МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

ОТЧЕТ

по дисциплине
«Программирование»

по теме: РГЗ

Студент:

В.Е. Клишин

Группы ИКС-431

Преподаватель: *А.И. Вейлер*

Новосибирск 2025

Задание:

Вариант 6. Шифрование текста шифром Цезаря

Разработать программу Cezar, выполняющую шифрование в заданном тексте и DeCezar – дешифровку текста. Текст до шифрования, после шифрования и после дешифровки должен выводиться на экран.

Критерии оценки: Оценка «удовлетворительно»: реализована проверка того, что исходный текст и полученные после дешифровки совпадают. Не предусмотрено динамическое выделение памяти под входные данные. Функции записаны в статическую библиотеку.

Оценка «хорошо»: на вход программы подается 2 файла. Первый файл содержит текст на русском языке. Второй файл будет содержать зашифрованный текст. Обязательно динамическое выделение памяти под входные данные. Функции записаны в статическую библиотеку.

Оценка «отлично»: оценить криптостойкость шифра. Обязательно динамическое выделение памяти под входные данные. Функции записаны в динамическую библиотеку. Указания к выполнению задания Одним из первых документально зафиксированных шифров является шифр Цезаря, использовавшийся известным полководцем в собственной переписке. В шифре каждая буква замещается на букву, находящуюся k символами правее в алфавите по модулю, равному количеству букв в алфавите: $c = (m + k) \bmod n$, где m- порядковый номер буквы в алфавите, C – порядковый номер замещающей буквы, n – мощность входного алфавита (количество букв в используемом алфавите). Таким образом, ключом шифрования здесь является число k, определяющее размер смещения. Таким образом, ключом шифрования здесь является число k, определяющее размер смещения. Очевидно, что обратной подстановкой является $m = (c - k) \bmod n$. При необходимости алфавит можно расширить знаками препинания, заглавными буквами, цифрами, чтобы шифр мог обрабатывать все символы исходного текста. Общее количество допустимых ключей равно n.

Анализ задачи:

Основной алгоритм - это **шифр Цезаря**, который является подстановочным шифром.

Математическая формула шифра Цезаря:

- **Шифрование:** $C=(M+K) \bmod N$
- **Дешифрование:** $M=(C-K) \bmod N$

Где:

- M - порядковый номер исходной буквы в алфавите (0-индексированный).
- C - порядковый номер зашифрованной буквы в алфавите (0-индексированный).
- K - ключ смещения.
- N - мощность алфавита (количество букв).

Основной Алгоритм Шифрования/Дешифрования (Функции CEZAR и DECEZAR)

Псевдокод для CEZAR(текст, ключ):

Функция CEZAR(текст: строка широких символов, ключ: целое число):

ДлинаТекста = Длина(текст)

Для каждого символа в тексте от i = 0 до ДлинаТекста - 1:

Если символ[i] является буквой:

Если символ[i] является заглавной буквой:

Если символ[i] находится в диапазоне 'А'...'Я' (русский алфавит):

Символ[i] = 'А' + (Символ[i] - 'А' + ключ) % 33

Иначе Если символ[i] находится в диапазоне 'A'...'Z' (английский алфавит):

Символ[i] = 'A' + (Символ[i] - 'A' + ключ) % 26

Иначе (символ[i] является строчной буквой):

Если символ[i] находится в диапазоне 'а'...'я' (русский алфавит):

Символ[i] = 'а' + (Символ[i] - 'а' + ключ) % 33

Иначе Если символ[i] находится в диапазоне 'a'...'z' (английский алфавит):

