**Пояснительная записка [JINT] (FOCUSSTART-20758)**

**Программа для сортировки файлов слиянием (merge sort)**

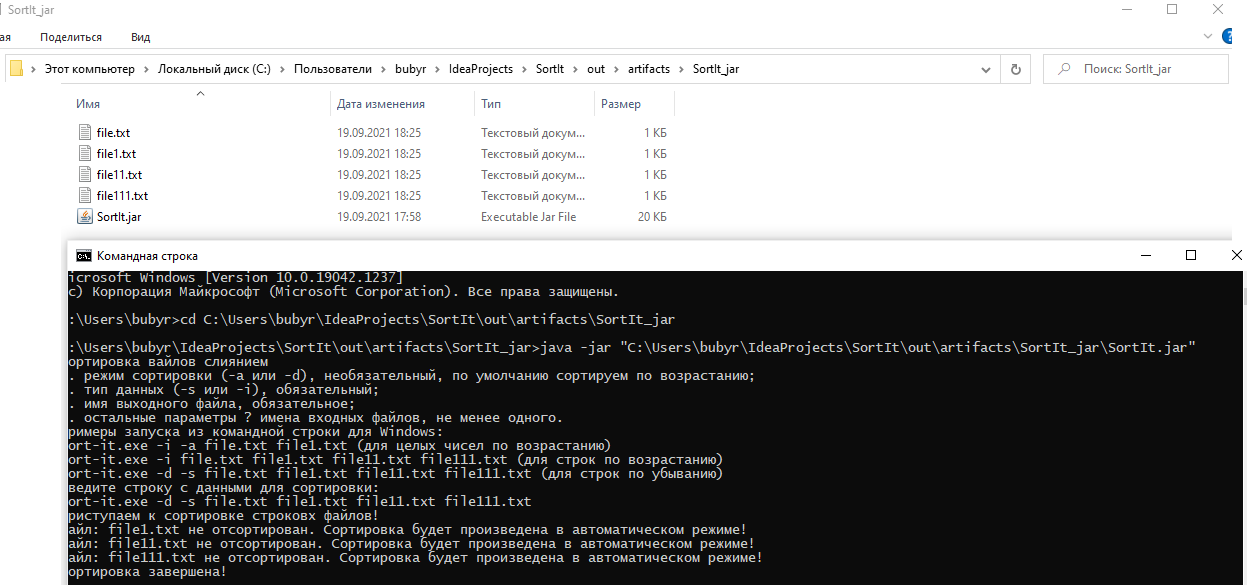
**Язык программирования - java**

**Запуск программы**

Запуск программы возможен как из командной строки windows (запускаем jar файл), так и из IDE.

