Вариант 10

Разработать программу, которая меняет на обратный порядок следования символов **каждого слова** в ASCII–строке символов. Порядок слов остается неизменным. Слова состоят только из букв. Разделителями слов являются все прочие символы. Преобразование осуществляется внутри введенной строки.

Отчёт программы на 4–5 баллов:

- Приведено решение программы на ассемблере. Программа из файла читает данные. Результаты записываются в другой файл.

- Все изменяемые параметры программы вводятся с графического диалогового окна (так как программа была сразу сделана на целевую оценку 10).

- Обработка данных, полученных из файла сформирована в виде отдельной подпрограммы. load_text.asm читает текст из файла, reverse_text.asm обрабатывает текст, output_text.asm записывает данные в файл.

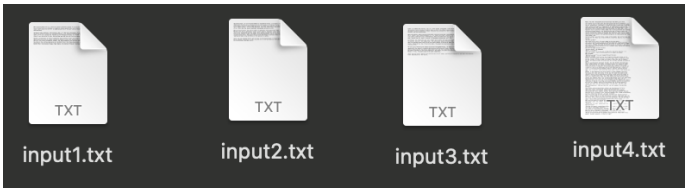
- В подкаталоге данных присутствуют файлы, используемые для тестирования: input1.txt, input2.txt, input3.txt, input4.txt.

- Для чтения текста из файла реализован буфер ограниченного размера, равного 512 байтам. При этом программа читает файлы размером до 10 килобайт. (так как программа разрабатывалась сразу на целевую оценку 10).

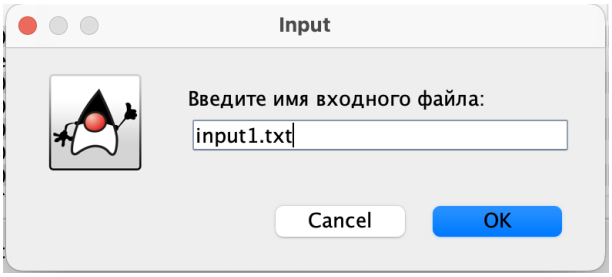
- При чтении файла размером, превышающим размер буфера, не происходит падения программы. Программа корректно обрабатывает введенный «урезанный» текст.

ТЕСТОВОЕ ПОКРЫТИЕ:

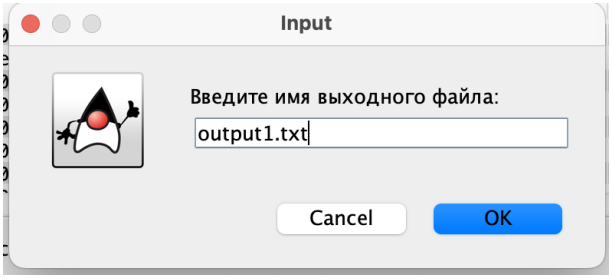
Файлы для тестирования (входные данные):



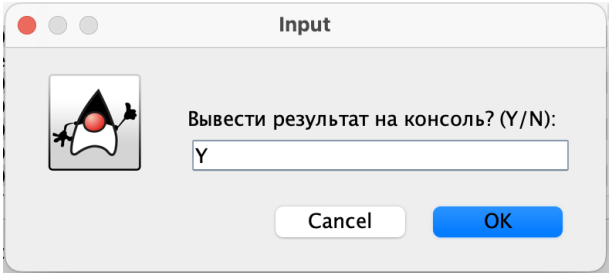
Как выглядит работа программмы?
Ввод имени входного файла:



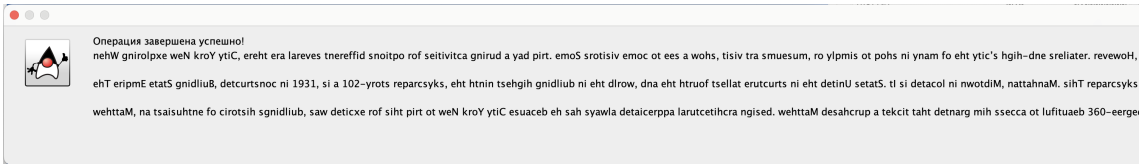
Ввод имени выходного файла:



