

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

en[i, j] = alf\_en[k];

k++;

}

}

k = 0;

char[,] RU = new char[6, 6];

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

RU[i, j] = ALF\_RU[k];

k++;

}

}

k = 0;

char[,] ru = new char[6, 6];

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

ru[i, j] = alf\_ru[k];

k++;

}

}

//Создадим одномерный массив координат для каждого квадрата

int[,] koord\_EN = new int[2, Source.Length];

int ind\_EN = 0;

int[,] koord\_en = new int[2, Source.Length];

int ind\_en = 0;

int[,] koord\_RU = new int[2, Source.Length];

int ind\_RU = 0;

int[,] koord\_ru = new int[2, Source.Length];

int ind\_ru = 0;

//Заполним созданные массивы

k = 0;

bool flag = false;

for (k = 0; k < Source.Length; k++)

{

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == EN[i, j])

{

koord\_EN[0, ind\_EN] = j;

koord\_EN[1, ind\_EN] = i;

ind\_EN++;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == en[i, j])

{

koord\_en[0, ind\_en] = j;

koord\_en[1, ind\_en] = i;

ind\_en++;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == RU[i, j])

{

koord\_RU[0, ind\_RU] = j;

koord\_RU[1, ind\_RU] = i;

ind\_RU++;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == ru[i, j])

{

koord\_ru[0, ind\_ru] = j;

koord\_ru[1, ind\_ru] = i;

ind\_ru++;

flag = true;

break;

}

}

if (flag)

break;

}

flag = false;

}

//Перепишем координаты в 1 строку

k = 0;

int[] coord\_str\_EN = new int[ind\_EN \* 2];

for (int i = 0; i < 2; i++)

for (int j = 0; j < ind\_EN; j++)

{

coord\_str\_EN[k] = koord\_EN[i, j];

k++;

}

k = 0;

int[] coord\_str\_en = new int[ind\_en \* 2];

for (int i = 0; i < 2; i++)

for (int j = 0; j < ind\_en; j++)

{

coord\_str\_en[k] = koord\_en[i, j];

k++;

}

k = 0;

int[] coord\_str\_RU = new int[ind\_RU \* 2];

for (int i = 0; i < 2; i++)

for (int j = 0; j < ind\_RU; j++)

{

coord\_str\_RU[k] = koord\_RU[i, j];

k++;

}

k = 0;

int[] coord\_str\_ru = new int[ind\_ru \* 2];

for (int i = 0; i < 2; i++)

for (int j = 0; j < ind\_ru; j++)

{

coord\_str\_ru[k] = koord\_ru[i, j];

k++;

}

//Запишем шифртекст, используя координаты из строки

ind\_EN = 0;

ind\_en = 0;

ind\_RU = 0;

ind\_ru = 0;

string result = "";

for (k = 0; k < Source.Length; k++)

{

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == EN[i, j])

{

result += EN[coord\_str\_EN[ind\_EN + 1], coord\_str\_EN[ind\_EN]];

ind\_EN += 2;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == en[i, j])

{

result += en[coord\_str\_en[ind\_en + 1], coord\_str\_en[ind\_en]];

ind\_en += 2;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == RU[i, j])

{

result += RU[coord\_str\_RU[ind\_RU + 1], coord\_str\_RU[ind\_RU]];

ind\_RU += 2;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == ru[i, j])

{

result += ru[coord\_str\_ru[ind\_ru + 1], coord\_str\_ru[ind\_ru]];

ind\_ru += 2;

flag = true;

break;

}

}

if (flag)

break;

}

if (!flag)

result += Source[k];

flag = false;

}

f4\_fieldCiphertext.Text = result;

}

**Алгоритм расшифрования:**

private void f4\_ButtonDecrypt\_Click(object sender, EventArgs e)

{

string ALF\_RU = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ⁣⁤⁢"; //невидимые разделители

string ALF\_EN = "ABCDEFGHIKLMNOPQRSTUVWXYZ"; //J

string alf\_ru = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя­͏⁠"; //невидимые разделители

string alf\_en = "abcdefghiklmnopqrstuvwxyz"; //j

string Source = f4\_fieldCiphertext.Text.Replace("J", "I").Replace("j", "i");

string result = "";

