Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №3**

«Основы криптоанализа шифров полиалфавитной подстановки»

тема

Преподаватель \_\_\_\_\_\_\_\_\_ А.А. Сидарас

подпись, дата инициалы, фамилия

Студент КИ18-17/1б 031830504 \_\_\_\_\_\_\_\_\_ Е.В. Железкин

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2020

Содержание

[Содержание 2](#_Toc18956750)

[1 Цель работы 3](#_Toc18956751)

[2 Задача работы 3](#_Toc18956752)

[3 Ход работы 3](#_Toc18956753)

[4 Вывод 10](#_Toc18956754)

# Цель работы

• введение в криптосистемы полиалфавитной подстановки на примере шифра Виженера;

• изучение методов криптоанализа шифров полиалфавитной подстановки на примере шифра Виженера.

# Задача работы

Используя любой из способов криптоанализа (рассмотренных ранее) шифров полиалфавитной подстановки расшифруйте текст сообщения (согласно вашему персональному варианту), а также определите секретный ключ, которым оно было зашифровано. Известные сведения о шифрограмме следующие:

• для зашифрования сообщения, которое представляет собой отрывок из литературного произведения, использовался шифр Виженера, с алфавитом, представленным в таблице;

• знаки препинания, а также знак пробела не входят в алфавит шифрования и, соответственно, не является частью шифротекста, т.е. шифротекст с расставленными знаками препинания и пробелами эквивалентен шифротексту без них;

• весь шифротекст представляет собой одну строку;

• в качестве ключа шифрования использовалось одно слово (из букв алфавита), длина которого не более 10 символов. Для осуществления статистического криптоанализа необходимо разработать программу для дешифрования со следующими возможностями:

• автоматическое приближённое нахождение периода шифрования (числа μ) с помощью теста Касиски;

• автоматическая проверка правильности выбора μ с помощью нахождения индексов соответствия (для всех подгрупп шифротекста);

• в случаях, когда тест Касиски не удаётся применить или значение μ, найденное с его помощью, не подтверждается после определения индексов соответствия, μ определяется подбором (такой случай возможен в случаях, когда длина ключа более 7 знаков и/или зашифрованный текст слишком короткий);

• автоматическое вычисление всех потенциально возможных вариантов ключа шифрования, и вывод их пользователю для принятия решения;

• расшифровка шифротекста, по выбранному пользователем ключу шифрования.

**Вариант 7.**

# Ход работы

1. Для начала необходимо заполнить графу шифротекста одним из двух способов (из файла или вручную).

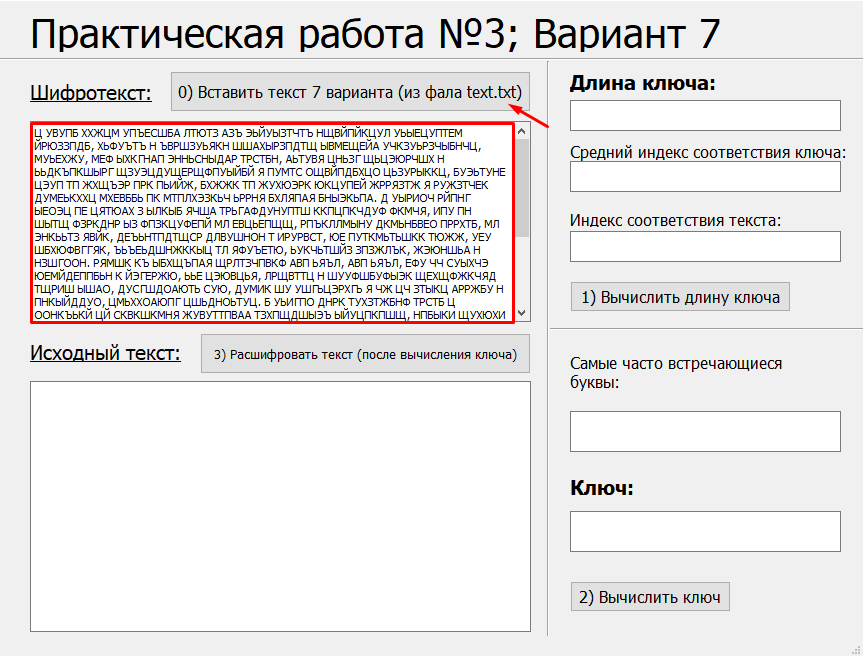


Рисунок 1 – руководство пользователя 1

1. Далее необходимо вычислить длину ключа и удостовериться, что индекс соответствия текста и ключа примерно идентичны (для табличных ключей <8 знаков).

