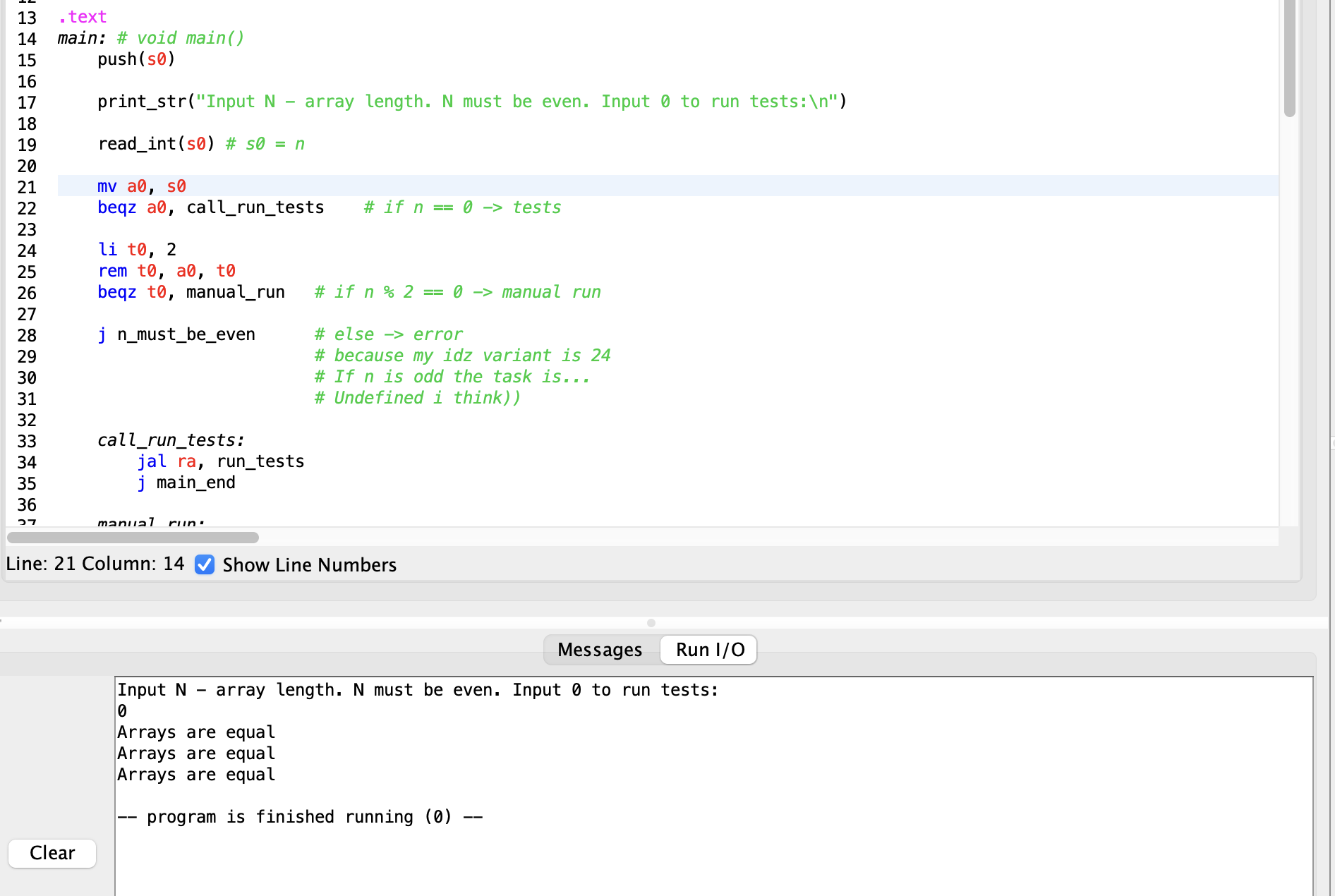
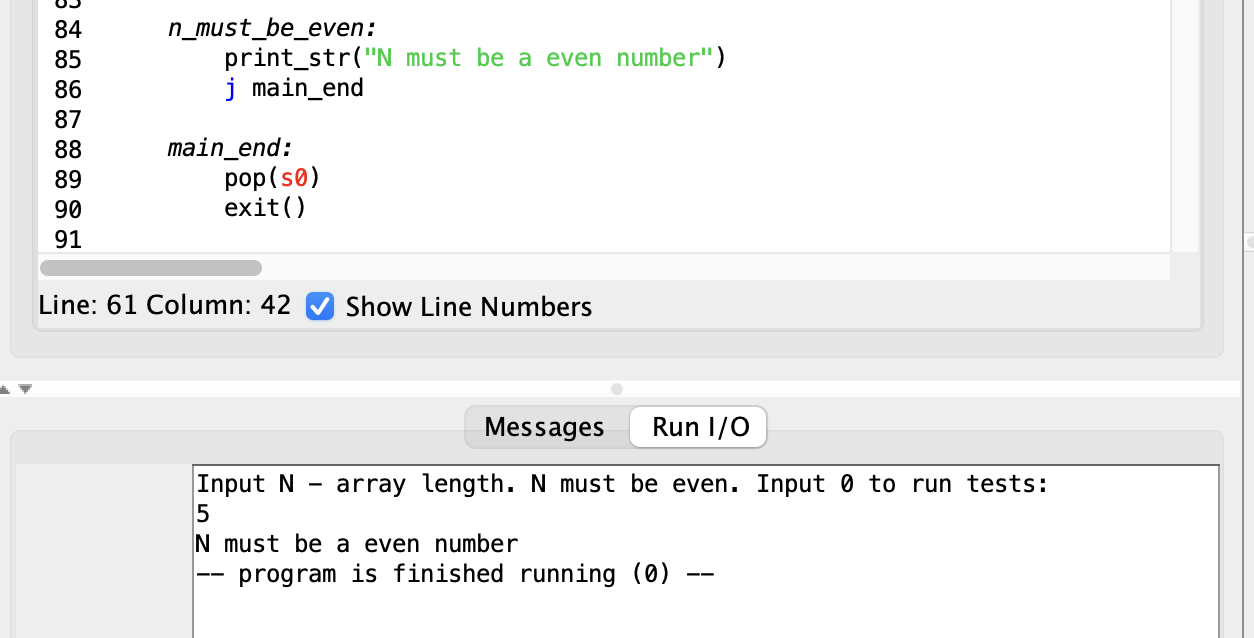
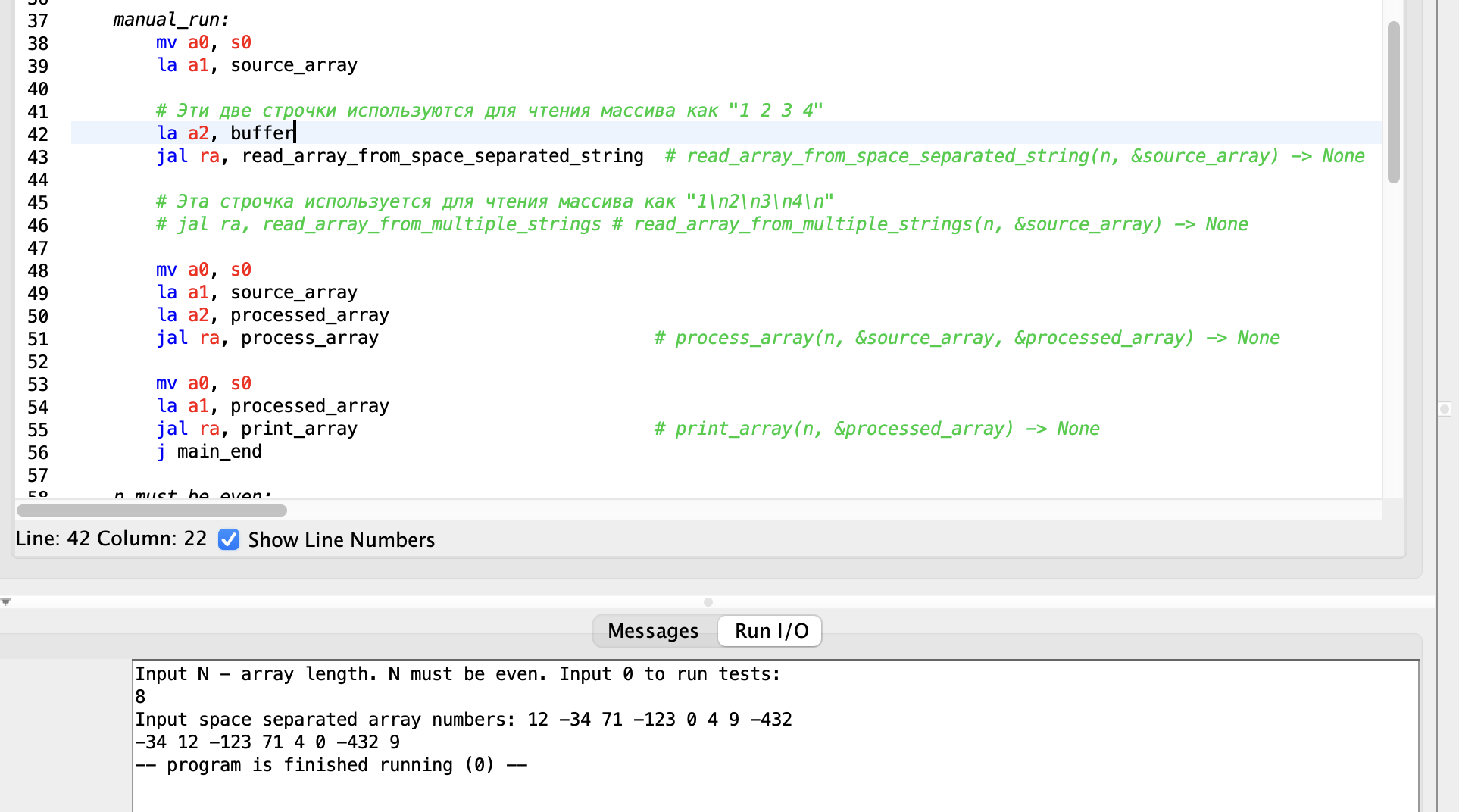
* Горячев Иван Сергеевич, БПИ236
* Вариант 24 ([ссылка на ячейку в таблице](https://docs.google.com/spreadsheets/d/19lfazOenTnUWMmakZ6Fi3dQBMMAFLdxB28mdT2o2P8A/edit?gid=1233953292#gid=1233953292&range=B9)):  
  Сформировать массив B из элементов массива A поменяв местами элементы, стоящие на чётных и нечётных местах:  
  A0 ↔A1;A2 ↔A3...
* Я решил, что для нечетной длины массива задание не определено, поэтому программа при получении нечетного числа на вход завершается с сообщением о некорректных данных