//Создание матриц - квадратов Полибия

char[,] EN = new char[5, 5];

int k = 0;

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

EN[i, j] = ALF\_EN[k];

k++;

}

}

k = 0;

char[,] en = new char[5, 5];

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

en[i, j] = alf\_en[k];

k++;

}

}

k = 0;

char[,] RU = new char[6, 6];

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

RU[i, j] = ALF\_RU[k];

k++;

}

}

k = 0;

char[,] ru = new char[6, 6];

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

ru[i, j] = alf\_ru[k];

k++;

}

}

//Создадим одномерный массив координат для каждого квадрата

int[] coord\_str\_EN = new int[Source.Length \* 2];

int ind\_EN = 0;

int[] coord\_str\_en = new int[Source.Length \* 2];

int ind\_en = 0;

int[] coord\_str\_RU = new int[Source.Length \* 2];

int ind\_RU = 0;

int[] coord\_str\_ru = new int[Source.Length \* 2];

int ind\_ru = 0;

k = 0;

bool flag = false;

for (k = 0; k < Source.Length; k++)

{

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == EN[i, j])

{

coord\_str\_EN[ind\_EN] = j;

coord\_str\_EN[ind\_EN + 1] = i;

ind\_EN += 2;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == en[i, j])

{

coord\_str\_en[ind\_en] = j;

coord\_str\_en[ind\_en + 1] = i;

ind\_en += 2;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == RU[i, j])

{

coord\_str\_RU[ind\_RU] = j;

coord\_str\_RU[ind\_RU + 1] = i;

ind\_RU += 2;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == ru[i, j])

{

coord\_str\_ru[ind\_ru] = j;

coord\_str\_ru[ind\_ru + 1] = i;

ind\_ru += 2;

flag = true;

break;

}

}

if (flag)

break;

}

flag = false;

}

//Перепишем координаты букв в две строки, содержащие координаты по строке и столбцу

int[,] koord\_EN = new int[2, ind\_EN / 2];

k = 0;

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < ind\_EN / 2; j++)

{

koord\_EN[i, j] = coord\_str\_EN[k];

k++;

}

}

int[,] koord\_en = new int[2, ind\_en / 2];

k = 0;

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < ind\_en / 2; j++)

{

koord\_en[i, j] = coord\_str\_en[k];

k++;

}

}

int[,] koord\_RU = new int[2, ind\_RU / 2];

k = 0;

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < ind\_RU / 2; j++)

{

koord\_RU[i, j] = coord\_str\_RU[k];

k++;

}

}

int[,] koord\_ru = new int[2, ind\_ru / 2];

k = 0;

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < ind\_ru / 2; j++)

{

koord\_ru[i, j] = coord\_str\_ru[k];

k++;

}

}

//Расшифруем текст, используя записанные координаты

ind\_EN = 0;

ind\_en = 0;

ind\_RU = 0;

ind\_ru = 0;

for (k = 0; k < Source.Length; k++)

{

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == EN[i, j])

{

result += EN[koord\_EN[1, ind\_EN], koord\_EN[0, ind\_EN]];

ind\_EN++;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 5; i++)

{

for (int j = 0; j < 5; j++)

{

if (Source[k] == en[i, j])

{

result += en[koord\_en[1, ind\_en], koord\_en[0, ind\_en]];

ind\_en++;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == RU[i, j])

{

result += RU[koord\_RU[1, ind\_RU], koord\_RU[0, ind\_RU]];

ind\_RU++;

flag = true;

break;

}

}

if (flag)

break;

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 6; j++)

{

if (Source[k] == ru[i, j])

{

result += ru[koord\_ru[1, ind\_ru], koord\_ru[0, ind\_ru]];

ind\_ru++;

flag = true;

break;

}

}

if (flag)

break;

}

if (!flag)

result += Source[k];

flag = false;

}

f4\_fieldOriginal.Text = result;

