**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

Факультет «Информационных технологий и программной инженерии»

Кафедра «Систем обработки данных»

|  |  |
| --- | --- |
| Направление подготовки: | 09.03.02 - Информационные системы и технологии |
| Направленность (профиль): | Технологии проектирования защищенных систем обработки данных |

**Практическая работа № 2**

по дисциплине:

|  |
| --- |
| Кроссплатформенное программирование |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Выполнили студенты группы ИСТ-321 | |  |
|  |  | Чаушьянц Р.В. | | |
|  |  | Нариц А. А. | | |
|  |  |  | Заведующий кафедры СОД | |
| *оценка* |  |  | *уч. степень, уч. звание* | |
|  |  | Бородянский Юрий Михайлович | | |
| *дата, подпись* |  | *Фамилия И. О.* | | |

**Использование операторов и работа с примитивными типами. Приложение, реализующее побитовые операции.**

**Цель работы**

Изучение общих принципов программирования в среде java. Написание приложения, осуществляющего «шифрование» текста путем преобразования символов с помощью побитовых операций.

**Задание**

Необходимо создать консольное приложение, осуществляющее «шифрование» небольшого текста путем сложения по "XOR" числового кода каждого символа с ключом "секрет". Кроме сложения по "XOR", приложение должно поддерживать шифрование с помощью циклического сдвига вдоль кодовой таблицы (алфавита) на заданное число позиций. Текст для шифрования взять произвольно в количестве не менее трех строк. Приложение должно иметь текстовое меню со следующими пунктами:

1. Шифрование с помощью операции "XOR"
2. Шифрование с помощью циклического сдвига
3. Ввести число позиций для шифрования путем циклического сдвига

При выборе пунктов 1 и 2, приложение должно распечатывать исходный текст, зашифрованный текст и расшифрованный текст. При выборе пункта 3 приложение должно запросить у пользователя число, которое определяет величину сдвига, затем также распечатать исходный, зашифрованный и расшифрованный текст. Числовое значение буквы исходного текста брать по её порядковому номеру в алфавите. Так букве "А" соответствует число 1, букве "Б" соответствует число 2 и т.д. Регистр букв не учитывать.

**Теоретическая часть**

**Изучение общих принципов программирования в среде Java. Написание приложения, осуществляющего «шифрование» текста путем преобразования символов с помощью побитовых операций**

**Введение** Java – это один из самых популярных языков программирования, разработанный компанией Sun Microsystems (ныне Oracle). Его главными преимуществами являются кроссплатформенность, надежность и удобство работы. Благодаря виртуальной машине Java (JVM) код можно запускать на разных операционных системах без изменений. Этот язык широко используется в веб-разработке, создании мобильных приложений, корпоративного программного обеспечения и других областях.

**Основные принципы программирования в Java** Программирование в Java базируется на нескольких ключевых концепциях:

1. **Объектно-ориентированное программирование (ООП)** – код организуется в классы и объекты, что упрощает его поддержку и расширение.
2. **Кроссплатформенность** – компиляция в байт-код позволяет выполнять программы на разных устройствах без изменений исходного кода.
3. **Автоматическое управление памятью** – благодаря механизму «сборки мусора» освобождение памяти происходит автоматически.
4. **Безопасность и надежность** – встроенные механизмы безопасности позволяют предотвращать выполнение вредоносного кода.

**Шифрование текста с помощью побитовых операций** Побитовые операции – это эффективный способ обработки данных на уровне отдельных битов. В криптографии такие операции применяются для простых методов шифрования, включая шифрование на основе логической операции XOR (исключающее ИЛИ).

Принцип работы XOR-шифрования заключается в том, что каждый бит входных данных изменяется в зависимости от соответствующего бита ключа. Если бит ключа равен 1, бит изменяется на противоположный, если 0 – остается неизменным. Такой метод позволяет зашифровать данные, а затем, применяя ту же операцию с тем же ключом, выполнить расшифровку.