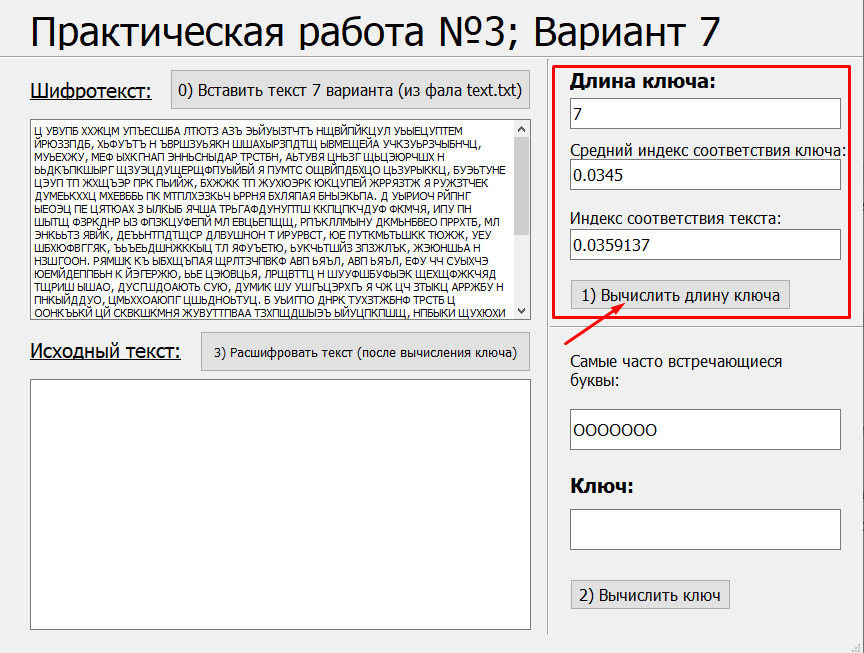


Рисунок 2 – руководство пользователя 2

1. Теперь необходимо нажать на кнопку вычисления ключа и подправить его, при необходимости.

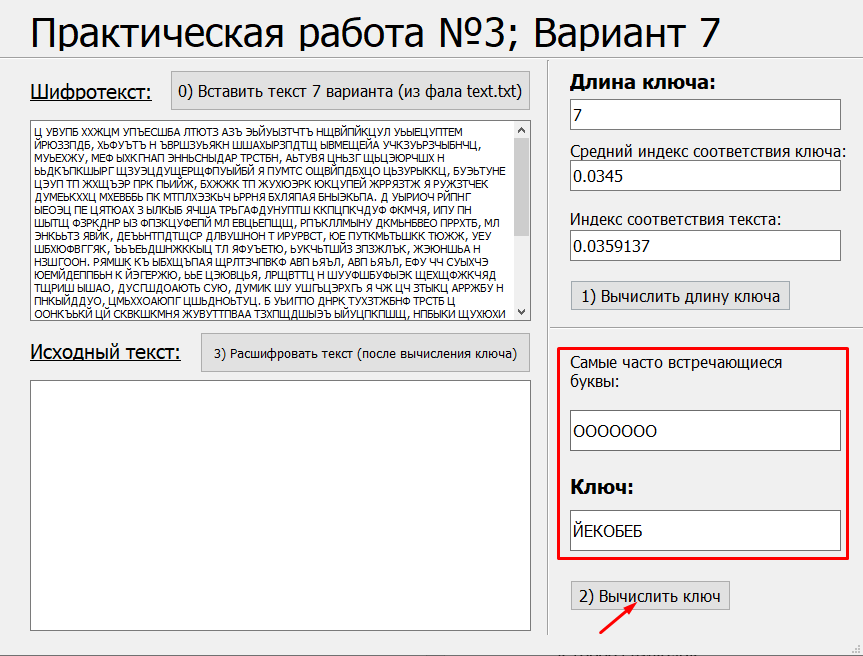


Рисунок 3 – руководство пользователя 3

1. После правки ключа, нажатием на соответствующую кнопку текст будет расшифрован и отображён во втором текстовом поле.

