ИДЗ 3

Исполнитель: Чапурина Валерия Сергеевна

Группа: БПИ237

Вариант: 4

Условие задания

> ASCII–строка — строка, содержащая символы таблицы кодировки ASCII. **Размер строки** может быть достаточно большим, чтобы вмещать многостраничные тексты, например, главы из книг, если задача связана с использованием файлов или строк, порождаемых генератором случайных чисел. **Тексты** при этом могут не нести смыслового содержания. Для обработки в программе предлагается использовать данные, содержащие символы только из первой половины таблицы (коды в диапазоне $0–127_{10}$), что связано с использованием кодировки UTF-8 в ОС Linux в качестве основной. Символы, содержащие коды выше 127_{10}, **должны отсутствовать** во входных данных кроме оговоренных специально случаев.

> Разработанная программа должна читать обрабатываемый текст из файла и загружать полученные результаты также в файл. Ввод имен входного и выходного файлов должен осуществляться с использованием консоли. Аналогичным образом осуществляет ввод остальных параметров, необходимых для выполнения программы.

4️⃣ Разработать программу, находящую в заданной ASCII–строке последнюю при перемещении слева направо последовательность N символов, каждый элемент которой определяется по условию *«больше предшествующего»* (N вводится как отдельный параметр).

Понятие *«больше предшествующего»* определяется в соответствии с порядком следования символов от начала строки.

Текст программы

лежит тут

Некоторые пояснения к программе

Неочевидные моменты

- “Понятие *«больше предшествующего»* определяется в соответствии с порядком следования символов от начала строки.” - не очень поняла пояснение, полагаю, что элемент x в найденной последовательности больше предшествующего y, если он стоит правее у в исходной строке. То есть нужно найти последние N символов в строке и записать в файл в порядке следования

- Если пользователь вводит число N > длины рассматриваемой строки, то результатом является вся строка

Переменные

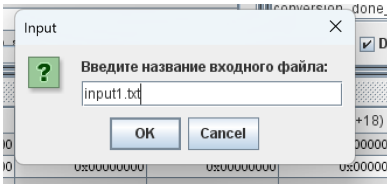
- файловый дескриптор входного файла ⟶ регистр `s0`
- файловый дескриптор выходного файла ⟶ регистр `s1`
- и ещё какие-то

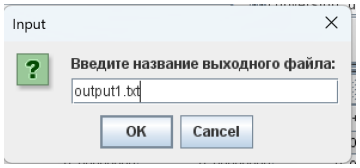
Соответствие требованием на оценку: 10

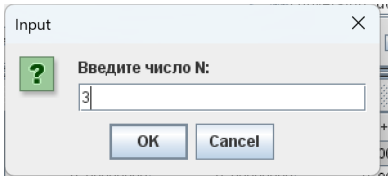
4-5 баллов

- Приведено решение программы на ассемблере. Программа из файла читает данные. Результаты записываются в другой файл.
- Все изменяемые параметры программы вводятся с консоли.
- В программе присутствуют комментарии, поясняющие выполняемые ей действия.

- Решение задачи на ассемблере приведено, программа читает данные из файла, название которого предоставлено пользователем (если такого файла в директории программы нет, выводится сообщение об ошибке). Результаты записываются в файл, название которого также предоставляется пользователем (вводит с клавиатуры, если такой файл в директории программы есть - его содержание перезаписывается, если такого файла нет - он создаётся и в него записывается результат)
- Названия файлов и число N - это все изменяемые параметры программы, они вводятся с клавиатуры пользователем, но не в консоль, а при помощи графических диалоговых окон, о которых говорят в требованиях на оценку 10



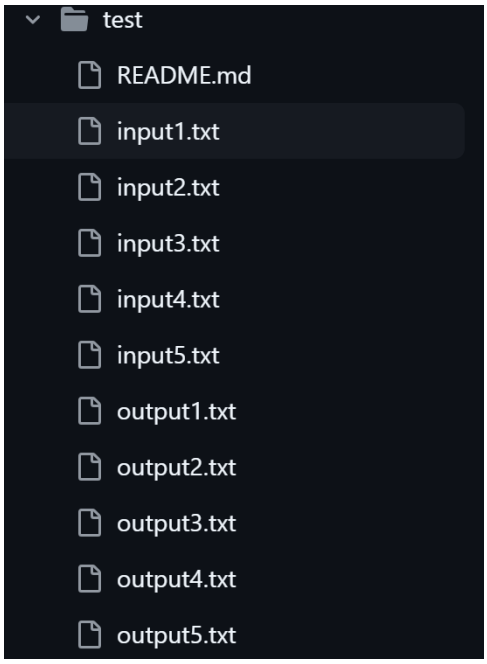




- В программе присутствуют комментарии (правда в большинстве своём на русском)

  - Обработка данных, полученных из файла сформирована в виде отдельной подпрограммы.
  - В подкаталоге данных присутствуют файлы, используемые для тестирования.