**Заключение** Использование побитовых операций, таких как XOR, является эффективным методом простого симметричного шифрования. Java предоставляет удобные инструменты для работы с такими операциями, что делает возможным реализацию надежных алгоритмов защиты данных. Данный подход используется в различных областях, включая базовое шифрование, цифровую обработку сигналов и сетевую безопасность.

**Ход работы**

В ходе работы была написана программа, реализующая следующие функции: шифрование с помощью операции "XOR", шифрование с помощью циклического сдвига (рисунок 1). Логика программы продемонстрирована на схеме 1.

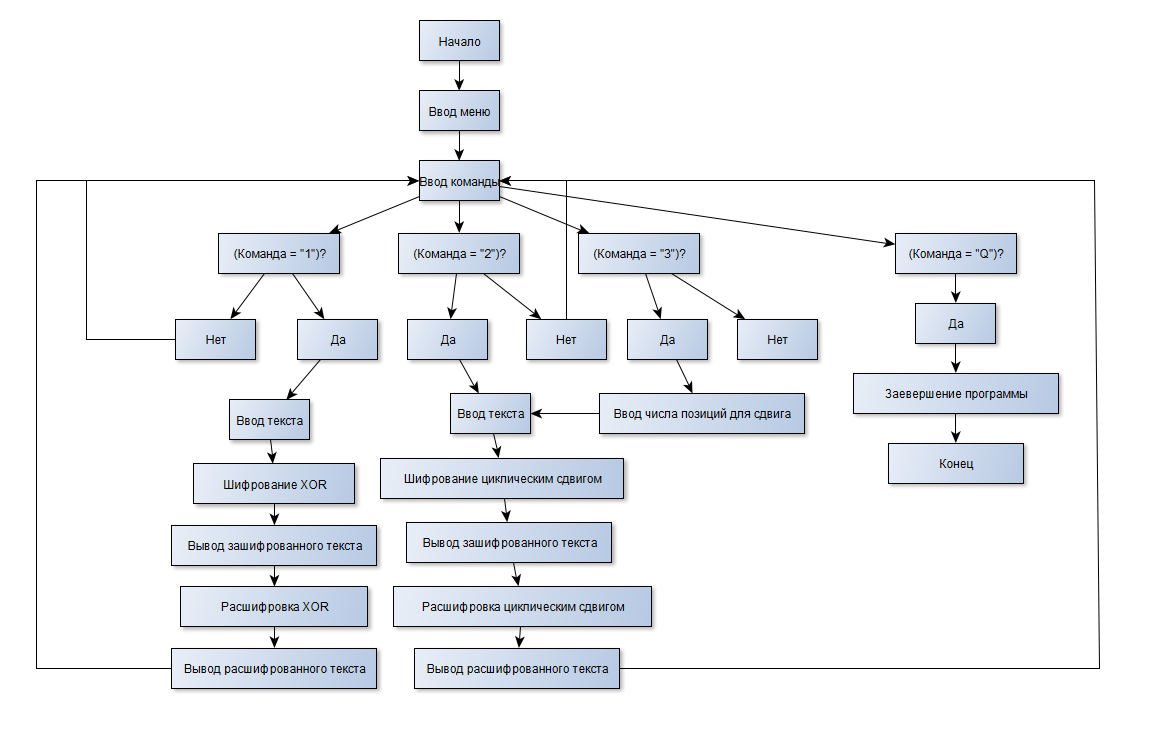
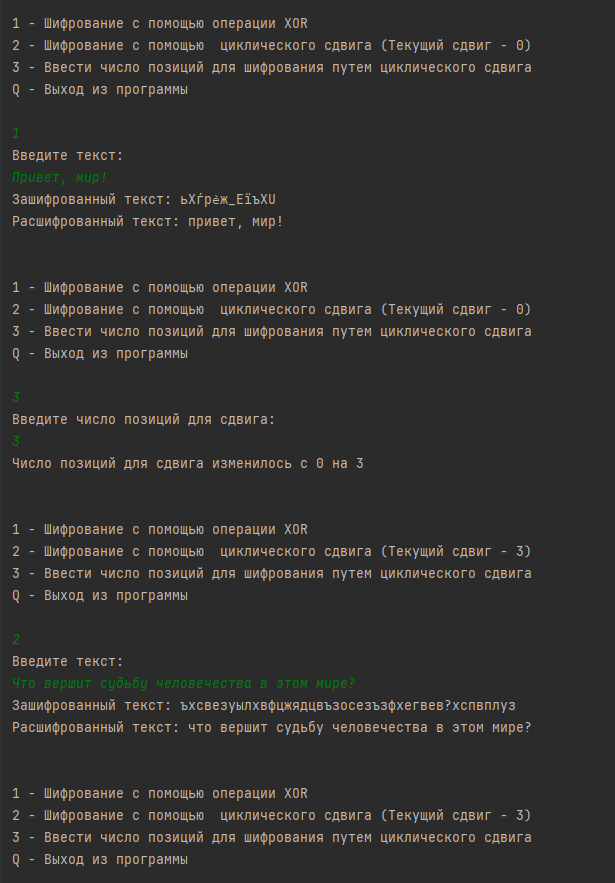