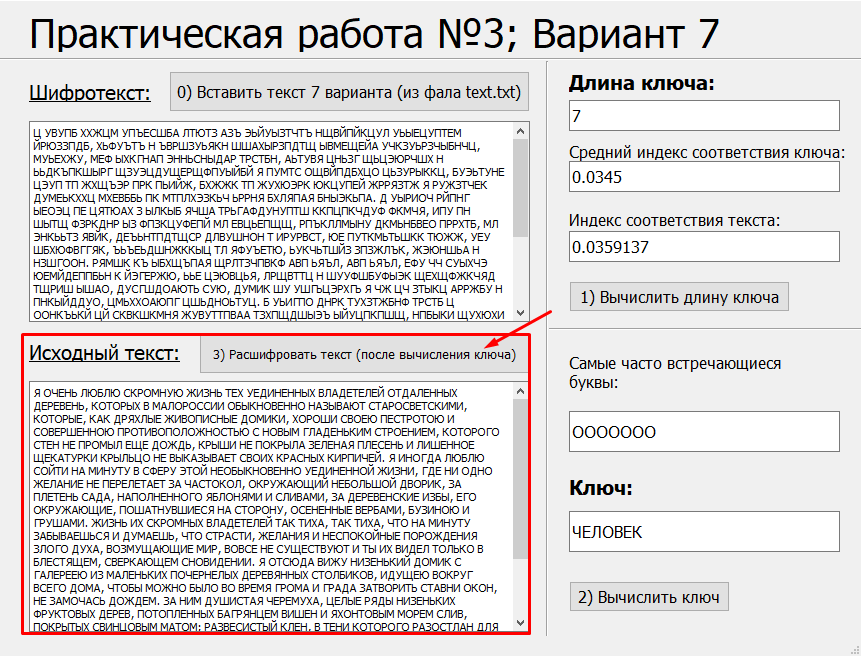


Рисунок 4 – руководство пользователя 4

Листинг 1 – файл main.py

*# coding=utf8*from PyQt5 import QtCore, QtGui, QtWidgets, uic  
from PyQt5.QtWidgets import QMainWindow  
import sys  
from funcs import encoder  
  
  
class MainForm(QMainWindow):  
 pushButtonCalcKeyLength: QtWidgets.QPushButton  
 pushButtonEnterText: QtWidgets.QPushButton  
 pushButtonEncode: QtWidgets.QPushButton  
 textEdit: QtWidgets.QTextEdit  
 keyLengthLine: QtWidgets.QLineEdit  
 keyLine: QtWidgets.QLineEdit  
 keyIndexLine: QtWidgets.QLineEdit  
 textIndexLine: QtWidgets.QLineEdit  
 encodedText: QtWidgets.QTextEdit  
 pushButtonCalcKey: QtWidgets.QPushButton  
 mostCountedLine: QtWidgets.QLineEdit  
  
 def \_\_init\_\_(self):  
 super(MainForm, self).\_\_init\_\_()  
 self.ui = uic.loadUi(**'form.ui'**, self)  
 self.temp\_enc = encoder()  
 self.enterTextFromFile()  
 self.pushButtonCalcKeyLength.clicked.connect(self.update\_key\_length)  
 self.pushButtonEnterText.clicked.connect(self.enterTextFromFile)  
 self.keyLine.textChanged.connect(self.check\_key)  
 self.keyLine.textChanged.connect(self.on\_key\_update)  
 self.mostCountedLine.textChanged.connect(self.check\_most\_counted)  
 self.pushButtonEncode.clicked.connect(self.do\_text\_encode)  
 self.pushButtonCalcKey.clicked.connect(self.calculate\_key)  
  
 def update\_key\_length(self):  
 self.temp\_enc.text = self.temp\_enc.text\_translator(self.textEdit.toPlainText())  
 self.temp\_enc.main\_func()  
 self.mostCountedLine.setText(**''**.join(**'О'** for \_ in range(0, self.temp\_enc.key\_length)))  
 self.keyLengthLine.setText(str(self.temp\_enc.key\_length))  
 self.keyIndexLine.setText((lambda x: self.temp\_enc.conformity\_key\_index[0] if (x > 7) else str(  
 self.temp\_enc.conformity\_key\_index[x]))(self.temp\_enc.key\_length))  
 self.textIndexLine.setText(str(**'%.7f'** % self.temp\_enc.text\_index))  
  
