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 «Технологии разработки программного обеспечения»

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

Выполнил: студент гр. ИТИ-21

Щербенко М.А.

Принял: преподаватель

Гуменников Е. Д.

Гомель 2023

**Цель работы:** ознакомиться с моделированием и алгоритмизацией как средствами проектирования программного обеспечения.

**Практическое задание**

**Задание.**

1. Описать блок-схему алгоритма согласно варианта (таблица 3).
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.

Таблица 3

|  |  |
| --- | --- |
| **Вариант** | Условие задачи |
| 8 | Сортировка выбором |

**Ход работы**

**Порядок выполнения задания.**

На рисунке 1 показана блок-схема алгоритма.

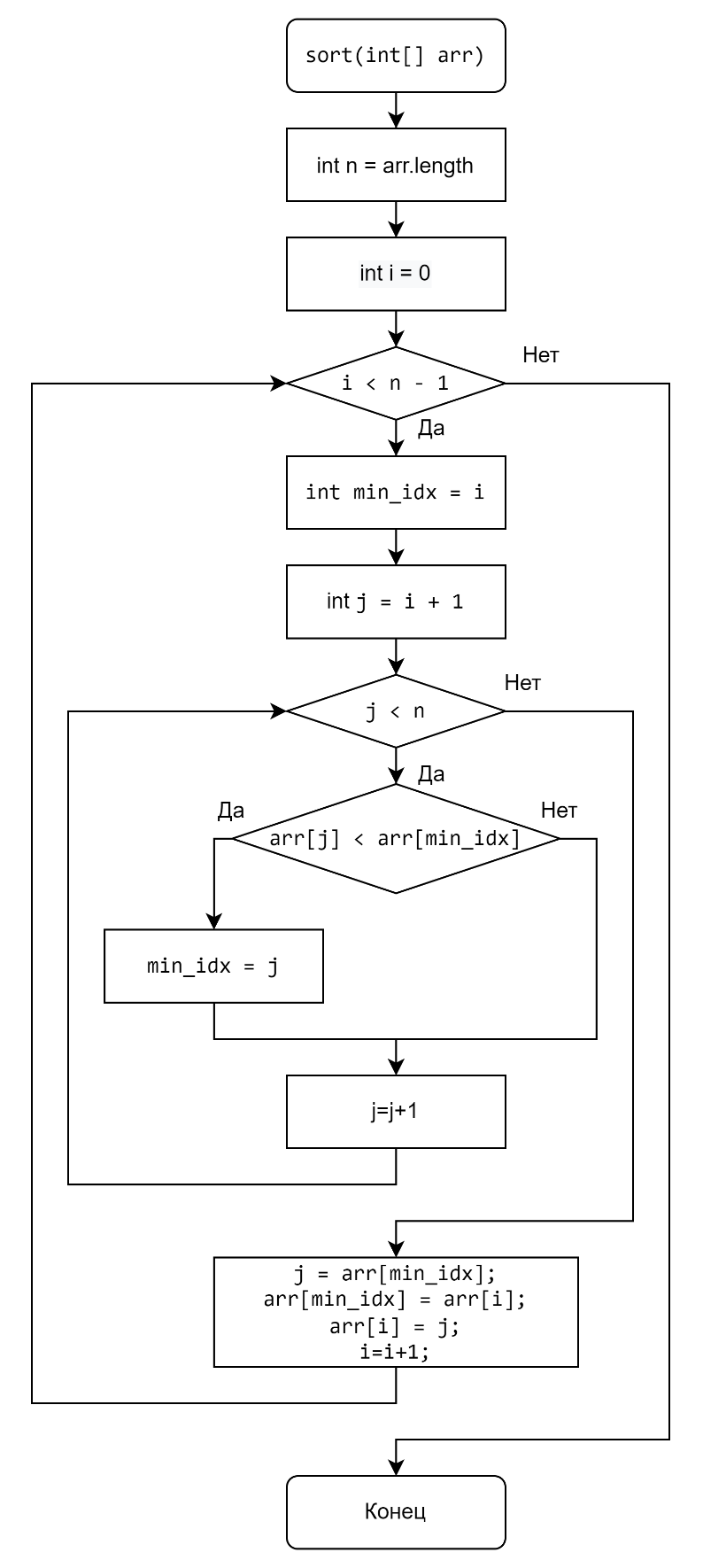


Рисунок 1 – Блок-схема алгоритма

На рисунке 2 показано выполнение программы.

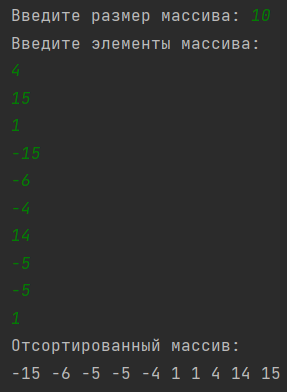


Рисунок 2 – Выполнение программы программы

На рисунке 3 показан результат выполнения модульных тестов.

