Исследование криптографических шифров на основе перестановки символов

Выполнил: Макаров Алексей Игоревич, 3 курс 4 группа 2 подгруппа

2024

**Ход работы**

**Вариант 8**

В результате выполнения лабораторной работы было выполнено зашифрование/расшифрование текстового документа, созданного на основе перестановочных шифров.

Сущность перестановочного шифрования состоит в том, что исходный текст (М) и зашифрованный текст (С) основаны на использовании одного и того же алфавита, а тайной или ключевой информацией является алгоритм перестановки. Шифры перестановки относятся к классу симметричных. Элементами текста могут быть отдельные символы.

В классической криптографии шифры перестановки делятся на два подкласса:

• шифры простой, или одинарной, перестановки – при зашифровании символы открытого текста Мi перемещаются с исходных позиций в новые (в шифртексте Сi) один раз;

• шифры сложной, или множественной, перестановки – при зашифровании символы открытого текста Мi перемещаются с исходных позиций в новые (в шифртексте Сi) несколько раз.

Среди шифров рассматриваемого подкласса иногда выделяют шифры простой перестановки (или перестановки без ключа). Символы открытого текста Мi перемешиваются по каким-либо правилам. Формально каждое из таких правил может рассматриваться в качестве ключа.

Основой современных шифров рассматриваемого типа является геометрическая фигура, обычно прямоугольник или прямоугольная матрица. В ячейки этой фигуры по определенному маршруту (слева направо, сверху вниз или каким-либо иным образом) записывается открытый текст. Для получения шифрограммы нужно записать символы этого сообщения в иной последовательности, т. е. по иному маршруту. Согласна таблице заданий – мое приложение реализует шифр на основе маршрута змея. Реализация представлена в листинге 1.

|  |
| --- |
| def snake\_permutation(text):  """  Переставляет текст в змейку.  """  rows = int(len(text) \*\* 0.5)  if rows \* rows < len(text):  rows += 1  columns = rows  matrix = [['' for \_ in range(columns)] for \_ in range(rows)]  direction = 0 # 0 - вправо, 1 - вниз  row, col = 0, 0  for char in text:  matrix[row][col] = char  if direction == 0:  if col == columns - 1:  direction = 1  row += 1  else:  col += 1  else:  if row == rows - 1:  direction = 0  col += 1  else:  row += 1  return ''.join([''.join(row) for row in matrix])  def encrypt\_snake(text):  """  Шифрует текст с использованием маршрутной перестановки "змейка".  """  snake\_text = snake\_permutation(text)  encrypted\_text = ''.join(chr((ord(char) - ord('A') + 1) % 26 + ord('A')) for char in snake\_text)  return encrypted\_text  def decrypt\_snake(encrypted\_text):  """  Расшифровывает текст, зашифрованный с использованием маршрутной перестановки "змейка".  """  snake\_text = ''.join(chr((ord(char) - ord('A') - 1) % 26 + ord('A')) for char in encrypted\_text)  original\_text = snake\_permutation(snake\_text)  return original\_text |

Листинг 1 – Реализация зашифрования/расшифрования

Таким образом код наглядно показывает процесс шифрования и расшифрования: перед глазами матрица, по которой можно визуально отследить процесс шифрования. Результат работы кода представлен на рисунке 1 и 2.

A white background with black text

Description automatically generated

Рисунок 1 – Результат зашифрования сообщения

A screenshot of a computer program

Description automatically generated

Рисунок 2 – Результат расшифрования сообщения

Шифры множественной перестановки. Особенностью шифров данного подкласса является минимум двукратная перестановка символов шифруемого сообщения. В простейшем случае это может задаваться перемешиванием не только столбцов (как в примере 4), но и строк. Таким образом, этот случай соответствует использованию двух основных ключей: длина одного из них равна числу столбцов, другого – числу строк. К ключевой информацию мы можем относить также способы вписывания сообщения и считывания отдельных символов из текущего столбца матрицы.