 def calculate\_key(self):  
 self.temp\_enc.encode\_key(self.mostCountedLine.text())  
 self.keyLine.setText(self.temp\_enc.key)  
  
 def on\_key\_update(self):  
 self.temp\_enc.key = self.keyLine.text()  
  
 def enterTextFromFile(self):  
 file = open(**'text.txt'**, **'r'**)  
 text = file.read()  
 file.close()  
 self.textEdit.setText(text)  
 self.temp\_enc.text = self.temp\_enc.text\_translator(text)  
  
 def check\_key(self):  
 key = self.keyLine.text()  
  
 if len(key) > self.temp\_enc.key\_length:  
 self.keyLine.setText(self.keyLine.text()[0:7])  
 return  
 for i in key:  
 if i not in self.temp\_enc.alphabet:  
 key = key.replace(i, **''**)  
 if len(key) != self.temp\_enc.key\_length:  
 self.keyLine.setText(key)  
  
 def check\_most\_counted(self):  
 text = self.mostCountedLine.text()  
  
 if len(text) > self.temp\_enc.key\_length:  
 self.mostCountedLine.setText(self.mostCountedLine.text()[0:7])  
 return  
 for i in text:  
 if i not in self.temp\_enc.alphabet:  
 text = text.replace(i, **''**)  
 if len(text) != self.temp\_enc.key\_length:  
 self.mostCountedLine.setText(text)  
  
 def do\_text\_encode(self):  
 if self.textEdit.toPlainText() != **''** and \  
 self.temp\_enc.key\_length != -1 and \  
 self.temp\_enc.key != **''** and \  
 self.temp\_enc.key\_length == len(self.temp\_enc.key):  
 self.encodedText.setText(self.temp\_enc.text\_encoder())  
  
  
if \_\_name\_\_ == **'\_\_main\_\_'**:  
 app = QtWidgets.QApplication([])  
 start\_window = MainForm()  
 start\_window.show()  
 sys.exit(app.exec())

Листинг 2 – файл funcs.py

class encoder:  
 alphabet = **'АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ'** conformity\_key\_index = {0: **'Длина ключа больше "7" !'**, 1: 0.058, 2: 0.046, 3: 0.04,  
 4: 0.0375, 5: 0.036, 6: 0.0352, 7: 0.0345}  
  
 @property  
 def text(self):  
 return self.\_text  
  
 @text.setter  
 def text(self, value):  
 self.\_text = value  
  
 @property  
 def trigrams\_dict(self):  
 return self.\_trigrams\_dict  
  
 @property  
 def distance\_dict(self):  
 return self.\_distance\_dict  
  
 @property  
 def key\_length(self):  
 return self.\_key\_length  
  
 @key\_length.setter  
 def key\_length(self, value):  
 self.\_key\_length = value  
  
 @property  
 def key(self):  
 return self.\_key  
  
 @key.setter  
 def key(self, value):  
 self.\_key = value  
  
 @property  
 def source\_text(self):  
 return self.\_source\_text  
  
 @source\_text.setter  
 def source\_text(self, value):  
 self.\_source\_text = value  
  
 @property  
 def text\_index(self):  
 return self.\_text\_index  
  
 @text\_index.setter  
 def text\_index(self, value):  
 self.\_text\_index = value  
  
 def \_\_init\_\_(self):  
 self.\_key = **''** self.\_text = **''** self.\_text\_index = 0  
 self.\_trigrams\_dict = {}  
 self.\_distance\_dict = {}  
 self.\_key\_length = -1  
 self.\_most\_counted\_letter = []  
 self.\_source\_text = **''** pass  
  
 def text\_translator(self, input\_text):  
 result = []  
  
 self.source\_text = input\_text  
  
 for i in input\_text:  
 if i in self.alphabet:  
 result.append(i)  
 return **''**.join(result)  
  
 def main\_func(self):  
  
 self.find\_double\_and\_more\_trigrams()  
  
 self.drop\_excess\_trigrams()  
  
 self.gen\_distance\_dict()  
  
 self.key\_length = self.calculate\_key\_length()  
  
 self.text\_index = self.calculate\_index(self.letters\_count(), self.text)  
  
 self.reset()  
  
 return  
  
 def text\_encoder(self):  
  
