**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 № 1**

по дисциплине:

|  |
| --- |
| Кроссплатформенное программирование |

|  |  |
| --- | --- |
|  | Выполнили студенты группы ИСТ-321  Чаушьянц Р.В.  Оценка: ssssssssssssssss  Нариц А.А.  Оценка: ssssssssssssssss |
|  |
|  | Принял:  Заведующий кафедры СОД  Бородянский Юрий Михайлович |
|  | (подпись) |

**Работа с массивами и строками. Задача обработки палиндрома.**

**Цель работы**

Изучение общих принципов программирования в среде java. Написание приложения, осуществляющего поиск в случайно созданном тексте "слов", являющихся палиндромами, заполнение найденными "словами" массива и распечатка полученного массива на экране.

**Задание**

Необходимо создать консольное приложение, осуществляющее генерацию случайного текста в размере не менее чем 100 строк. Все "слова" текста составляются программой автоматически из случайно выбранных букв. После создания текста программа осуществляет поиск в тексте слов – палиндромов (то есть таких слов, которые одинаково читаются слева – направо и справа – налево), заполняет этими словами массив и распечатывает на экране исходный текст и найденные палиндромы. Для простоты размер всех "слов" взять одинаковым. Размер "слова" задается пользователем путем ввода с клавиатуры. Так же желательно задавать с клавиатуры число строк и число слов в строке.

**Теоретическая часть.**

Java – это объектно-ориентированный язык программирования, широко используемый для разработки приложений различного уровня сложности. Для успешного освоения Java необходимо понимать синтаксис языка, принципы объявления переменных, а также разницу между простыми (примитивными) и ссылочными типами данных. Кроме того, важно уметь работать с циклами и массивами.

**Общая структура программы**

Программа на Java состоит из классов и методов. Точка входа в программу – это метод main, который имеет следующий вид:

public class Main {

public static void main(String[] args) {

System.out.println("Hello, world!");

}

}

Здесь:

* public class Main – объявление класса Main.
* public static void main(String[] args) – главный метод, с которого начинается выполнение программы.
* System.out.println("Hello, world!"); – вывод текста в консоль.

**Объявление переменных**

В Java переменные объявляются с указанием типа данных. Примитивные типы данных включают:

* int (целые числа)
* double (числа с плавающей запятой)
* boolean (логический тип)
* char (символы)

Пример объявления переменных:

int age = 25;

double price = 99.99;

boolean isActive = true;

char grade = 'A';

Ссылочные типы данных, такие как String и массивы, хранят ссылку на объект в памяти:

String name = "Java";

int[] numbers = {1, 2, 3, 4, 5};

**Разница между примитивными и ссылочными типами**

Примитивные типы хранят конкретное значение в памяти, тогда как ссылочные типы содержат ссылку на объект в куче (heap). Например:

int a = 10;

int b = a;

b = 20; // a остается 10

А при использовании ссылочных типов:

int[] arr1 = {1, 2, 3};

int[] arr2 = arr1;

arr2[0] = 10; // arr1 также изменится

**Циклы и массивы**

Циклы позволяют повторять выполнение блока кода. В Java есть три основных типа циклов:

* for – используется, когда известное количество итераций.
* while – выполняется, пока истинно заданное условие.
* do-while – выполняется хотя бы один раз.

Пример использования for для работы с массивом:

int[] numbers = {1, 2, 3, 4, 5};

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

System.out.println(numbers[i]);

}

Альтернативный способ – цикл for-each:

for (int num : numbers) {

System.out.println(num);

}

Использование циклов и массивов в Java позволяет эффективно обрабатывать данные и автоматизировать повторяющиеся операции.

**Ход работы**

В ходе работы была написана программа, осуществляющее генерацию случайного текста в размере не менее чем 100 строк. Все "слова" текста составляются программой автоматически из случайно выбранных букв. После создания текста программа осуществляет поиск в тексте слов – палиндромов (то есть таких слов, которые одинаково читаются слева – направо и справа – налево), заполняет этими словами массив и распечатывает на экране исходный текст и найденные палиндромы (Рис.1.1, рис.1.2). Для простоты размер всех "слов" взят одинаковым (5 символов). Размер "слова", количество строк и количество слов в строке задается пользователем путем ввода с клавиатуры. Логика выполнения программы представлена на схеме 1.