- Да, есть подпрограмма `find_last_symbols` в файле subroutines.s, которая ищет последние N символов файла и записывает в файл для вывода
- Присутствуют, лежат в подкаталоге test. В файлах с названиями inputi.txt записаны тестовые данные, в файлах с названиями outputi.txt результаты выполнения программы на тестовых данных (inputi.txt → outputi.txt)



- Буфер для текста программы имеет фиксированный размер размером не менее 4096 байт, допускающий ввод без искажений только тексты, ограниченные этим размером.
- При чтении файла размером, превышающим размер буфера, не должно происходить падения программы. Программа должна корректно обработать введенный «урезанный» текст.
- Сформирован отчет с результатами тестовых прогонов.

При невыполнении хотя бы одного из требований оценка снижается на балл.

- буфер для текста имеет фиксированный размер 10240 байт (10кб), поскольку о таком размере пишут в требованиях на следующую оценку. Это больше 4096 байт. Программа работает только для текстов размера ≤ 10 кб, поскольку об этом пишут в требованиях на следующую оценку

```
.data
        buffer: .space 10240          # Буфер для хранения текста (10 кбайт = 10 * 1024 байт)
```

- при чтении файла размером > 10кб падения программы не происходит, она корректно завершает работу, читая урезанный текст (первые 10кб)
- отчёт сформирован, некоторые скриншоты с результатами работы программы будут ниже, результаты выполнения на 5 тестовых файлах есть в папке test в файлах типа output

6-7 баллов

При разработке программы на данную оценку необходимо учитывать все требования, предъявляемые на предшествующую оценку.

- Внутри функций необходимо использовать регистровые или локальные (при нехватке) переменные.
- Для чтения текста из файла реализовать буфер ограниченного размера, равного 512 байтам. При этом программа должна читать файлы размером до 10 килобайт.

- переменные внутри функций организованы и используются
- при чтении текста из файла использовался обычный буфер на 10кб, но его длина искусственно ограничивалась - читали по 512 байт за итерацию. Программа читает файлы размером до 10кб полностью, если размер больше - читает первые 10кб данных

```
.data
        buffer: .space 10240          # Буфер для хранения текста (10 кбайт = 10 * 1024 байт)
        number_buffer: .space 8       # Буфер для хранения числа в виде строки
```

```
# Определяем, сколько байт читать в текущей итерации
li t1 BUFF_SIZE
blt t0 t1 set_bytes_to_read
mv t2 t1                    # bytes_to_read = 512
j update_values
```

main.s* | macrolib.s | subroutines.s | macrolib-dialog-boxes.s | macrolib-files.s | consts.s | autotests.s

```
1  .eqv MAX_FILE_SIZE 10240        # Максимальный возможный размер файла
2  .eqv BUFF_SIZE 512              # Размер буфера для текста
3  .eqv PATH_SIZE 64               # Размер буфера для имени файла
4  .eqv NUMBER_SIZE 8              # Размер буфера для числа N
```

- Реализовать ввод исходных данных, их обработку, вывод результатов через соответствующие подпрограммы. Подпрограммы должны получать необходимые им данные через параметры в соответствии с принятым соглашением о передаче параметров.
- Возвращаемые из подпрограмм значения должны возвращаться через параметры в соответствии с общепринятыми соглашениями.
- Расширить отчет, дополнив его новыми данными.

При невыполнении хотя бы одного из требований оценка снижается на балл.

- подпрограммы для ввода исходных данных, обработки и вывода результата реализованы. Для ввода данных используется input_string_dialog и соответствующая макро-обёртка, для записи последних N символов текста в файл используется подпрограмма `find_last_symbols`, для конвертации N из строки в число используется макро `string_to_int(%reg_string, %reg_int)`
- всё так и делаем

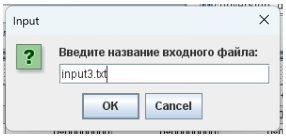
8 баллов

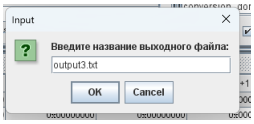
При разработке программы на данную оценку необходимо учитывать все требования, предъявляемые на предшествующие оценки.

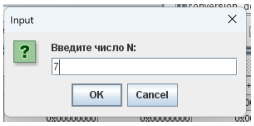
- Добавить в программу возможность дополнительного вывода результатов на консоль. Выводить или нет решает пользователь отвечая «Y» или «N» на соответствующий вопрос компьютерной программы. Данная возможность полезна при быстром отображении коротких данных. Вывод программы при этом должен полностью соответствовать выводу результатов в файл.