 encoded\_text = **''** for i in range(0, len(self.text)):  
 encoded\_text += chr((ord(self.text[i]) - ord(self.key[i % self.key\_length])) % 32 + ord(**'А'**))  
  
 return self.return\_punctuation(encoded\_text)  
  
 def return\_punctuation(self, text\_to\_update):  
  
 result = **''** text\_counter = 0  
  
 for i in range(0, len(self.source\_text)):  
 if self.source\_text[i] in self.alphabet:  
 result += text\_to\_update[text\_counter]  
 text\_counter += 1  
 else:  
 result += self.source\_text[i]  
  
 return result  
  
 def reset(self):  
 self.\_trigrams\_dict = {}  
 self.\_distance\_dict = {}  
  
 def encode\_key(self, support\_str):  
  
 letters\_counter = []  
 for i in range(0, self.key\_length):  
 letters\_counter.append({})  
  
 for i in range(0, len(self.text)):  
 if self.text[i] not in letters\_counter[i % self.key\_length]:  
 letters\_counter[i % self.key\_length][self.text[i]] = 0  
 letters\_counter[i % self.key\_length][self.text[i]] += 1  
  
 temp\_key = **''** self.key = **''** for i in range(0, self.key\_length):  
 temp\_key += list(filter(lambda x: letters\_counter[i][x[0]] == max(letters\_counter[i].values()), letters\_counter[i].items()))[0][0]  
  
 for i in range(0, self.key\_length):  
 self.key += chr(abs(ord(temp\_key[i]) - ord(support\_str[i])) % 32 + ord(**'А'**))  
  
 def calculate\_key\_length(self):  
  
 keys\_dict = {}  
  
 for i in self.distance\_dict:  
 for j in range(0, len(self.distance\_dict[i]) - 1):  
 self.distance\_dict[i][j + 1] = self.gcd(self.distance\_dict[i][j], self.distance\_dict[i][j + 1])  
 temp\_key = min(self.distance\_dict[i])  
 if temp\_key not in keys\_dict:  
 keys\_dict[temp\_key] = 0  
 keys\_dict[min(self.distance\_dict[i])] += 1  
  
 for i in list(keys\_dict):  
 if i < 3:  
 keys\_dict.pop(i)  
  
 return list(filter(lambda x: keys\_dict[x[0]] == max(keys\_dict.values()), keys\_dict.items()))[0][0]  
  
 def gen\_distance\_dict(self):  
 for i in self.trigrams\_dict:  
 for j in range(0, len(self.trigrams\_dict[i]) - 1):  
 if i not in self.distance\_dict:  
 self.distance\_dict[i] = []  
 self.distance\_dict[i].append(self.trigrams\_dict[i][j + 1] - self.trigrams\_dict[i][j])  
  
 def find\_double\_and\_more\_trigrams(self):  
 for i in range(0, len(self.text) - 2):  
 if self.text[i + 2] not in self.alphabet or self.text[i + 1] not in self.alphabet or \  
 self.text[i] not in self.alphabet:  
 continue  
 pos = self.text.find(self.text[i:i + 3], i + 1)  
 while pos != -1:  
 if self.text[i:i + 3] not in self.trigrams\_dict:  
 self.trigrams\_dict[self.text[i:i + 3]] = []  
 if i not in self.trigrams\_dict[self.text[i:i + 3]]:  
 self.trigrams\_dict[self.text[i:i + 3]].append(i)  
 if pos not in self.trigrams\_dict[self.text[i:i + 3]]:  
 self.trigrams\_dict[self.text[i:i + 3]].append(pos)  
  
 pos = self.text.find(self.text[i:i + 3], pos + 1)  
  
 def drop\_excess\_trigrams(self):  
 for i in list(self.trigrams\_dict):  
 if len(self.trigrams\_dict[i]) < 3:  
 self.trigrams\_dict.pop(i)  
  
 def letters\_count(self):  
 letters = {}  
 for i in self.alphabet:  
 for j in self.text:  
 if i == j:  
 if i not in letters:  
 letters[i] = 1  
 continue  
 letters[i] += 1  
 return letters  
  
 @staticmethod  
 def calculate\_index(dict\_with\_letters: dict, text: str):  
 letters\_sum = 0  
 for i in dict\_with\_letters:  
 letters\_sum += dict\_with\_letters[i] \* (dict\_with\_letters[i] - 1)  
 return letters\_sum / (len(text) \* (len(text) - 1))  
  