Схема 1. Логика работы программы



**Рисунок 1. Результат выполнения программы**

**Выводы**

Входе выполнения лабораторной работы, мы научились нескольким важным навыкам, которые пригодятся нам в будущем:

1. **Работа со строками и символами** – мы научились находить символы в алфавите, изменять их и обрабатывать текст, что полезно для программирования и работы с данными.
2. **Шифрование с использованием XOR** – этот метод важен в криптографии и защите данных, например, для шифрования сообщений.
3. **Циклический сдвиг** – помогает в обработке текста, создании шифров и работы с алгоритмами перестановок, что используется в защите информации и обработке данных.
4. **Работа с пользовательским вводом** – наш код умеет взаимодействовать с пользователем, считывать данные, обрабатывать их и выдавать результат, что важно при создании приложений и интерфейсов.
5. **Использование меню и обработка команд** – мы реализовали систему команд, которая делает программу удобной для пользователя, а это ключевой навык при разработке программного обеспечения.

Эти знания помогут нам создавать более сложные программы, понимать принципы шифрования и обработки данных, а также развивать навыки в программировании на Java.

**Приложение**

import java.util.Scanner;

public class Main {

// Константа для ключа шифрования

static final String KEY = "секрет";

// Алфавит, который используется для шифрования и дешифрования

static final String ALPHABET = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя,!?.:;'\" ";

// Создаём объект Scanner для ввода с консоли

static final Scanner scanner = new Scanner(System.in);

// Переменная для хранения сдвига для циклического сдвига

static int shift = 0;

/\*\*

\* XOR-шифрование – это метод симметричного шифрования, при котором каждый символ исходного текста

\* складывается по модулю 2 (операция XOR) с соответствующим символом ключа. Если ключ короче текста,

\* он повторяется циклически. Этот метод обеспечивает простое и обратимое преобразование текста.

\* @param text исходный или зашифрованный текст

\* @return результат шифрования или дешифрования

\*/

public static String xor\_cipher(String text) {

StringBuilder result = new StringBuilder();

// Проходим по каждому символу текста

for (int i = 0; i < text.length(); i++) {

// Находим индекс текущего символа в алфавите

int num\_text\_char = ALPHABET.indexOf(text.charAt(i));

// Находим индекс соответствующего символа ключа

int num\_key\_char = ALPHABET.indexOf(KEY.charAt(i % KEY.length()));

// Применяем операцию XOR

int encrypted\_char = num\_text\_char ^ num\_key\_char;

// Проверяем, есть ли символ в алфавите

if (num\_text\_char != -1 && num\_key\_char != -1) {

// Добавляем зашифрованный символ, если оба символа есть в алфавите

result.append(ALPHABET.charAt(encrypted\_char));

} else {

// Если символа нет в алфавите, добавляем оригинальный символ

result.append(text.charAt(i));

}

}

return result.toString();

}

/\*\*

\* Выполняет циклический сдвиг символов в тексте.