}

**Результат работы программы:**

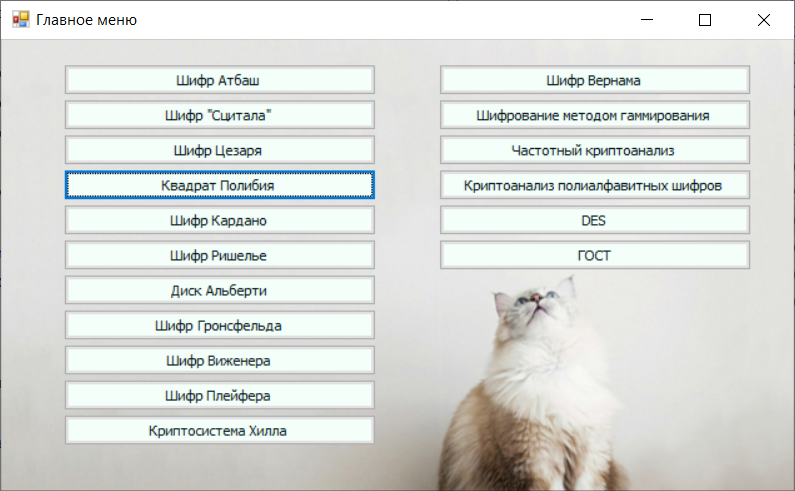
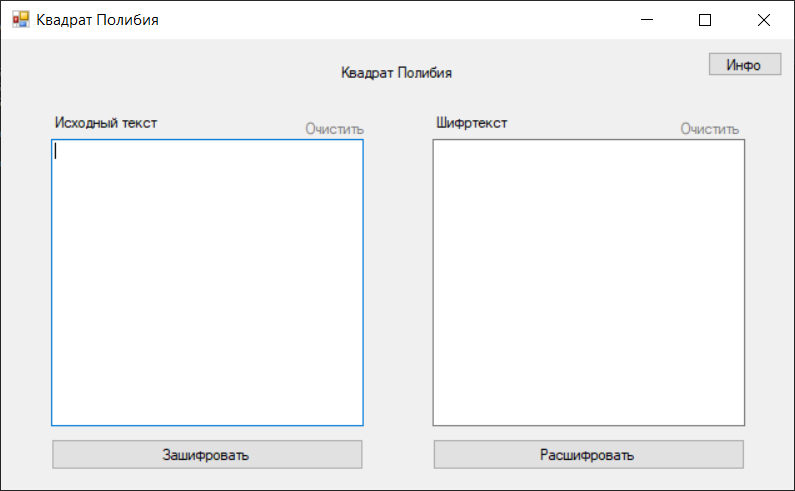


Рисунок 1. Главное меню программы.

Рисунок 2. Окно квадрата Полибия.

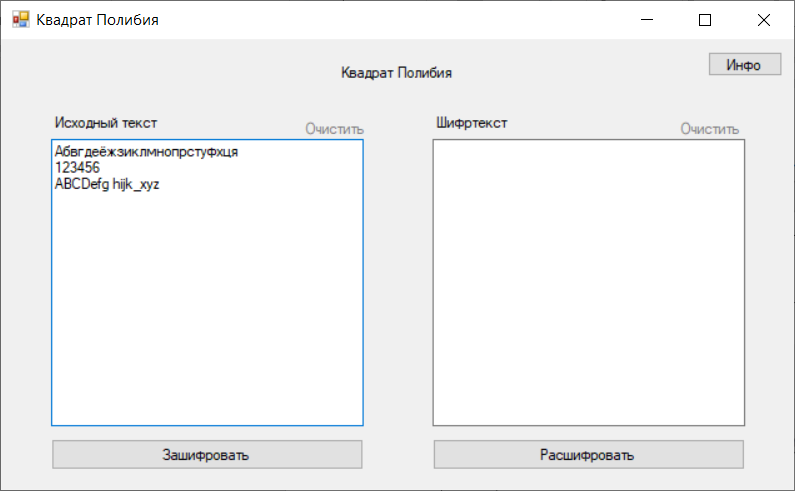


Рисунок 3. Исходный текст.