 @staticmethod  
 def gcd(a, b):  
 while a != b:  
 if a > b:  
 a = a - b  
 else:  
 b = b - a  
 return a

Листинг 3 – файл form.ui

*<?*xml version="1.0" encoding="UTF-8"*?>*<ui version="4.0">  
 <class>MainWindow</class>  
 <widget class="QMainWindow" name="MainWindow">  
 <property name="geometry">  
 <rect>  
 <x>0</x>  
 <y>0</y>  
 <width>863</width>  
 <height>655</height>  
 </rect>  
 </property>  
 <property name="windowTitle">  
 <string>MainWindow</string>  
 </property>  
 <widget class="QWidget" name="centralwidget">  
 <widget class="QPushButton" name="pushButtonCalcKeyLength">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>280</y>  
 <width>221</width>  
 <height>31</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>1) Вычислить длину ключа</string>  
 </property>  
 </widget>  
 <widget class="QPushButton" name="pushButtonEnterText">  
 <property name="geometry">  
 <rect>  
 <x>170</x>  
 <y>70</y>  
 <width>361</width>  
 <height>41</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>0) Вставить текст 7 варианта (из фала text.txt)</string>  
 </property>  
 </widget>  
 <widget class="QTextEdit" name="textEdit">  
 <property name="geometry">  
 <rect>  
 <x>30</x>  
 <y>120</y>  
 <width>501</width>  
 <height>201</height>  
 </rect>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label">  
 <property name="geometry">  
 <rect>  
 <x>30</x>  
 <y>10</y>  
 <width>741</width>  
 <height>41</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>30</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>Практическая работа №3; Вариант 7</string>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label\_2">  
 <property name="geometry">  
 <rect>  
 <x>30</x>  
 <y>80</y>  
 <width>161</width>  
 <height>21</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>15</pointsize>  
 <weight>50</weight>  
 <italic>false</italic>  
 <bold>false</bold>  
 <underline>true</underline>  
 <strikeout>false</strikeout>  
 </font>  
 </property>  
 <property name="text">  
 <string>Шифротекст:</string>  
 </property>  
 </widget>  
 <widget class="Line" name="line">  
 <property name="geometry">  
 <rect>  
 <x>0</x>  
 <y>50</y>  
 <width>871</width>  
 <height>16</height>  
 </rect>  
 </property>  
 <property name="orientation">  
 <enum>Qt::Horizontal</enum>  
 </property>  
 </widget>  
 <widget class="Line" name="line\_2">  
 <property name="geometry">  
 <rect>  
 <x>540</x>  
 <y>60</y>  
 <width>16</width>  
 <height>611</height>  
 </rect>  
 </property>  
 <property name="orientation">  
 <enum>Qt::Vertical</enum>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label\_3">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>70</y>  
 <width>151</width>  
 <height>21</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>15</pointsize>  
 <weight>75</weight>  
 <bold>true</bold>  
 </font>  
 </property>  
 <property name="text">  
 <string>Длина ключа:</string>  
 </property>  
 </widget>  
 <widget class="QLineEdit" name="keyLengthLine">  
 <property name="enabled">  
 <bool>true</bool>  
 </property>  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>99</y>  
 <width>271</width>  
 <height>31</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="readOnly">  
 <bool>true</bool>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label\_4">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>140</y>  
 <width>281</width>  
 <height>21</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>Средний индекс соответствия ключа:</string>  