\* Сдвиг производится внутри ограниченного алфавита с использованием модульной арифметики,

\* что позволяет корректно обрабатывать границы массива символов.

\* @param text Исходный текст

\* @param encode Если true, выполняется шифрование (сдвиг вперёд), иначе – дешифрование (сдвиг назад)

\* @return Преобразованный текст

\*/

public static String cyclic\_shift(String text, boolean encode) {

StringBuilder result = new StringBuilder();

// Проходим по каждому символу текста

for (int i = 0; i < text.length(); i++) {

// Получаем индекс символа в алфавите

int index\_alfabet = ALPHABET.indexOf(text.charAt(i));

// Если символ есть в алфавите

if (index\_alfabet != -1) {

// Сдвигаем индекс на нужное количество позиций

int position\_shift = encode ? (index\_alfabet + shift) : (index\_alfabet - shift);

// Применяем модуль для правильной обработки границ алфавита

position\_shift = (position\_shift + ALPHABET.length()) % ALPHABET.length();

// Добавляем результат сдвига в строку

result.append(ALPHABET.charAt(position\_shift));

} else {

// Если символа нет в алфавите, добавляем оригинальный символ

result.append(text.charAt(i));

}

}

return result.toString();

}

// Обработка команды шифрования XOR

public static void command1() {

System.out.println("Введите текст:");

String original\_text = scanner.nextLine();

// Приводим текст к нижнему регистру перед шифрованием

String encrypted\_text = xor\_cipher(original\_text.toLowerCase());

System.out.println("Зашифрованный текст: " + encrypted\_text);

// Расшифровываем текст, используя ту же функцию

System.out.println("Расшифрованный текст: " + xor\_cipher(encrypted\_text) + '\n');

}

// Обработка команды шифрования циклическим сдвигом

public static void command2() {

System.out.println("Введите текст:");

String original\_text = scanner.nextLine();

// Приводим текст к нижнему регистру перед шифрованием

String encrypted\_text = cyclic\_shift(original\_text.toLowerCase(), true);

System.out.println("Зашифрованный текст: " + encrypted\_text);

// Расшифровываем текст с помощью циклического сдвига в другую сторону

System.out.println("Расшифрованный текст: " + cyclic\_shift(encrypted\_text, false) + '\n');

}

// Установка нового значения сдвига

public static void command3() {

System.out.println("Введите число позиций для сдвига:");

// Сохраняем старое значение сдвига для отображения

int old\_shift = shift;

// Читаем новое значение сдвига

try {

shift = Integer.parseInt(scanner.nextLine()); // Пытаемся преобразовать строку в число

System.out.printf("Число позиций для сдвига изменилось с %d на %d\n\n", old\_shift, shift);

} catch (NumberFormatException e) {

System.out.println("Ошибка! Вы ввели не число...");

}

}

// Вывод меню с текущим значением сдвига

public static void print\_menu() {

System.out.printf("""

1 - Шифрование с помощью операции XOR

2 - Шифрование с помощью циклического сдвига (Текущий сдвиг - %d)

3 - Ввести число позиций для шифрования путем циклического сдвига

Q - Выход из программы

""", shift); // Отображаем меню с текущим значением сдвига

}

// Главный цикл программы

public static void main(String[] args) {

String command;

do {

// Печатаем меню и ждем команду

print\_menu();

command = scanner.nextLine();

switch (command) {

case "1":

command1(); // Обрабатываем команду 1

break;

case "2":

command2(); // Обрабатываем команду 2

break;

case "3":

command3(); // Обрабатываем команду 3

break;

case "Q":

break; // Выход из программы

default:

System.out.println("Команда не распознана!\n"); // Обработка неверных команд

}

} while (!"Q".equals(command)); // Повторяем цикл до команды "Q"

scanner.close(); // Закрываем scanner

}

}