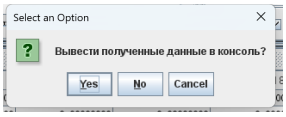
- такая возможность была добавлена. Если пользователь выбирает “Yes” в диалоговом окне, то результаты обработки данных выводятся как в консоль, так и в файл. Если выбирает “No’, результаты записываются

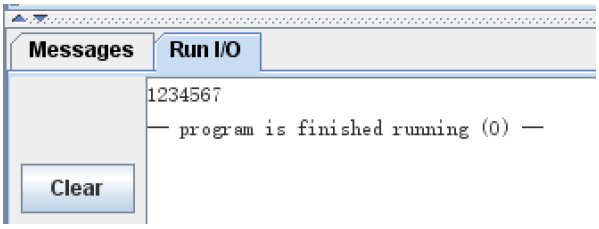
только в файл. Приведём пример для тестовых данных из файла input3.txt и N = 7, результат запишем в файл output3.txt

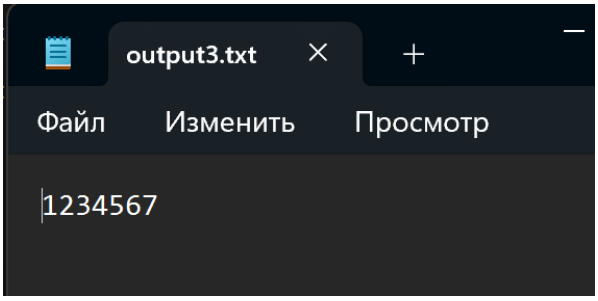












- Реализовать дополнительную тестовую программу которая осуществляет многократный вызов процедур, обеспечивающих ввод файлов, их обработку и вывод для различных исходных данных, расположенных в каталоге с исходными тестовыми данными.

  - Расширить отчет, дополнив его новыми данными.

При невыполнении хотя бы одного из требований оценка снижается на балл.

- реализована, это программа autotests, она работает для 5 тестовых файлов input, записывает данные в 5 соответствующих файлов output

9 баллов

При разработке программы на данную оценку необходимо учитывать все требования, предъявляемые на предшествующие оценки.

- Добавить в программу использование макросов для реализации ввода, вывода, и обработки данных. Макросы должны поддерживать повторное использование с различными массивами и другими параметрами. Внутри макросов должны быть расположены вызовы соответствующих подпрограмм.
- Реализовать дополнительную тестовую программу, которая вызывает выполняемые подпрограммы через макросы, реализуя ту же функциональность, что и предыдущая тестовая программа. Это должна быть дополнительная тестовая программа.
- Расширить отчет, дополнив его новыми данными.

При невыполнении хотя бы одного из требований оценка снижается на балл.

- макросы для реализации ввода, вывода и обработки данных добавлены (про них писалось выше). Они поддерживают повторное использование с различными параметрами. Вызовы соответствующих подпрограмм в них расположены
- при написании отчёта увидела слово “дополнительная”, не очень поняла, зачем нужно 2 тестовых программы, у меня autotests соответствует описанию этой, то есть вызывает выполняемые подпрограммы через макросы. Можно конечно покопипастить код подпрограмм и макросов и получить ещё одну тестовую программу, но как будто бы незачем

10 баллов

При разработке программы на данную оценку необходимо учитывать все требования, предъявляемые на предшествующие оценки.

- Программа должна быть разбита на несколько единиц компиляции. При этом подпрограммы ввода–вывода должны составлять унифицированные модули, используемые повторно как в программе, осуществляющей ввод исходных данных, так и в программе, осуществляющей тестовое покрытие.
- Макросы должны быть выделены в отдельную автономную библиотеку

- разбита на несколько единиц компиляции, подпрограммы ввода-вывода являются отдельными независимыми программами (унифицированными модулями), которые используются в обеих программах: осуществляющей тестовое покрытие и работающей с пользователем
- макросы выделены в 3 отдельных автономных библиотеки (решила классифицировать их и разбить так, чтобы удобнее было; при этом не все написанные макросы используются в программе, некоторые лежат просто чтобы было (чтобы можно было воспользоваться при необходимости))

- Использовать дополнительные графические диалоговые окна для ввода и отображения диалогов, предоставляемые симулятором RARS.
- Расширить отчет, дополнив его новыми данными.

При невыполнении хотя бы одного из требований оценка снижается на балл.

- дополнительные графические диалоговые окна реализованы для ввода данных и отображения диалогов. Выше были примеры, есть ещё окно,

сигнализирующее об ошибке (например, если введённого названия файла со входными данными нет рядом с программой)