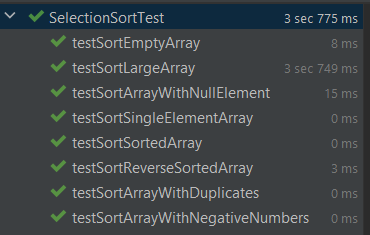


Рисунок 3 – Выполнение модульных тестов

**Вывод:** в процессе работы ознакомились с моделированием и алгоритмизацией как средствами проектирования программного обеспечения.

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Текст программы**

**Класс SelectionSort:**

import java.util.Scanner;  
  
public class SelectionSort {  
 public SelectionSort() {  
 }  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.print("Введите размер массива: ");  
 int size = scanner.nextInt();  
 int[] arr = new int[size];  
 System.*out*.println("Введите элементы массива:");  
  
 for(int i = 0; i < arr.length; ++i) {  
 arr[i] = scanner.nextInt();  
 }  
  
 *sort*(arr);  
 System.*out*.println("Отсортированный массив: ");  
 int[] var8 = arr;  
 int var5 = arr.length;  
  
 for(int var6 = 0; var6 < var5; ++var6) {  
 int i = var8[var6];  
 System.*out*.print("" + i + " ");  
 }  
  
 }  
  
 public static void sort(int[] arr) {  
 int n = arr.length;  
  
 for(int i = 0; i < n - 1; ++i) {  
 int min\_idx = i;  
  
 int j;  
 for(j = i + 1; j < n; ++j) {  
 if (arr[j] < arr[min\_idx]) {  
 min\_idx = j;  
 }  
 }  
  
 j = arr[min\_idx];  
 arr[min\_idx] = arr[i];  
 arr[i] = j;  
 }  
  
 }  
}

**Класс SelectionSortTest:**

import java.util.Arrays;  
import org.junit.Assert;  
import org.junit.Test;  
public class SelectionSortTest {  
 public SelectionSortTest() {  
 }  
  
 @Test  
 public void testSortEmptyArray() {  
 int[] arr = new int[0];  
 SelectionSort.*sort*(arr);  
 int[] expectedArr = new int[0];  
 Assert.*assertArrayEquals*(expectedArr, arr);  
 }  
  
 @Test  
 public void testSortSingleElementArray() {  
 int[] arr = new int[]{1};  
 SelectionSort.*sort*(arr);  
 int[] expectedArr = new int[]{1};  
 Assert.*assertArrayEquals*(expectedArr, arr);  
 }  
  
 @Test  
 public void testSortSortedArray() {  
 int[] arr = new int[]{1, 2, 3, 4, 5};  
 SelectionSort.*sort*(arr);  
 int[] expectedArr = new int[]{1, 2, 3, 4, 5};  
 Assert.*assertArrayEquals*(expectedArr, arr);  
 }  
  
 @Test  
 public void testSortReverseSortedArray() {  
 int[] arr = new int[]{5, 4, 3, 2, 1};  
 SelectionSort.*sort*(arr);  
 int[] expectedArr = new int[]{1, 2, 3, 4, 5};  
 Assert.*assertArrayEquals*(expectedArr, arr);  
 }  
  
 @Test  
 public void testSortArrayWithDuplicates() {  
 int[] arr = new int[]{5, 1, 3, 2, 5};  
 SelectionSort.*sort*(arr);  
 int[] expectedArr = new int[]{1, 2, 3, 5, 5};  
 Assert.*assertArrayEquals*(expectedArr, arr);  
 }  
  
 @Test  
 public void testSortLargeArray() {  
 int[] arr = new int[100000];  
  
 int i;  
 for(i = 0; i < arr.length; ++i) {  
 arr[i] = (int)(Math.*random*() \* 100000.0);  
 }  
  
 SelectionSort.*sort*(arr);  
  
 for(i = 0; i < arr.length - 1; ++i) {  
 Assert.*assertTrue*(arr[i] <= arr[i + 1]);  
 }  
  
 }  
  
 @Test  
 public void testSortArrayWithNegativeNumbers() {  
 int[] arr = new int[]{-5, -1, -3, -2, -4};  
 SelectionSort.*sort*(arr);  
 int[] expectedArr = new int[]{-5, -4, -3, -2, -1};  
 Assert.*assertArrayEquals*(expectedArr, arr);  
 }  
  
 @Test(  
 expected = NullPointerException.class  
 )  
 public void testSortArrayWithNullElement() {  
 Integer[] arr = new Integer[]{1, 2, 3, null, 4};  
 int[] intArr = Arrays.*stream*(arr).mapToInt(Integer::intValue).toArray();  
 SelectionSort.*sort*(intArr);  
 int[] expectedArr = new int[]{1, 2, 3, 4, 0};  
 Assert.*assertArrayEquals*(expectedArr, intArr);  
 }  
}