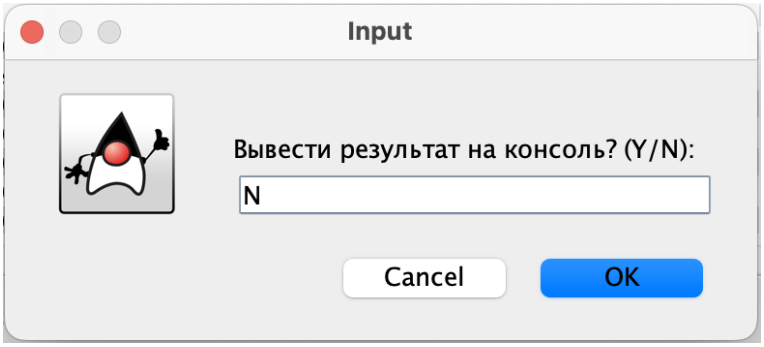
Хотим ли вывести результат на экран?:

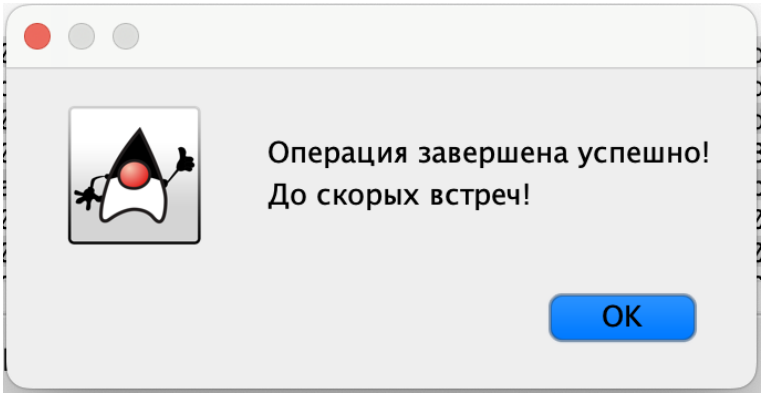


Результат на экране:



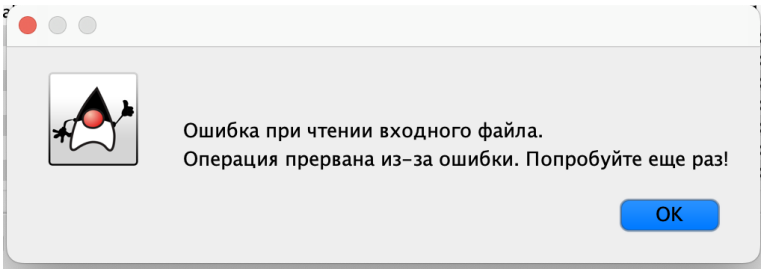
Если отказаться от вывода на экран:



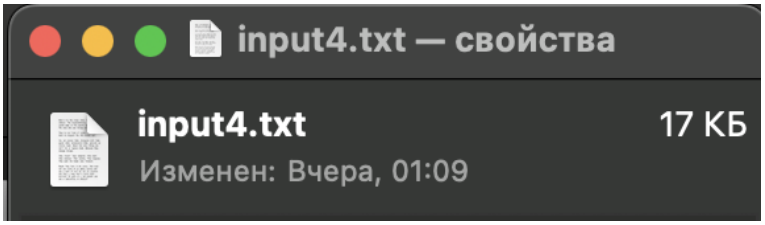


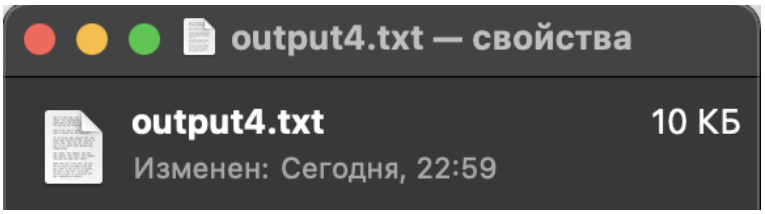
Обработанный текст сохранён:

Если введено неправильное название входного файла:

