**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

# по дисциплине «Введение в разработку программного обеспечения»

на тему: «Моделирование и алгоритмизация как средства проектирования программного обеспечения»

Выполнил: студент гр. ИТП-21

Рылушкин Б. А.

Принял: преподаватель-стажёр

Башаримов Ю.С.

Гомель 2024

Цель работы: реализовать алгоритм быстрой сортировки на языке *Java,* протестировать его и создать консольное приложение для демонстрации работы.

**Ход работы**

**Задание.** 1. Описать блок-схему алгоритма согласно варианту 1.

2. На основе описанной блок-схемы реализовать алгоритм средствами языка *Java.*

3. Протестировать разработанный алгоритм при помощи модульных тестов.

4. Создать консольное приложение для демонстрации работы алгоритма. 5. Составить отчет о проделанной работе.

Условие к заданию 1 приведено на рисунке 1.



Рисунок 1 – Условие к заданию 1

Разработанная блок-схема алгоритма приведена в приложении А.

Код программы находится в приложении Б.

Результат тестирования разработанного алгоритма приведен на рисунке 2.

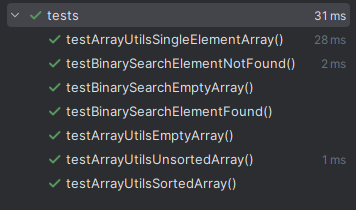


Рисунок 2 – Результат тестирования разработанного алгоритма

Результат выполнения задания 1 представлен на рисунке 3.

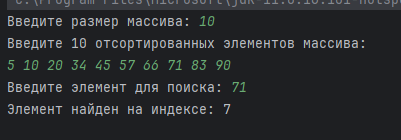


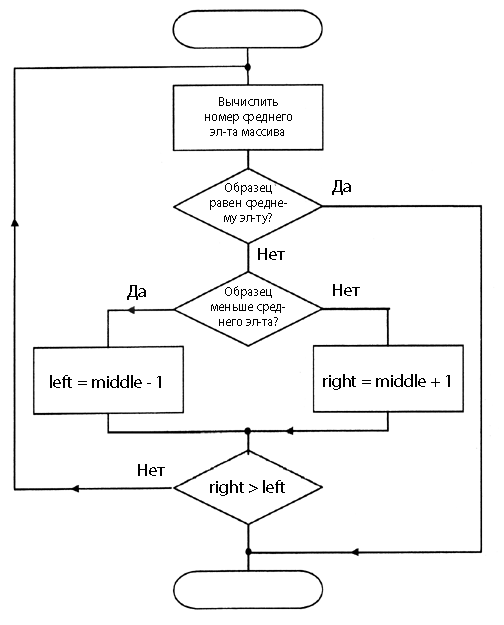
Рисунок 3 – Результат выполнения программы

Вывод: в ходе выполнения лабораторной работы мы реализовали алгоритм бинарного поиска на языке *Java,* протестировали его и создали консольное приложение для демонстрации работы.  В заключение, моделирование и алгоритмизация являются неотъемлемыми инструментами в проектировании программного обеспечения. Их применение позволяет создать более качественные, устойчивые и легко поддерживаемые системы, что, в свою очередь, отвечает требованиям современного рынка и пользователей.

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Блок-схема**



**ПРИЛОЖЕНИЕ Б**

(обязательное)

**Текст программы**

***Main.java*:**

// Main.java

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Ввод размера массива

System.out.print("Введите размер массива: ");

int size = scanner.nextInt();

int[] sortedArray = new int[size];

// Ввод элементов массива

System.out.println("Введите " + size + " отсортированных элементов массива:");

for (int i = 0; i < size; i++) {

sortedArray[i] = scanner.nextInt();

}

// Проверка, отсортирован ли массив

if (!ArrayUtils.isSorted(sortedArray)) {

System.out.println("Ошибка: массив не отсортирован.");

scanner.close();

return; // Завершение программы

}

// Ввод элемента для поиска

System.out.print("Введите элемент для поиска: ");

int target = scanner.nextInt();

// Создание экземпляра BinarySearch и поиск элемента

BinarySearch binarySearch = new BinarySearch();

int result = binarySearch.binarySearch(sortedArray, target);

if (result == -1) {

System.out.println("Элемент не найден");

} else {

System.out.println("Элемент найден на индексе: " + result);

}

scanner.close();

}

}

***Test.java*:**

// CombinedTest.java

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import static org.junit.jupiter.api.Assertions.assertTrue;

import static org.junit.jupiter.api.Assertions.assertFalse;

public class tests {

// Тесты для BinarySearch

@Test

public void testBinarySearchElementFound() {

BinarySearch binarySearch = new BinarySearch();

int[] array = {2, 3, 4, 10, 40};

int target = 10;

int result = binarySearch.binarySearch(array, target);

assertEquals(3, result); // Индекс элемента 10

}

@Test

public void testBinarySearchElementNotFound() {

BinarySearch binarySearch = new BinarySearch();

int[] array = {2, 3, 4, 10, 40};

int target = 5;

int result = binarySearch.binarySearch(array, target);

assertEquals(-1, result); // Элемент не найден

}

@Test

public void testBinarySearchEmptyArray() {

BinarySearch binarySearch = new BinarySearch();

int[] array = {};

int target = 1;

int result = binarySearch.binarySearch(array, target);

assertEquals(-1, result); // Элемент не найден

}

// Тесты для ArrayUtils

@Test

public void testArrayUtilsSortedArray() {

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

assertTrue(ArrayUtils.isSorted(sortedArray)); // Массив отсортирован

}

@Test

public void testArrayUtilsUnsortedArray() {

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

assertFalse(ArrayUtils.isSorted(unsortedArray)); // Массив не отсортирован

}

@Test

public void testArrayUtilsSingleElementArray() {

int[] singleElementArray = {1};

assertTrue(ArrayUtils.isSorted(singleElementArray)); // Массив отсортирован

}

@Test

public void testArrayUtilsEmptyArray() {

int[] emptyArray = {};

assertTrue(ArrayUtils.isSorted(emptyArray)); // Пустой массив считается отсортированным

}

}

***BinarySearch.java:***

public class BinarySearch {

public int binarySearch(int[] array, int target) {

int left = 0;

int right = array.length - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (array[mid] == target) {

return mid; // Элемент найден

}

if (array[mid] < target) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return -1; // Элемент не найден

}

}