«Защита информации» Лабораторная работа №1 ДОМАШИННЫЕ МЕТОДЫ ШИФРОВАНИЯ	Выполнил:	Зенин М.А.
	Преподаватель:	Жариков Д.Н.
	Дата выполнения:	
	Подпись преподавателя	

Исходный код программы:

Шифр цезаря:

```
def encrypt(text: str, alphabet: str, shift: int, debug: bool = False):
    text = text.replace(" ", "")
    alphabet = sorted(list(set(alphabet.replace(" ", "").lower())))

    alphabet_length = len(alphabet)

    if debug:
        print(f"alphabet length: {alphabet_length}")

    output = ""
    for letter in text:
        letter_index =alphabet.index(letter)
        output += alphabet[(letter_index+shift) % alphabet_length]

    return output


def decrypt(text: str, alphabet: str, shift: int, debug: bool = False):
    text = text.replace(" ", "")
    alphabet = sorted(list(set(alphabet.replace(" ", "").lower())))

    alphabet_length = len(alphabet)

    if debug:
        print(f"alphabet length: {alphabet_length}")

    output = ""
```

```
    for letter in text:
        letter_index = alphabet.index(letter)
        output += alphabet[(letter_index-shift) % alphabet_length]

    return output
```

Шифр ключевым словом:

```
def encrypt(text: str, keyword: str, debug: bool = False):
    text = text.replace(" ", "").lower()
    keyword = "".join(keyword.replace(" ", "").lower())

    keyword_length = len(keyword)
    text_length = len(text)

    order = sorted(range(keyword_length), key=lambda i: keyword[i])

    if debug:
        print(f"Text length {text_length} Kw length {keyword_length}")
        indexes = sorted(list(set(keyword)))
        print("".join([str(indexes.index(letter)) for letter in keyword]))
        for i in range(0, text_length, keyword_length):
            print(text[i:i+keyword_length])
        print()

    output = ""
    for column in order:
        for index in range(column, text_length, keyword_length):
            output += text[index]

    return output


def decrypt(text: str, keyword: str, debug: bool = False) -> str:
    text = text.replace(" ", "").lower()
    keyword = "".join(keyword.replace(" ", "").lower())

    keyword_length = len(keyword)
    text_length = len(text)
```

```
order = sorted(range(keyword_length), key=lambda i: keyword[i])

rows = (text_length + keyword_length - 1) // keyword_length
# full_columns = text_length % keyword_length
full_columns = text_length % keyword_length
if not full_columns:
    full_columns = keyword_length

if debug:
    print(f"Text length {text_length} Kw length {keyword_length}")
    print(f"Rows: {rows}, Full columns: {full_columns}")

column_lengths = []
for index in range(keyword_length):
    column_index = order[index]
    length = rows if column_index < full_columns else rows - 1
    column_lengths.append(max(0, length))

if debug:
    print(f"Column lengths: {column_lengths}")

columns = []
index = 0
for length in column_lengths:
    columns.append(text[index:index+length])
    index += length

if debug:
    print("Columns:")
    for row in range(rows):
        for column in columns:
            if len(column) > row:
                print(column[row], end="")
            else:
                print(end=" ")
        print()

sorted_columns = [""] * keyword_length
for index, column_index in enumerate(order):
    sorted_columns[column_index] = columns[index]
```

```
    if debug:
        print("Sorted columns:")
        for row in range(rows):
            for column in sorted_columns:
                if len(column) > row:
                    print(column[row], end="")
                else:
                    print(end=" ")
            print()
        print()

    output = []
    for row in range(rows):
        for column in range(keyword_length):
            if row < len(sorted_columns[column]):
                output.append(sorted_columns[column][row])

    return "".join(output)
```

Функция читает файл, если это файл, иначе возвращает сам текст

```
def resolve_file(text: str):
    p = Path(text)
    if p.exists():
        if p.is_file():
            return p.read_text()
    return text
```

Шифрующая функция

```
def main(text: str, alphabet: str, shift: int, keyword: str, out: str | None = None, debug:
bool = False) -> None:
    text = resolve_file(text)
    alphabet = resolve_file(alphabet)
    keyword = resolve_file(keyword)

    print(text, alphabet, keyword)
```

```
    result = caesar_cypher.encrypt(keyword_cypher.encrypt(text, keyword, debug),
alphabet, shift, debug)

    if out:
        with open(out, "w", encoding="utf-8") as f:
            f.write(result)
    else:
        print(result)
```

Дешифрующая функция

```
def main(text: str, alphabet: str, shift: int, keyword: str, out: str | None = None, debug:
bool = False) -> None:
    text = resolve_file(text)
    alphabet = resolve_file(alphabet)
    keyword = resolve_file(keyword)

    result = caesar_cypher.decrypt(keyword_cypher.decrypt(text, keyword, debug),
alphabet, shift, debug)
    # result = keyword_cypher.decrypt(caesar_cypher.decrypt(text, alphabet, shift,
debug), keyword, debug)

    if out:
        with open(out, "w", encoding="utf-8") as f:
            f.write(result)
    else:
        print(result)
```

Основная функция:

```
    parser = argparse.ArgumentParser(description="Зашифровать шифром Цезаря с
ключевым словом")
    parser.add_argument("text", help="Шифруемый текст (или файл с текстом)")
    parser.add_argument("alphabet", help="Алфавит (или файл с алфавитом)")
    parser.add_argument("shift", type=int, help="Смещение шифра Цезаря")
    parser.add_argument("keyword", help="Ключевое слово (или файл с ключевым
словом)")
    parser.add_argument("--out", "-o", help="Выходной файл")
    parser.add_argument("--debug", "-d", help="Дебаг вывод", action="store_true")
```

```
args = parser.parse_args()
main(args.text, args.alphabet, args.shift, args.keyword, args.out, args.debug)
```

Работа программы:

```
PS D:\homework\data_security\lab1> python .\encrypt.py ПрикладнаяМатематика ПрикладнаяМатематика 2 Шифр
ПрикладнаяМатематика ПрикладнаяМатематика Шифр
яедклмрееелиппмтнеаа
PS D:\homework\data_security\lab1>
```

```
PS D:\homework\data_security\lab1> python .\decrypt.py яедклмрееелиппмтнеаа ПрикладнаяМатематика 2 Шифр
прикладнаяматематика
PS D:\homework\data_security\lab1>
```