Символ[i] = 'a' + (Символ[i] - 'a' + ключ) % 26

// Иначе (не буква), символ остается неизменным

Вернуть текст

Псевдокод для DECEZAR(текст, ключ):

```
Функция DECEZAR(текст: строка широких символов, ключ: целое число):
    ДлинаТекста = Длина(текст)
    Для каждого символа в тексте от i = 0 до ДлинаТекста - 1:
        Если символ[i] является буквой:
            Если символ[i] является заглавной буквой:
                Если символ[i] находится в диапазоне 'А'...'Я':
                // Для корректного обращения отрицательного % N: (X % N + N) % N
                Символ[i] = 'А' + ((Символ[i] - 'А' - ключ) % 33 + 33) % 33
                Иначе Если символ[i] находится в диапазоне 'A'...'Z':
                Символ[i] = 'A' + ((Символ[i] - 'A' - ключ) % 26 + 26) % 26
            Иначе (символ[i] является строчной буквой):
                Если символ[i] находится в диапазоне 'а'...'я':
                    Символ[i] = 'а' + ((Символ[i] - 'а' - ключ) % 33 + 33) % 33
                Иначе Если символ[i] находится в диапазоне 'a'...'z':
                    Символ[i] = 'a' + ((Символ[i] - 'a' - ключ) % 26 + 26) % 26
    Вернуть текст
```

Криптостойкость Шифра Цезаря:

Метод оценки криптостойкости: Шифр Цезаря **очень слаб**. Его криптостойкость оценивается как **низкая**.

- **Малый размер ключа:** Количество возможных ключей `k` равно размеру алфавита `N`. Для русского алфавита это 33, для английского - 26.
- **Атаки методом грубой силы (Brute-Force Attack):** Перебор всех возможных ключей (`k`) занимает очень мало времени. Злоумышленник может просто попробовать все 26 или 33 варианта расшифровки, и обычно осмысленный текст будет легко различим.
- **Частотный анализ:** Это более сложный, но очень эффективный метод. В любом достаточно длинном тексте буквы встречаются с определенной статистической частотой. Например, в русском языке самая частая буква - 'О', затем 'Е', 'А' и т.д. В английском - 'E', 'T', 'A', 'O'. Шифр Цезаря просто сдвигает все частоты, сохраняя их относительное распределение. Злоумышленник может проанализировать частоты

символов в зашифрованном тексте, сопоставить их с известными частотами языка и определить смещение (ключ `k`).

Тестовые данные:

```
// --- Тесты на ШИФРОВАНИЕ (функция CEZAR) ---
void test_cezar_russian_uppercase_encrypt(void) {
    wchar_t* original = wcsdup_custom(L"ПРИВЕТ");
    wchar_t* expected = L"ТУЛЕИХ";
    wchar_t* encrypted = CEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, encrypted, L"Шифрование русских заглавных");
    free(original);
}

void test_cezar_russian_lowercase_encrypt(void) {
    wchar_t* original = wcsdup_custom(L"привет");
    wchar_t* expected = L"тулеих";
    wchar_t* encrypted = CEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, encrypted, L"Шифрование русских строчных");
    free(original);
}

void test_cezar_english_uppercase_encrypt(void) {
    wchar_t* original = wcsdup_custom(L"HELLO");
    wchar_t* expected = L"KHOOR";
    wchar_t* encrypted = CEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, encrypted, L"Шифрование английских заглавных");
    free(original);
}

void test_cezar_english_lowercase_encrypt(void) {
    wchar_t* original = wcsdup_custom(L"hello");
    wchar_t* expected = L"khoor";
    wchar_t* encrypted = CEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, encrypted, L"Шифрование английских строчных");
    free(original);
}

void test_cezar_mixed_encrypt(void) {
    wchar_t* original = wcsdup_custom(L"Привет, World!");
    wchar_t* expected = L"Тулеих, Zruog!";
    wchar_t* encrypted = CEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, encrypted, L"Шифрование смешанного текста");
    free(original);
}

void test_cezar_non_alpha_chars_encrypt(void) {
    wchar_t* original = wcsdup_custom(L"123!@#$ %^&*()");
    wchar_t* expected = L"123!@#$ %^&*()";
    wchar_t* encrypted = CEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, encrypted, L"Неалфавитные символы при
шифровании");
    free(original);
}
```

```
// --- Тесты на ДЕШИФРОВАНИЕ (функция DECEZAR) ---

void test_cezar_russian_uppercase_decrypt(void) {
    wchar_t* original = wcsdup_custom(L"ТУЛЕИХ"); // Зашифрованный текст
    wchar_t* expected = L"ПРИВЕТ"; // Ожидаемый дешифрованный текст
    wchar_t* decrypted = DECEZAR(original, 3); // Используем DECEZAR
    ASSERT_WCS_EQUAL(expected, decrypted, L"Дешифрование русских заглавных");
    free(original);
}

void test_cezar_russian_lowercase_decrypt(void) {
    wchar_t* original = wcsdup_custom(L"тулеих");
    wchar_t* expected = L"привет";
    wchar_t* decrypted = DECEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, decrypted, L"Дешифрование русских строчных");
    free(original);
}

void test_cezar_english_uppercase_decrypt(void) {
    wchar_t* original = wcsdup_custom(L"KHOOR");
    wchar_t* expected = L"HELLO";
    wchar_t* decrypted = DECEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, decrypted, L"Дешифрование английских заглавных");
    free(original);
}

void test_cezar_english_lowercase_decrypt(void) {
    wchar_t* original = wcsdup_custom(L"khoor");
    wchar_t* expected = L"hello";
    wchar_t* decrypted = DECEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, decrypted, L"Дешифрование английских строчных");
    free(original);
}

void test_cezar_mixed_decrypt(void) {
    wchar_t* original = wcsdup_custom(L"Тулеих, Zruog!"); // Зашифрованный смешанный
текст
    wchar_t* expected = L"Привет, World!"; // Ожидаемый дешифрованный текст
    wchar_t* decrypted = DECEZAR(original, 3);
    ASSERT_WCS_EQUAL(expected, decrypted, L"Дешифрование смешанного текста");
    free(original);
}
```

