Описание процесса компиляции:

Запустить командную строку, установить путь поиска исполняемых файлов в директорию jdk\bin (пример: path C:\Program Files\Java\jdk-9.0.1\bin).

Указать путь к папке src (пример: cd /d D:\Java projects\infosystems\src).

Скопировать данные из файла “Команда для сборки javac.txt” и применить в командной строке.

Минимальная версия java – 7. Уже скомпилированные jdk-9.0.1.

Тестирование.

Запуск всех подготовленных тестов производится путем запуска jar файла, Solution.jar. Результат работы программы выводится в консоль.

Легкие задачи.

1) Проверка на неотсортированном массиве. Проверка на массиве с большим количеством повторяющихся максимумов (программа не должна вывести максимум). Проверка на массиве из двух элементов (программа не должна выбросить исключение ArrayIndexOutOfBoundsException). Если 2 по значению числа не существует выводится минимальное значение типа long.
2) Проверка работы на числах которые легко сосчитать, на простом числе, на числе 2, на числах меньше 2 (при реализации было решено бросать исключение, поэтому оно обрабатывается с помощью блока try catch)
3) Проверка разных слов, предложения-палиндрома. Проверка пустой строки, 1 символа.
4) Проверяется программа при малых значениях, при которых можно убедиться в правильности работы. Тест при больших входных значениях опущен так как производит слишком большой вывод в консоль.
5) Проверка при присутствии искомой подстроки, при отсутствии, при похожей подстроке, при полном совпадении, пустой строки, при 1 символе в строке.

Средние задачи.

1) Аналогично первой легкой задаче, а также при длине массива равному n с неповторяющимися элементами. Если n-го по значению числа не существует выводится минимальное значение типа long.
2) Проверки правильности работы, на пустую строку, 1 символ.
3) Аналогично предыдущему плюс проверка неизменности строки после кодирования и декодирования.
4) Проверяется программа при малых значениях, при которых можно убедиться в правильности работы, при отрицательном значении, при отсутствии разменов.

Сложная задача.

Утилита вызывается, в аргументах передается имя файла в исполняемой директории (при обращении напрямую к класс файлу в директории src). Для тестирования создан файл hardText.txt. После выполнения программы в той же директории создается файл “имя входного файла”_output.txt (если имя занято, то “имя входного файла”_output_i.txt где i – номер успешной попытки создания) с рассчитанными значениями.

Проверка осуществляется с примером входных значений в файле с заданием, и работы функций с 0 значениями и др.

Слабые места в решении сложной задачи:

Функция Аккермана переполняет установленный по умолчанию стек памяти (бросает исключение java.lang.StackOverflowError) при аргументах 5 0, 4 4 и выше.

Программа работает в однопоточном режиме, возможна реализация многопоточного вычисления, что увеличит скорость выполнения (условие

позволяющее изменять порядок строк в выходном файле упрощает реализацию многопоточности).

Реализация требований.

Программа может обрабатывать большие файлы, так как файл не загружается полностью, а постепенно, через буфер.

Расширения списка функций производится путем добавления новой функции в класс Functions и конструкцию switch - case в методе functionSwitch-case COMMAND.

Изменения типа данных операндов производится в методе functionSwitch, путем извелечения типа из массива строк аргументов.