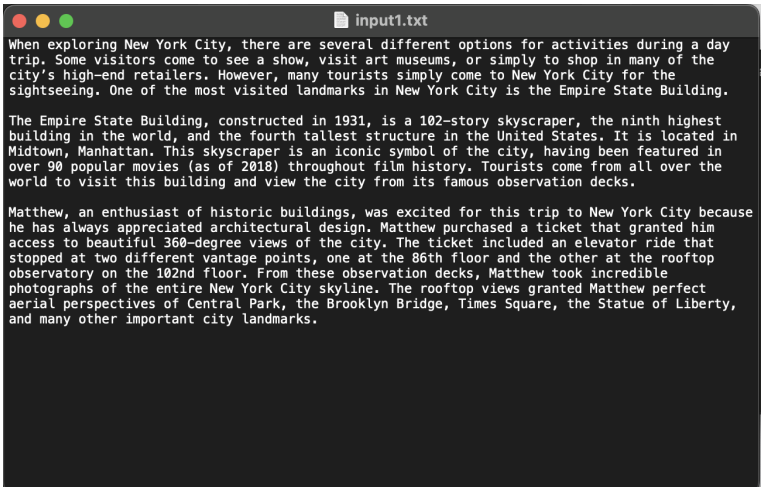


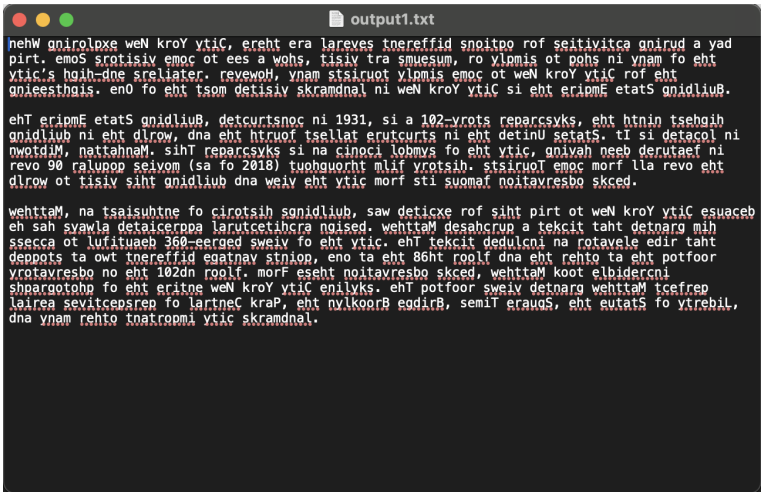
Если входной файл размером больше 10кб (файл input4.txt – 17кб), программа прочитает только 10кб (как сказано по критериям) и обработает этот кусок текста:





Пример текста до и после обработки. (input1.txt):





Как мы видим, всё исправно работает! :)

Отчёт программы на 6–7 баллов:

- Внутри функций используются регистровые или локальные (при нехватке) переменные.

- Для чтения текста из файла реализован буфер ограниченного размера, равного 512 байтам. При этом программа читает файлы размером до 10 килобайт.

- Реализован ввод исходных данных (load_text.asm), их обработка (reverse_text.asm), вывод результатов (output_text.asm) через соответствующие подпрограммы. Подпрограммы получают необходимые им данные через параметры в соответствии с принятым соглашением о передаче параметров.

- Возвращаемые из подпрограмм значения возвращаются через параметры в соответствии с общепринятыми соглашениями.

Отчёт программы на 8 баллов:

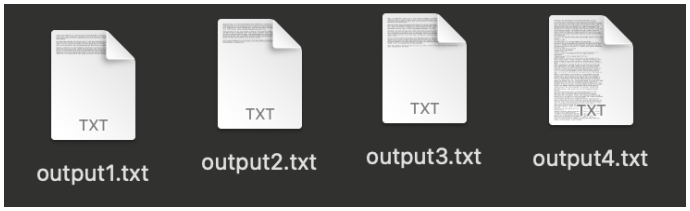
- В программу добавлена возможность дополнительного вывода результатов на дополнительное графическое диалоговое окно (тк было разработна на целевую оценку 10). Выводить или нет решает пользователь отвечая «Y» или «N» на соответствующий вопрос компьютерной программы. Вывод программы при этом полностью соответсвует выводу результатов в файл.