Скриншоты с результатами:

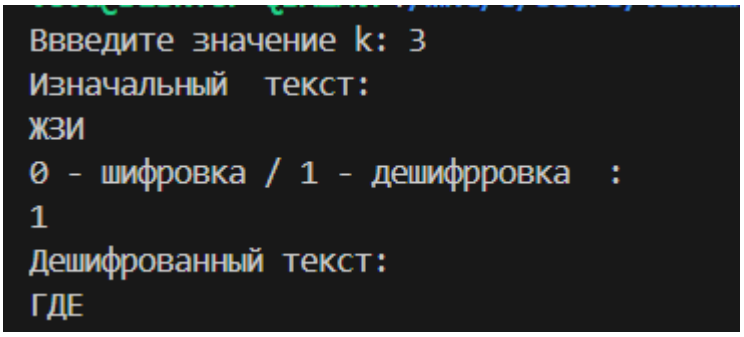

Рисунок 1 – Дешифровка строки

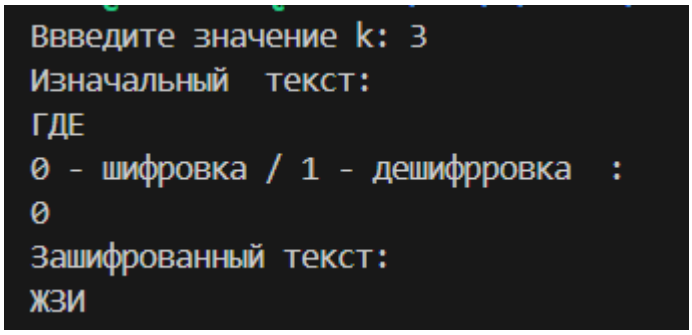

Рисунок 2 – Шифровка строки

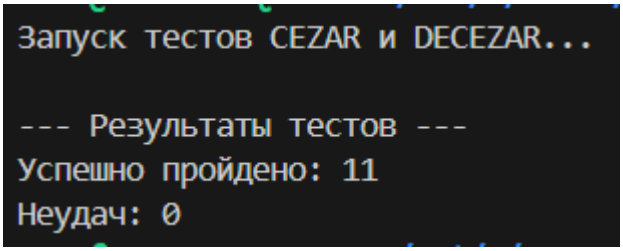

Рисунок 3 – Запуск тестов

Ссылка на GitHub:

МОЙ ГИТХАБ