Реализация этого шифра представлена в листинге 2.

|  |
| --- |
| def permute\_text(text, keyword):  """  Переставляет символы в тексте в соответствии с ключевым словом.  """  # Создаем словарь для хранения позиций символов в ключевом слове  keyword\_dict = {char: index for index, char in enumerate(keyword)}    # Сортируем символы текста по их позициям в ключевом слове  sorted\_text = ''.join(sorted(text, key=lambda char: keyword\_dict.get(char, len(keyword))))    return sorted\_text  def encrypt\_with\_keyword(text, keyword):  """  Шифрует текст с использованием метода множественной перестановки с ключевым словом.  """  permuted\_text = permute\_text(text, keyword)  encrypted\_text = ''.join(chr((ord(char) - ord('A') + 1) % 26 + ord('A')) for char in permuted\_text)  return encrypted\_text  def decrypt\_with\_keyword(encrypted\_text, keyword):  """  Расшифровывает текст, зашифрованный с использованием метода множественной перестановки с ключевым словом.  """  permuted\_text = ''.join(chr((ord(char) - ord('A') - 1) % 26 + ord('A')) for char in encrypted\_text)  original\_text = permute\_text(permuted\_text, keyword)  return original\_text[::-1]  text = "HELLO"  print("исходный текст: ",text)  print("змейка:")  encrypted\_text = encrypt\_snake(text)  print(f"Зашифрованный текст: {encrypted\_text}")  decrypted\_text = decrypt\_snake(encrypted\_text)  print(f"Расшифрованный текст: {decrypted\_text}")  keyword = "MAKAROVALEXEY"  encrypted\_text = encrypt\_with\_keyword(text, keyword)  print(f"Зашифрованный текст: {encrypted\_text}")  decrypted\_text = decrypt\_with\_keyword(encrypted\_text, keyword)  print(f"Расшифрованный текст: {decrypted\_text}") |

Листинг 2 – Реализация шифра множественной перестановки

Таким образом код поэтапно показывает процесс шифрования и расшифрования: от построение таблицы до сортировки. Результат работы кода представлен на рисунке 3 и 4.

A screenshot of a computer

Description automatically generated

Рисунок 3 – Результат зашифрования сообщения

A screenshot of a computer

Description automatically generated

Рисунок 4 – Результат расшифрования сообщения

На основе исходного сообщения и текста, полученного в результате шифрования, был получены гистограммы частот, которые представлены на рисунке 5.

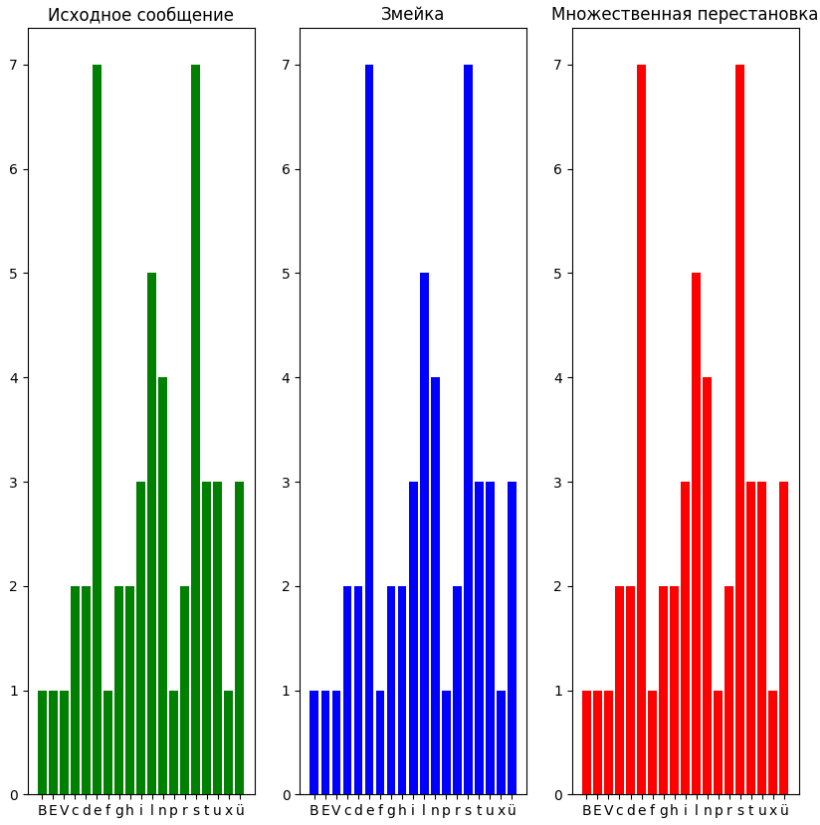


Рисунок 5 – Гистограммы частот символов

Гистограммы одинаковые, так как символы у нас не изменяются, а лишь меняют свой порядок.

Таким образом, была разработана программа для зашифрования/расшифрования текста, на основе которых были сделаны гистограммы частот. Было изучено и приобретены практические навыки разработки и использования приложений для реализации перестановочных шифров.