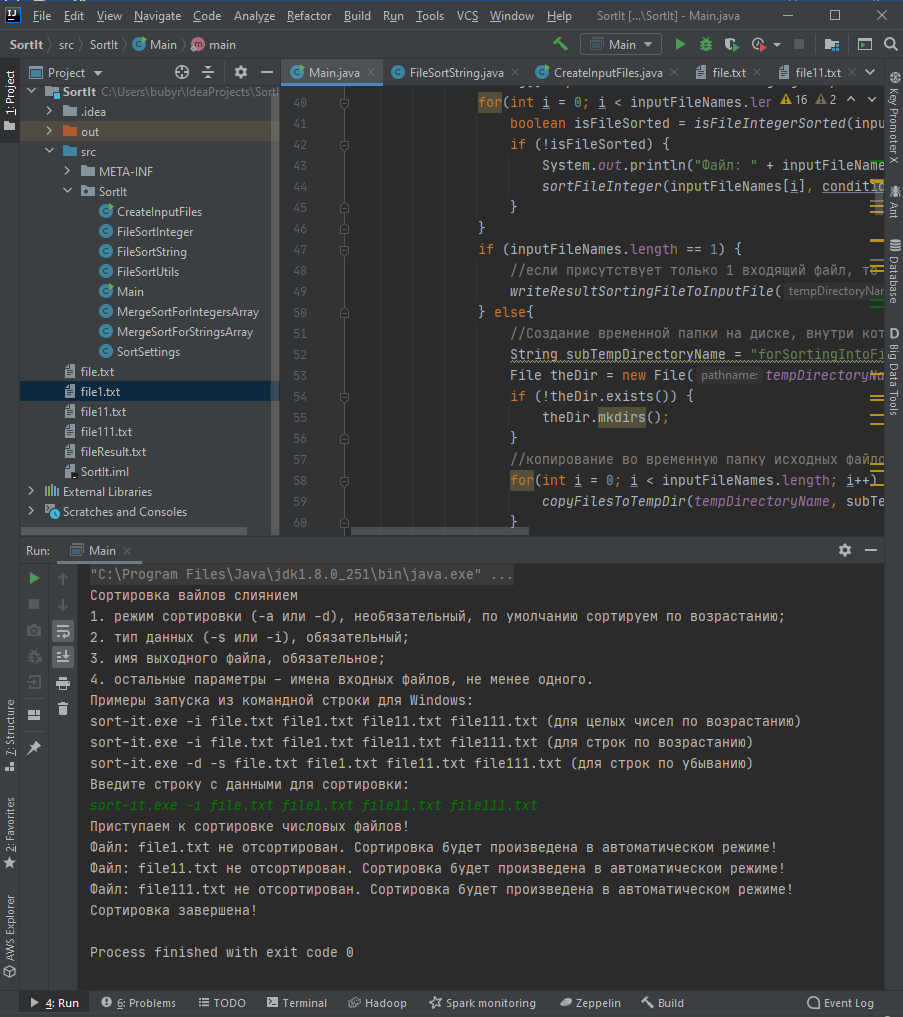
**Вариант 1. Jar файл.**

В этом случае сортируемые файлы должны находиться в одной папке с файлом jar. Также в самой командной строке нужно перейти в папку с файлом jar.



**Вариант 2. IDE.**

В этом случае сортируемые файлы должны находиться в папке проекта



**Структура программы.**

Основные классы:

**FileSortInteger**

Класс содержит методы для проверки числового файла на отсортированность и сортировки не отсортированного числового файла.

Объемность методов класса вызвана необходимостью предусмотреть работу с файлами, целиком непомещающимися в оперативную память, в таком случае происходит:

* -разделение большого файла на много небольших;
* -сортировка содержимого небольших файлов;
* -cсортировка слиянием небольших файлов обратно в общий файл.

**FileSortString**

Класс содержит методы для проверки строкового файла на отсортированность и сортировку не отсортированного строкового файла.

Объемность методов вызвана необходимостью предусмотреть работу с файлами, целиком непомещающимися в оперативную память, в таком случае происходит:

* разделение большого файла на много небольших;
* сортировка содержимого небольших файлов;
* cсортировка слиянием небольших файлов обратно в общий файл.

**FileSortUtils**

класс содержит служебные методы для сортировки файлов и сортировки содержимого файлов.

**Main**

Основной класс программы, содержит методы для приема, валидации, обработки пользовательского запроса, вызова методов остальных классов.

**MergeSortForIntegersArray**

Класс содержит методы для сортировки числового массива данных, целиком помещающихся в оперативную память.

**MergeSortForStringsArray**

Класс содержит методы для сортировки строкового массива данных, целиком помещающегося в оперативную память.

**SortSettings**

Экземпляр класса во время работы программы, хранит исходные данные для сортировки, режимы сортировки, название файлов.

**Процесс работы программы**

При начале работы программы вызывается метод main класса Main.

Далее, после вывода на экран общей информации запускается метод takeStringFromUserAndCreateSortSettings класса Main, этот метод служит для считывания у пользователя строки, и получения из строки данных для сортировки. Внутри данного метода запускается метод validateString, который производит валидацию принимаемой от пользователя строки и защищает основную программу от большинства ошибок. Методом validateString проверяются:

* наличие в строке обязательных параметров;
* соответствие имени файла-результата виду ”xxx.yyy”;
* наличие в папке проекта исходных файлов с именами, переданными пользователем.