 </property>  
 </widget>  
 <widget class="QLineEdit" name="keyIndexLine">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>160</y>  
 <width>271</width>  
 <height>31</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="readOnly">  
 <bool>true</bool>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label\_5">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>210</y>  
 <width>221</width>  
 <height>16</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>Индекс соответствия текста:</string>  
 </property>  
 </widget>  
 <widget class="QLineEdit" name="textIndexLine">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>230</y>  
 <width>271</width>  
 <height>31</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="readOnly">  
 <bool>true</bool>  
 </property>  
 </widget>  
 <widget class="Line" name="line\_3">  
 <property name="geometry">  
 <rect>  
 <x>550</x>  
 <y>320</y>  
 <width>341</width>  
 <height>16</height>  
 </rect>  
 </property>  
 <property name="orientation">  
 <enum>Qt::Horizontal</enum>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label\_6">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>470</y>  
 <width>71</width>  
 <height>31</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>15</pointsize>  
 <weight>75</weight>  
 <bold>true</bold>  
 </font>  
 </property>  
 <property name="text">  
 <string>Ключ:</string>  
 </property>  
 </widget>  
 <widget class="QLineEdit" name="keyLine">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>510</y>  
 <width>271</width>  
 <height>41</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 </widget>  
 <widget class="QTextEdit" name="encodedText">  
 <property name="geometry">  
 <rect>  
 <x>30</x>  
 <y>380</y>  
 <width>501</width>  
 <height>251</height>  
 </rect>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label\_7">  
 <property name="geometry">  
 <rect>  
 <x>30</x>  
 <y>340</y>  
 <width>231</width>  
 <height>21</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>15</pointsize>  
 <underline>true</underline>  
 <strikeout>false</strikeout>  
 </font>  
 </property>  
 <property name="text">  
 <string>Исходный текст:</string>  
 </property>  
 </widget>  
 <widget class="QPushButton" name="pushButtonEncode">  
 <property name="geometry">  
 <rect>  
 <x>200</x>  
 <y>332</y>  
 <width>331</width>  
 <height>41</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>10</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>3) Расшифровать текст (после вычисления ключа)</string>  
 </property>  
 </widget>  
 <widget class="QPushButton" name="pushButtonCalcKey">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>580</y>  
 <width>161</width>  
 <height>31</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>2) Вычислить ключ</string>  
 </property>  
 </widget>  
 <widget class="QLabel" name="label\_8">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>350</y>  
 <width>241</width>  
 <height>41</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 <property name="text">  
 <string>Самые часто встречающиеся  
буквы:</string>  
 </property>  
 </widget>  
 <widget class="QLineEdit" name="mostCountedLine">  
 <property name="geometry">  
 <rect>  
 <x>570</x>  
 <y>410</y>  
 <width>271</width>  
 <height>41</height>  
 </rect>  
 </property>  
 <property name="font">  
 <font>  
 <pointsize>12</pointsize>  
 </font>  
 </property>  
 </widget>  
 </widget>  
 <widget class="QStatusBar" name="statusbar"/>  
 </widget>  
 <resources/>  
 <connections/>  
</ui>

