Приложение А

Вариант	Выполнил	Группа
7	Тимофеев	437-2

Вариант 7. Перестановочный шифр

Напишите программу, позволяющую зашифровать и расшифровать сообщения с использованием перестановочного шифра. Количество столбцов и строк задается в программе. Входные и выходные данные запишите в файл типа .txt.

Перестановочный шифр:

Перестановочный шифр - это метод симметричного шифрования, в котором элементы исходного открытого текста меняют местами. Элементами текста могут быть отдельные символы (самый распространённый случай), пары букв, тройки букв, комбинирование этих случаев и так далее.

> Типичными примерами перестановки являются анаграммы.

Описание алгоритма:

Алгоритм шифровки:

- создаём таблицу с заданным количеством столбцов
- записываем символы входной строки в таблицу построчно (слева-направо сверху-вниз), если в строке таблице нет свободных ячеик продолжаем писать с новой строки
- после того как входящая строка записана - считываем символы по столбцам (сверху-вниз слева-направо)

Алгоритм дешифровки:

- создаём таблицу с заданным количеством столбцов
- расчитываем количество полных столбцов
- в полных столбцах записываем элементы выходной строки по столбцам (сверху-вниз слева-направо)
- в неполных столбцах оставляем 1 пустой символ
- считываем (слева-направо сверху-вниз) (игнорируя пустые символы в конце)

Описываем класс для шифорвки и дешифровки шифра перестановками

У класса есть:

- **Конструктор**: Принимает количество столбцов в таблице перестановок
- **метод encode**: Метод для шифровки сообщения. Принимает текст который необходимо зашифровать
- **метод decode**: Метод для расшифорфки сообщения. Принимает текст который необходимо расшифровать

In [1]:
```
from pprint import pprint
```

```python
class Permutations:
    def __init__(self, table_cols: int):
        self._T = table_cols

    def encode(self, text: str):
        K: int = len(text) // self._T  # Количество полных стобцов
        D: int = len(text) %  self._T  # Количество элементов в неполной строке
        C: list[str] = []
        for i in range(self._T):
            if D > i:
                d = 1
            else:
                d = 0
            for j in range(K+d):
                C.append(text[j*self._T+i])

        return ''.join(C)

    def decode(self, text):
        K: int = len(text) // self._T  # Количество полных стобцов
        D: int = len(text) %  self._T  # Количество элементов в неполной строке
        M: list[str] = []
        for i in range(K):
            for j in range(D+1):
                k = i+j*(K+1)
                M.append(text[k])
            r = D*(K+1)
            for j in range(1,self._T-D):
                k = i+r+j*K
                M.append(text[k])
        for j in range(D):
            M.append(text[(j+1)*(K+1)-1])
        return ''.join(M)
```

Считываем текст который хотим зашифровать из файла input.txt

In [2]:
```python
input_name = 'input.txt'

with open(input_name, 'r', encoding='utf-8') as file:
    исходеый_текст = file.read()

print(f"Содержание {input_name}:")
pprint(исходеый_текст)
исходеый_текст
```

```
Содержание input.txt:
'123456789 Вот он текст который необходимо зашифровать'
```

Out[2]:
```
'123456789 Вот он текст который необходимо зашифровать'
```

Создаём объект Шифратора-Дешифратора

In [3]:
```python
количество_столбцов = 5
шифр_перестановками = Permutations(количество_столбцов)
```

Шифруем строку и смотрим что получилось

In [4]:
```python
зашифрованный_текст = шифр_перестановками.encode(исходеый_текст)
pprint(зашифрованный_текст)
зашифрованный_текст
```

```
'16Внст хоиа27о тоно фт38тт редзрь49 екыоиао5 окойбмшв'
```

Out[4]:
```
'16Внст хоиа27о тоно фт38тт редзрь49 екыоиао5 окойбмшв'
```

Теперь дешифруем строку обратно

In [5]:
```
расшифрованный_текст = шифр_перестановками.decode(зашифрованный_текст)
pprint(расшифрованный_текст)
расшифрованный_текст
```

```
'123456789 Вот он текст который необходимо зашифровать'
```

Out[5]:
```
'123456789 Вот он текст который необходимо зашифровать'
```

Запустим ещё раз с другим файлом

In [6]:
```
input_name = 'input2.txt'

with open(input_name, 'r', encoding='utf-8') as file:
    исходеый_текст = file.read()

print(f"Содержание {input_name}:")
pprint(исходеый_текст)
исходеый_текст
```

```
Содержание input2.txt:
('Мы знаем, что такое байты,\n'
 'Система поиска и сайты,\n'
 'Умеем файлы создавать,\n'
 'Картинки «мышкой» рисовать,\n'
 '\n'
 'И через интернет общаться,\n'
 'И в алгоритмах разбираться,\n'
 'И понимать язык программ.\n'
 '\n')
```

Out[6]:
```
'Мы знаем, что такое байты,\nСистема поиска и сайты,\nУмеем файлы создавать,\nКартинки «мышкой» рисовать,\n\nИ через интернет общаться,\nИ в алгоритмах разбираться,\nИ понимать язык программ.\n\n'
```

Создаём объект Шифратора-Дешифратора

Изменим количество столбцов

In [7]:
```
количество_столбцов = 10
шифр_перестановками = Permutations(количество_столбцов)
```

Шифруем строку и смотрим что получилось

In [8]:
```
зашифрованный_текст = шифр_перестановками.encode(исходеый_текст)
pprint(зашифрованный_текст)
зашифрованный_текст
```

```
('Мчбта\n'
 'лтк», оИтрпзмытае Уыьи \n'
 'иб маоым оймим , р\n'
 'нщватнк.з та ес\n'
 '«иИта хьи \n'
 'нты сеоКмс ета смп\n'
 'аа,памзаыочрьлряарек\n'
 'ой дршвенсга,томоСитфаткареяоз\n'
 'ьг,еисыавиотет,рбИ р  ск,йанйьз \n'
 'ии яа')
```

Out[8]:
```
'Мчбта\nлтк», оИтрпзмытае Уыьи \nиб маоым оймим , р\nнщватнк.з та ес\n«иИта хьи \nнты сеоКмс ета смп\naa,памзаыочрьлряарек\noй дршвенсга,томоСитфаткареяоз\nьг,еисыавиотет,рбИ р  ск,йанйьз \nии яа'
```

Теперь дешифруем строку обратно

In [9]:
```
расшифрованный_текст = шифр_перестановками.decode(зашифрованный_текст)
pprint(расшифрованный_текст)
расшифрованный_текст
```

```
('Мы знаем, что такое байты,\n'
 'Система поиска и сайты,\n'
 'Умеем файлы создавать,\n'
 'Картинки «мышкой» рисовать,\n'
 '\n'
 'И через интернет общаться,\n'
 'И в алгоритмах разбираться,\n'
 'И понимать язык программ.\n'
 '\n')
```

Out[9]:
```
'Мы знаем, что такое байты,\nСистема поиска и сайты,\nУмеем файлы создавать,\nКартинки «мышкой» рисовать,\n\nИ через интернет общаться,\nИ в алгоритмах разбираться,\nИ понимать язык программ.\n\n'
```