**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по учебной практике**

Тема: Пирамидальная сортировка

|  |  |  |
| --- | --- | --- |
| Студент гр. 6382 |  | Костин С.В. |
| Студентка гр. 6382 |  | Черкасова Е.И. |
| Студентка гр. 6383 |  | Терещенко В.Н. |
| Руководитель |  | Фирсов М.А. |

Санкт-Петербург

2018

**ЗАДАНИЕ**

**на учебную практику**

Студент Костин С.В. группы 6382

Студентка Черкасова Е.И. группы 6382

Студентка Терещенко В.Н. группы 6383

Тема практики: Пирамидальная сортировка

Задание на практику:

Командная итеративная разработка визуализатора алгоритма(ов) на Java с графическим интерфейсом.

Алгоритм: Пирамидальная сортировка.

Сроки прохождения практики: 27.06.2018 – 10.07.2018

Дата сдачи отчета: 09.07.2018

Дата защиты отчета: 09.07.2018

|  |  |  |
| --- | --- | --- |
|  | | |
| Студент |  | Костин С.В. |
| Студентка |  | Черкасова Е.И. |
| Студентка |  | Терещенко В.Н. |
| Руководитель |  | Фирсов М.А. |

**Аннотация**

В данной работе планируется изучить язык java и закрепить знания в ходе написания проектной работы, темой которой является «Пирамидальная сортировка». В ходе написания проекта должен быть написан алгоритм сортировки с визуализацией и создан пользовательский интерфейс.

**содержание**

Можно попробовать сделать автоматическое содержание. Ссылки -> Оглавление –> Автособираемое. Для этого, возможно, придется настроить стили всех заголовков через панель «Стили» («Заголовок 1», «Заголовок 2» и т.д.)!

|  |  |  |
| --- | --- | --- |
|  | Введение | 4 |
| 1. | Требования к программе | 5 |
| 1.1. | Исходные требования к программе\* | 0 |
| 1.2. | Уточнение требований после сдачи прототипа | 0 |
| 1.3.  1.4 | Уточнение требований после сдачи 1-ой версии  Уточнение требований после сдачи 2-ой версии | 0  0 |
| 2. | План разработки и распределение ролей в бригаде | 0 |
| 2.1. | План разработки | 0 |
| 2.2. | Распределение ролей в бригаде | 0 |
| 3. | Особенности реализации | 0 |
| 3.1. | Использованные структуры данных | 0 |
| 3.2. | Основные методы | 0 |
| 3.3 |  | 0 |
| 4. | Тестирование | 0 |
| 4.1 | Тестирование графического интерфейса | 0 |
| 4.2 | Тестирование кода алгоритма | 0 |
| 4.3 | … | 0 |
|  | Заключение | 0 |
|  | Список использованных источников | 0 |
|  | Приложение А. Название приложения | 0 |

**введение**

Цель проектной работы: познакомиться и изучить объектно-ориентированный язык программирования Java; параллельно со знакомством, для закрепления полученных знаний, разработать программу, осуществляющую алгоритм пирамидальной сортировки, с ее визуализацией и созданием пользовательского интерфейса для более удобного использования программы.

Задачи:

1. Написать алгоритм пирамидальной сортировки.
2. Визуализировать алгоритм.
3. Разработать пользовательский интерфейс.
4. Протестировать работу программы.
5. Получить опыт командной работы.

**1. требования к программе**

**1.1. Исходные Требования к программе**

Входные данные: на вход подается строка, состоящая из целых чисел, разделенных пробелами. Ввод данных осуществляется из файла (input.dat).

Выходные данные: на экран выводится бинарное дерево (на каждой итерации).

Интерфейс приложения. До начала работы алгоритма на экран выводится бинарное дерево, соответствующее массиву из элементов исходной строки, и кнопка “Next step”. В процессе работы программы строится бинарное дерево, соответствующее каждому шагу сортировки (построение каждого следующего изображения осуществляется после нажатия пользователем на кнопку “Next step”). На финальном шаге работы с программой помимо дерева выводится сообщение, объявляющее успешное окончание работы.

Алгоритм сортировки прототипа реализован на основе массива. Визуализация бинарного дерева (частичная ее реализация) основана на переопределении функций графической библиотеки Swing.

**1.2. Уточнение требований после сдачи прототипа**

Алгоритм изменен, теперь пирамидальная сортировка реализована на дереве. Визуализация дерева адаптирована под библиотеку GraphStream.

**2. План разработки и распределение ролей в бригаде**

**2.1. План разработки**

В ходе создания проекта не было написано прототипа программы в обычном его понимании, так как все части проекта разрабатывались и дорабатывались на протяжении всей работы над проектом. Объединение последних версий всех составляющих проекта были объединены в итоговую программу 08.07.

**2.2. Распределение ролей в бригаде**

Костин С.В. ответственный за интерфейс.

Черкасова Е.И. – за реализацию алгоритма.

Терещенко В.Н. – тестирование.

1. **ОСОБЕННОСТИ РЕАЛИЗАЦИИ**
   1. **Использованные структуры данных**

Если вдруг у вас будут рисунки (например, скрины GUI), то названия рисунков должны быть с автоматической нумерацией: правой кнопкой по картинке -> Вставить название. Возможно, придется отредактировать стиль «Названия объектов» на панели «Стили».

* 1. **Основные методы**

1. **Тестирование**
   1. **Тестирование кода алгоритма**
   2. **Тестирование графического интерфейса**

В каждом подпункте нужно указать:

* Реализованные тесты.
* Обнаруженные ошибки.
* Как ошибки были исправлены.
* Что показало повторное тестирование (снова ошибки или успешно пройденные тесты).

Обязательно опишите, какой системой контроля версий пользуетесь! Предоставьте рабочую ссылку на ваш репозиторий со всем исходным кодом.

**заключение**

В ходе выполнения проектной работы были получены знания о языке Java, необходимые для создания программы, были изучены: библиотека для создания графического интерфейса Swing при создании визуализации прототипа и пользовательского интерфейса, библиотека динамического графа GraphStream для построения бинарного дерева, класс для выполнения основных числовых операций Math и другие сопутствующие созданию проекта библиотеки. Проект был протестирован, (что-то было найдено, не найдено, как исправлено). Вышеперечисленная информация свидетельствует о выполнении поставленных задач.

**список использованных источников**

1. https://prog-cpp.ru/sort-pyramid
2. https://goo.gl/MqoTis
3. http://graphstream-project.org

Нельзя просто url-ссылки! Надо оформить соответствующим образом. Вот примеры: https://bio.spbu.ru/science/scienceinfo/el\_resourse.php

**приложение А**

**Полный код программы**

**--класс.class—**

Как только вы добавите рабочую ссылку на гит, в приложение можно будет вынести до 10 страниц кода наиболее важного на ваш взгляд класса/модуля/и т.д.