После получения от пользователя корректной строки метод takeStringFromUserAndCreateSortSettings создает экземпляр класса SortSettings, хранящий исходные данные для сортировки, режимы сортировки, название файлов.

Далее метод main запускает проверку входящих файлов на отсортированность, проверку осуществляют методы:

* для числовых файлов isFileIntegerSorted класса FileSortInteger, данный метод возвращает значение false только в случае неотсортированных данных, но и в случае некорректных значений внутри файла;
* для строковых файлов isFileStringSorted класса FileSortString.

Если содержимое файлов не отсортировано, то на экран выводится сообщение, и начинается сортировка файлов, которую производят методы:

* для числовых файлов sortFileInteger класса FileSortInteger;
* для строковых файлов sortFileString класса FileSortString.

Данные методы не только сортируют содержимое файлов, но и убирают ошибочные данные (пустые строки для строкового и числового файлов и буквенные/символьные значения для числового файла).

Далее происходит проверка количества входящих файлов, если для сортировки пользователь передал 1 файл, то вызывается метод writeResultSortingFileToInputFile класса FileSortUtils. Этот метод производит перепись содержимого исходного файла (при необходимости уже отсортированного предыдущими методами) в файл-результат.

Если же для сортировки пользователь передал более 1 файла, то происходит следующее:

* метод copyFilesToTempDir класса FileSortUtils копирует файлы во временную папку;
* методы mergeSortFilesInteger/ mergeSortFilesString классов FileSortInteger/FileSortString объединяют сортируют слиянием файлы во временной папке;
* метод writeResultSortingFileToInputFile класса FileSortUtils переписывает результат сортировки в переданный пользователем файл-результат;
* метод deleteDir класса FileSortUtils удаляет временную папку.

**Методы sortFileInteger/sortFilesString**

Методы sortFilesInteger/ sortFilesString классов FileSortInteger/FileSortString схожи и осуществляют сортировку слиянием числовых/строчных файлов. Также данные методы с помощью методов getIntegerArrayFromStringList и getStringArrayFromStringList класса FileSortUtils, убирают из файлов ошибочные данные, а именно пустые строки и нечисловые значения (для файлов, заявленных пользователем как числовые). Коротко разберем работу данных методов:

* считывание строк из файла-источника и разделение большого содержимого файла на много небольших массивов. Осуществляется внутри самих методов на файлы с фиксированным количеством строк (в данный момент это 2000, но может быть изменено в будущем). Валидация строк осуществляется с помощью методов методов getIntegerArrayFromStringList и getStringArrayFromStringList класса FileSortUtils;
* -сортировка массивов. Осуществляется методами mergeSortIntegers класса MergeSortForIntegersArray и mergeSortStrings классов MergeSortForIntegersArray и MergeSortForStringsArray;
* Запись массивов во временныефайлы во временной папке. Осуществляется методом writeOut класса FileSortUtils;
* -cсортировка слиянием небольших файлов обратно в общий файл. Осуществляется методами mergeSortFilesInteger и mergeSortFilesString.
* Передача отсортированного файла обратно пользователю методом writeResultSortingFileToInputFile класса FileSortUtils.

Если же в методы сортировки sortFilesInteger/sortFilesString передается меньший файл, целиком помещающийся в выбранный формат в 2000 строк, то его сортировка производится более коротким образом:

* Считывание строк из файла-источника и их валидация, затем соединение в массивы. Валидация строк осуществляется с помощью методов getIntegerArrayFromStringList и getStringArrayFromStringList класса FileSortUtils;
* -сортировка массивов. Осуществляется методами mergeSortIntegers класса MergeSortForIntegersArray и mergeSortStrings классов MergeSortForIntegersArray и MergeSortForStringsArray;
* Передача отсортированного файла обратно пользователю методом writeResultSortingArrayToInputFile класса FileSortUtils.