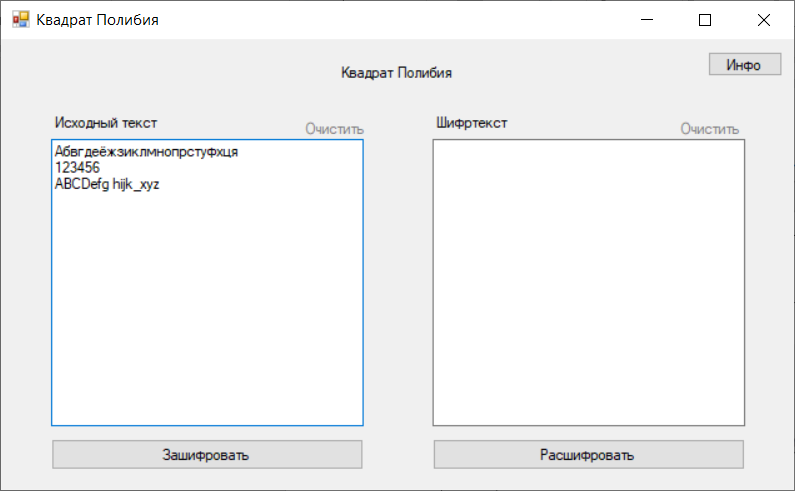


Рисунок 4. Шифртекст (результат зашифрования).

Очистим поле Исходный текст и расшифруем сообщение.

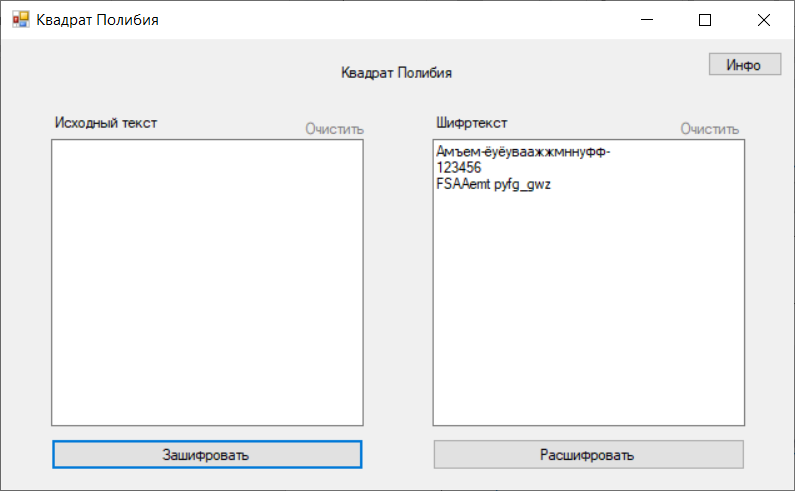


Рисунок 5. Очистка поля Исходный текст.

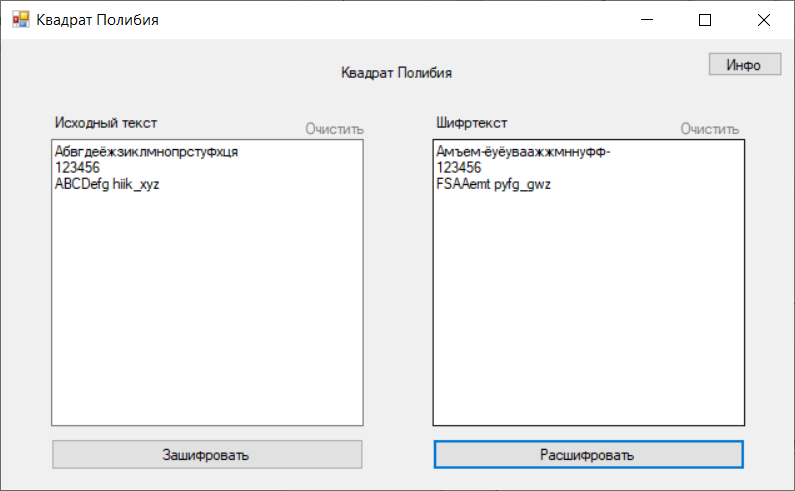


Рисунок 6. Исходный текст (результат расшифрования).

**Вывод:**

В ходе выполнения лабораторной работы я ознакомилась с алгоритмом шифрования при помощи Квадрата Полибия, реализовала на практике программу по зашифрованию и расшифрованию данным шифром на языке C#. Данные навыки я могу применить при реализации других шифров и при дальнейшем использовании данного шифра.