

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт (технический университет)»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| УГНС | | 09.00.00 | Информатика и вычислительная техника | | |
| Направление подготовки | | 09.03.01 | Информатика и вычислительная техника | | |
| Направленность (профиль) | |  | Системы автоматизированного проектирования | | |
| Форма обучения | |  | очная | | |
|  | |  |  | | |
| Факультет | |  | Информационных технологий и управления | | |
| Кафедра | |  | Систем автоматизированного проектирования и управления | | |
| Учебная дисциплина | |  | Разработка программных систем | | |
| Курс | 2 | | | Группа | 423 |

Отчёт по лабораторной работе №2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнитель: |  |  |  |  |
| обучающийся группы 423 |  |  |  | Привалов Никита Николаевич  Алиева Анастасия Андреевна |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |
| Проверил: |  |  |  | Соболевский Владислав Алексеевич |
|  |  | (дата, подпись) |  |  |
|  |  |  |  |  |

СОДЕРЖАНИЕ

[1 Постановка задачи 3](#_Toc151946343)

[2 Исходные данные 3](#_Toc151946344)

[3 Схема алгоритма решения задачи 3](#_Toc151946345)

[4 Описание хода выполнения 4](#_Toc151946346)

[5 Результаты работы программы 4](#_Toc151946347)

[6 Расчёт лучшей и худшей сложности реализованного алгоритма 5](#_Toc151946348)

[7 Выводы по заданию 7](#_Toc151946349)

[8 Исходный код полученного программного решения 7](#_Toc151946350)

## 1 Постановка задачи

Разработать программы сортировки согласно варианту (гномья сортировка).

## 2 Исходные данные

Исходные данные состоят из массива

## 3 Схема алгоритма решения задачи

На рисунке 1 представлена блок-схема алгоритма решения задачи.



Рисунок 1 – Блок-схема алгоритма решения задачи

## 4 Описание хода выполнения

Для выполнения задачи мы сначала изучили принцип работы гномьей сортировки. Затем мы приступили к написанию кода для этой сортировки. Мы распределили обязанности между членами группы, чтобы каждый работал над своим участком кода. После этого мы объединили эти участки в один проект и провели тестирование. После успешного тестирования мы создали блок-схему для нашего алгоритма и оценили его сложность. Наконец, мы подготовили отчет, в котором представили результаты нашей работы.

## 5 Результаты работы программы

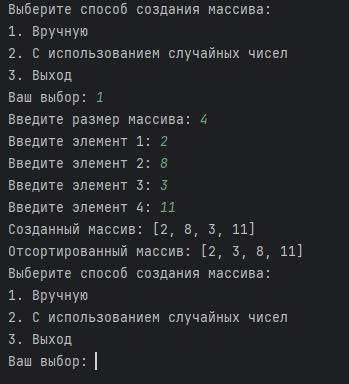


Рисунок 2 – Экранная копия первого тестирования работы разработанной программы

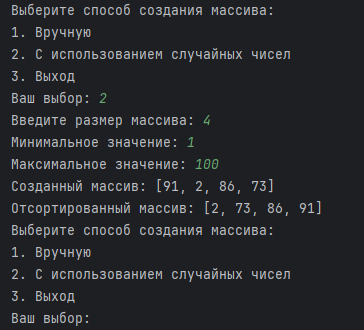


Рисунок 3 – Экранная копия второго тестирования работы разработанной программы

## 6 Расчёт лучшей и худшей сложности реализованного алгоритма

Оценка временной сложности гномьей сортировки.

Лучший случай для алгоритма гномьей сортировки возникает, когда массив уже отсортирован, и не требуется выполнение обменов. В таком случае, время выполнения будет линейным и равным O(n), где "n" - количество элементов в массиве.

Сложность алгоритма гномьей сортировки зависит от конкретной реализации и данных, над которыми производится сортировка. Однако в среднем случае сложность оценивается как O(n^2), где "n" - это количество элементов в массиве.

Средняя временная сложность Gnome Sort при сортировке случайных данных составляет O(n^2). Это объясняется тем, что в худшем случае (если массив отсортирован в обратном порядке), гномья сортировка выполняет множество обменов, что приводит к квадратичной сложности.

## 7 Выводы по заданию

В ходе выполнения задания были изучены различные методы сортировок, получены практические навыки по реализации метода гномьей сортировки и работе с системой контроля версий «Гитхаб».

## 8 Исходный код полученного программного решения

[Начало файла main.py]

import random  
  
  
def create\_array\_manually():  
 n = int(input("Введите размер массива: "))  
 arr = []  
 for i in range(n):  
 value = int(input(f"Введите элемент {i + 1}: "))  
 arr.append(value)  
 return arr  
  
  
def create\_array\_random():  
 n = int(input("Введите размер массива: "))  
 min\_value = int(input("Минимальное значение: "))  
 max\_value = int(input("Максимальное значение: "))  
 arr = [random.randint(min\_value, max\_value) for \_ in range(n)]  
 return arr  
  
  
def gnome\_sort(arr):  
 index = 0  
 while index < len(arr):  
 if index == 0:  
 index = 1  
 if arr[index] >= arr[index - 1]:  
 index += 1  
 else:  
 arr[index], arr[index - 1] = arr[index - 1], arr[index]  
 index -= 1  
 return arr  
  
  
while True:  
 print("Выберите способ создания массива:")  
 print("1. Вручную")  
 print("2. С использованием случайных чисел")  
 print("3. Выход")  
  
 choice = input("Ваш выбор: ")  
  
 if choice == '1':  
 manual\_array = create\_array\_manually()  
 print("Созданный массив:", manual\_array)  
 if len(manual\_array) == 1:  
 print("Отсортированный массив:", manual\_array)  
 else:  
 print("Отсортированный массив:", gnome\_sort(manual\_array))  
 elif choice == '2':  
 random\_array = create\_array\_random()  
 print("Созданный массив:", random\_array)  
 if len(random\_array) == 1:  
 print("Отсортированный массив:", random\_array)  
 else:  
 print("Отсортированный массив:", gnome\_sort(random\_array))  
 elif choice == '3':  
 break  
 else:  
 print("Неверный выбор. Пожалуйста, выберите 1, 2 или 3.")

[Конец файла main.py]