##### Перестановочный шифр

***Цель работы***: освоить метод шифрования перестановочным шифром.

***Задание:***

* Разработать алгоритм для шифрования сообщений перестановочным шифром;
* Разработать алгоритм для дешифрования сообщений зашифрованных перестановочным шифром;
* Составить приложение для шифрования/дешифрования текстовой строки с использованием перестановочного шифра;
* Составить приложение для шифрования/дешифрования отдельного файла с использованием перестановочного шифра.

***Выполнение работы***:

Для решения поставленной задачи мною был выбран язык TypeScript

Класс для шифрования содержит в себе две публичных функции

/\*\*

     \* Encode function

     \*

     \* @param source data to encode

     \* @param key encryption key

     \*/

    encode(source: Buffer, key: Buffer): Buffer;

    /\*\*

     \* Decode function

     \*

     \* @param source data to encode

     \* @param key encryption key

     \*/

    decode(source: Buffer, key: Buffer): Buffer;

Для входных и выходных параметров был выбран тип Buffer который представляет собой массив байтов, таким образом перестановка происходит побайтово, а не посимвольно (отражается на выходных данных для символов, отличных от латиницы), благодаря чему можно работать как со строками, так и с целыми файлами не меняя код

Так же в классе присутствует приватная функция которая строит таблицу байтов и изменяет порядок её столбцов в зависимости от ключа

type Byte = number;

type ByteTable = Byte[][];

/\*\*

     \* Creates table of Bytes and reorders it according to key

     \*

     \* @param source data to map

     \* @param key encryption key

     \* @param en encode/decode flag, flips table fill logic

     \*/

private toByteTable(source: Buffer, key: Buffer, en: Boolean): ByteTable;

Функции encode и decode генерируют одинаковую таблицу из полученных на вход данных, однако по-разному её складывают в результирующий массив

encode складывает столбцы:

        const result: Byte[] = table.reduce(

            (col: Byte[], acc: Byte[]) => [...col, ...acc],

            [] as Byte[]

        );

а decode строки:

        const result: Byte[] = [];

        for (let i: number = 0; i < table[0].length; i++) {

            for (let j: number = 0; j < table.length; j++) {

                result.push(table[j][i]);

            }

        }

Полный код класса: <https://github.com/wbydc/ISUCT-data-protection/blob/master/src/encoders/lab1/Encoder.ts>

Как запустить и протестировать: <https://github.com/wbydc/ISUCT-data-protection/blob/master/README.md>