- Реализована дополнительная тестовая программа (main_test.asm), которая осуществляет многократный вызов процедур, обеспечивающих ввод файлов, их обработку и вывод для различных исходных данных, расположенных в каталоге с исходными тестовыми данными.

Отображение прогресса прохождения тестов и сохранение файлов с обработаными текстами:

```
TEST 1 успешно пройден
TEST 2 успешно пройден
TEST 3 успешно пройден
TEST 4 успешно пройден
Все тесты успешно пройдены, обновлённые текста сохранены в новые файлы.

-- program is finished running (0) --
```



Отчёт программы на 9 баллов:

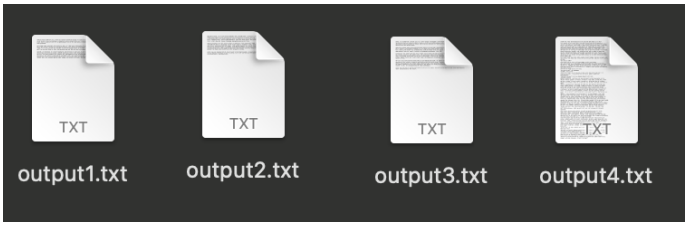
- В программу добавлено использование макросов для реализации ввода, вывода, и обработки данных. Макросы поддерживают повторное использование с различными массивами и другими параметрами. Внутри макросов расположены вызовы соответствующих подпрограмм.

- Реализована дополнительная тестовая программа (main_test_macrolib.asm), которая вызывает выполняемые подпрограммы через макросы, реализуя ту же функциональность, что и предыдущая тестовая программа. Это дополнительная тестовая программа.

Отображение прогресса прохождения тестов и сохранение файлов с обработаными текстами:

```
TEST 1 успешно пройден
TEST 2 успешно пройден
TEST 3 успешно пройден
TEST 4 успешно пройден
Все тесты завршились, обновлённые текста сохранены в новые файлы.

-- program is finished running (0) --
```



Функционал тестирования остался такой же как и в программе тестирования на оценку 8, но используя макросы, как сказано в критериях.

Отчёт программы на 10 баллов:

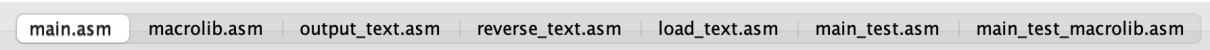
- Программа разбита на несколько единиц компиляции. При этом подпрограммы ввода–вывода составляют унифицированные модули, используемые повторно как в программе, осуществляющей ввод исходных данных, так и в программе, осуществляющей тестовое покрытие.

- Макросы выделены в отдельную автономную библиотеку (macrolib.asm).

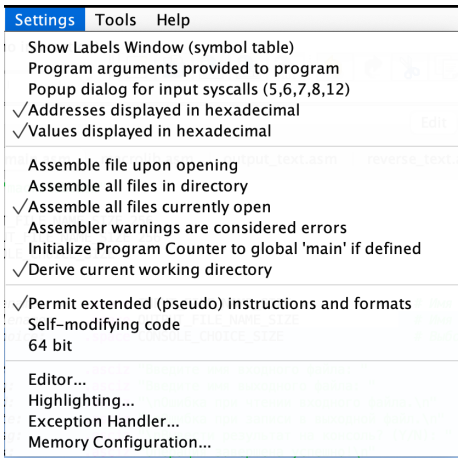
- Используются дополнительные графические диалоговые окна для ввода, отображения диалогов, предоставляемые симулятором RARS.

Обзор программы

Для начала надо открыть все файлы:



Следом выбрать: Settings -> Assemble all files currently open, а также Derive current working directory.



Далее выбираем: хотим ли мы ввод в графическое окно или прогнать тесты.

Если с вводом в графическое окно, то выбираем файл main.asm, ассемблируем и нажимаем Run.

Если хотим прогнать тесты, то выбираем файл main_test.asm или main_test_macrolib.asm, ассемблируем и нажимаем Run.

Примеры ввода в графическое окно можно найти в пункте "Отчёт программы на 4–5 баллов"в разделе "ТЕСТОВОЕ ПОКРЫТИЕ".

СПАСИБО ЗА ВНИМАНИЕ!