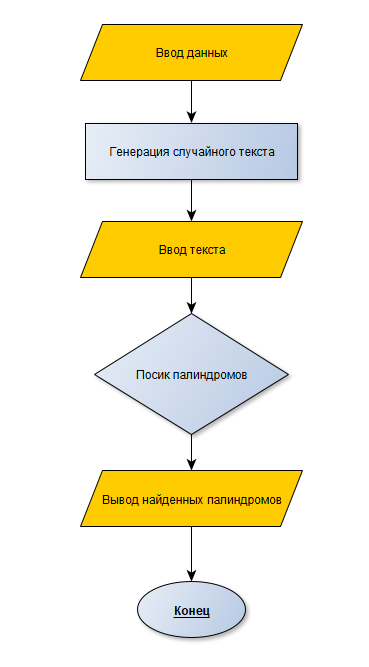


Схема 1. Логика работы программы

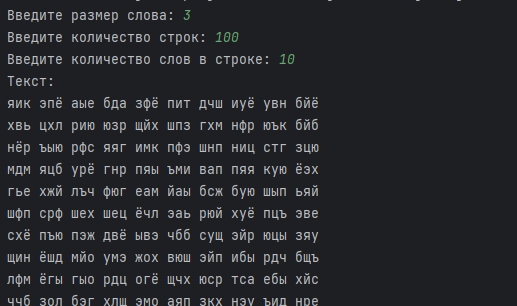
****

Рисунок 1.1 Результат выполнения программы

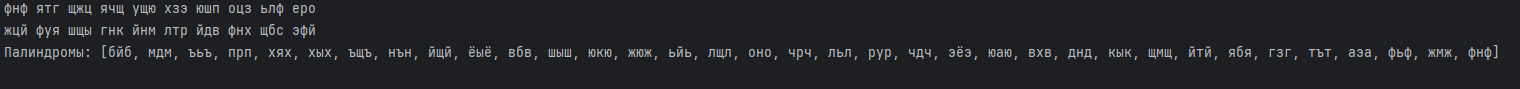
****

Рисунок 1.2 Результат выполнения программы

**Выводы**

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

1. **Работа с многомерными массивами** – мы научились создавать, заполнять и обрабатывать массивы строк (StringBuilder[][]), что полезно для организации данных в сложных структурах.
2. **Использование StringBuilder для работы со строками** – в отличие от String, StringBuilder позволяет эффективно изменять строки, что важно для оптимизации работы с текстом.
3. **Генерация случайных строк** – мы использовали Random для создания случайных слов, что применяется в тестировании, моделировании данных и генерации контента.
4. **Поиск палиндромов** – мы реализовали алгоритм проверки строк на палиндромность, что помогает в обработке текста, анализе данных и разработке алгоритмов на естественном языке.

Эти навыки пригодятся при разработке приложений, анализе данных и создании алгоритмов обработки текста.

**Приложение**

import java.util.ArrayList;

import java.util.Random;

import java.util.Scanner;

public class Main {

// Строка с алфавитом, используемая для генерации случайных символов

static final String ALPHABET = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя";

// Создаём объект Scanner для ввода с консоли

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

// Функция для поиска палиндромов в двумерном массиве текста

public static ArrayList<StringBuilder> get\_palindromes(StringBuilder[][] text) {

// Список для хранения найденных палиндромов

ArrayList<StringBuilder> palindromes = new ArrayList<>();

// Перебор всех строк в двумерном массиве

for (StringBuilder[] strings : text) {

// Перебор каждого слова в строках

for (StringBuilder string : strings) {

// Проверяем, является ли слово палиндромом

String original = string.toString(); // Сохраняем оригинальную строку

String reversed = string.reverse().toString(); // Переворачиваем строку и сохраняем её

// Если слово и его перевёрнутая версия совпадают, оно палиндром

if (original.equals(reversed)) {

palindromes.add(string); // Добавляем слово в список палиндромов

}

}

}

return palindromes; // Возвращаем список найденных палиндромов

}

// Функция для вывода текста на экран

public static void print\_text(StringBuilder[][] text) {

System.out.println("Текст: ");

// Перебор строк двумерного массива

for (StringBuilder[] strings : text) {

// Перебор слов в каждой строке

for (StringBuilder string : strings) {

System.out.print(string + " "); // Выводим каждое слово на экран

}

System.out.println(); // Переход на новую строку после вывода всех слов в строке

}

}

// Функция для генерации текста со случайными словами

public static StringBuilder[][] generate\_text(int size\_word, int count\_lines, int size\_line) {

Random random = new Random(); // Создаём объект Random для генерации случайных чисел

StringBuilder[][] text = new StringBuilder[count\_lines][size\_line]; // Двумерный массив для хранения текста

// Генерация текста

for (int i = 0; i < count\_lines; i++) { // Перебор строк

for (int j = 0; j < size\_line; j++) { // Перебор слов в строках

StringBuilder word = new StringBuilder();

// Генерация случайных слов

for (int h = 0; h < size\_word; h++) {

int randomNumber = random.nextInt(ALPHABET.length()); // Случайный индекс из алфавита

char randomChar = ALPHABET.charAt(randomNumber); // Получаем случайный символ

word.append(randomChar); // Добавляем символ к слову

}

text[i][j] = word; // Записываем слово в массив

}

}

return text; // Возвращаем двумерный массив с текстом

}

// Функция для ввода данных с проверкой с использованием try-catch

public static int getPositiveIntFromUser(String prompt) {

int number = -1;

boolean validInput = false;

// Повторяем ввод до тех пор, пока пользователь не введёт корректное число

while (!validInput) {

System.out.print(prompt); // Выводим подсказку пользователю

try {

number = Integer.parseInt(scanner.nextLine()); // Попытка считать число

if (number <= 0) {

System.out.println("Число должно быть положительным."); // Если число не положительное, выводим ошибку

} else {

validInput = true; // Ввод корректен

}

} catch (NumberFormatException e) {

System.out.println("Пожалуйста, введите корректное число."); // Если ошибка формата числа, выводим сообщение

}

}

return number; // Возвращаем корректное число

}

// Основной метод

public static void main(String[] args) {

// Ввод данных с проверкой

int size\_word = getPositiveIntFromUser("Введите размер слова: ");

int count\_lines = getPositiveIntFromUser("Введите количество строк: ");

int size\_line = getPositiveIntFromUser("Введите количество слов в строке: ");

// Генерация текста с введёнными данными

StringBuilder[][] text = generate\_text(size\_word, count\_lines, size\_line);

// Вывод текста

print\_text(text);

// Поиск палиндромов в тексте

ArrayList<StringBuilder> palindromes = get\_palindromes(text);

// Вывод найденных палиндромов

System.out.println("Палиндромы: " + palindromes);

scanner.close(); // Закрытие scanner после завершения работы с ним

}