1. Примеры работы программы с другими вариантами:

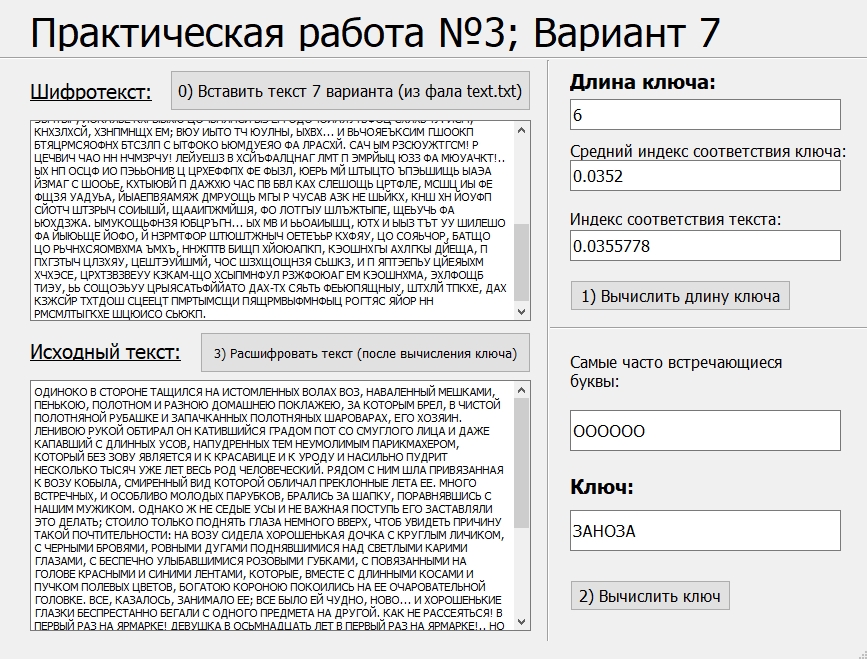


Рисунок 5 – вариант 2

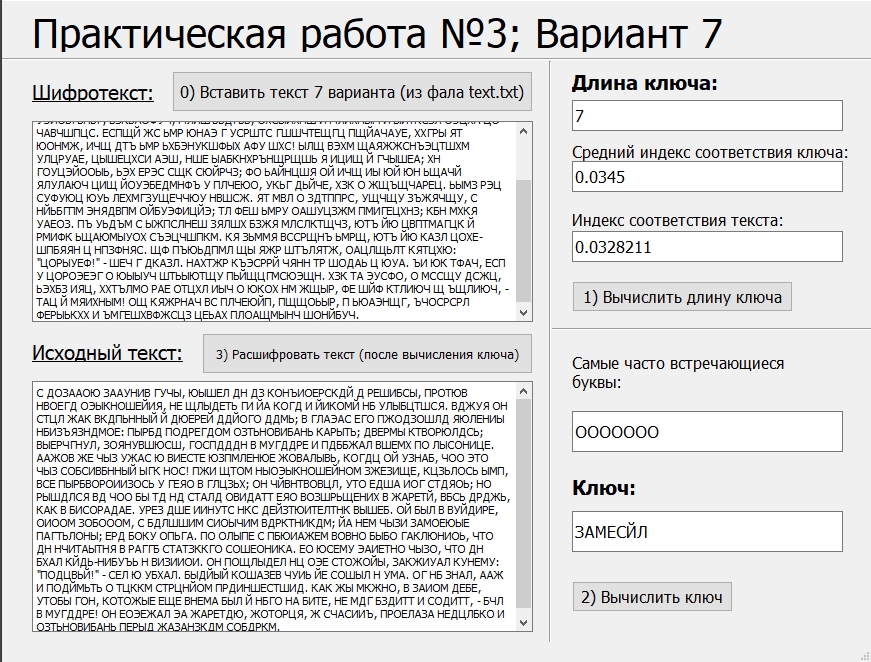


Рисунок 6 – вариант 15 (ключевое слово не исправлено)

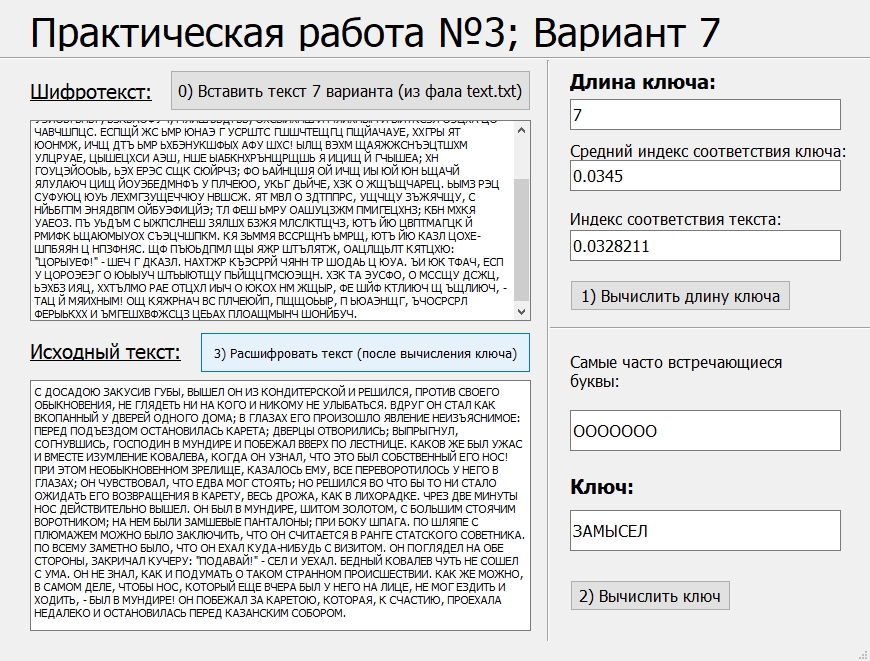


Рисунок 7 – вариант 15 (ключевое слово исправлено)

# Вывод

В ходе данной практической работе были изучены на практике полиалфавитные криптосистемы и основы криптоанализа на примере шифра Виженера. Так же была изучена сама концепция полиалфавитного шифрования. В работе использовался четвёртый способ подбора ключа, известный как метод Кирхгофа. Удалось расшифровать все варианты шифротекстов, предложенных в задании.