При запуске программы требуется ввести длину массива  
1) При введении 0 происходит запуск тестов   


2) При введении нечетного числа в качестве длины массива - происходит завершение программы



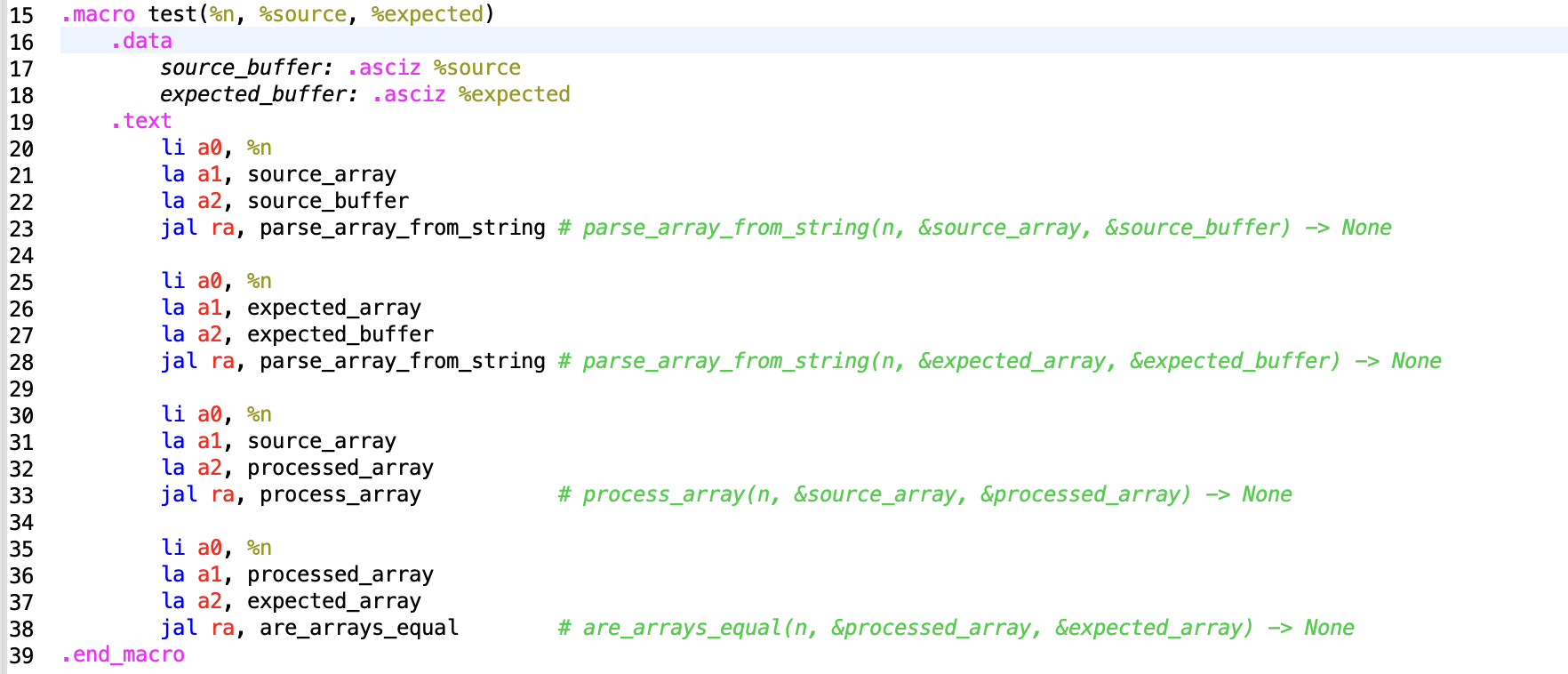
3) В остальных случаях происходит запрос на введение элементов массива и дальнейшая их обработка 

Введение элементов массива я реализовал двумя способами – по умолчанию включен первый (строчки 41-43). При желании можно включить второй вместо первого (строчки 45-46). В обоих способах предусмотрена также работа с отрицательными числами.

По критериям подряд (считаю что выполнил их все):  
4-5 баллов

1. Код на ассемблере написан. Ввод осуществляется с клавиатуры. Вывод осуществляется на дисплей
2. Комментарии присутствуют
3. Подпрограммы используются
4. В отчете скриншоты есть (выше)

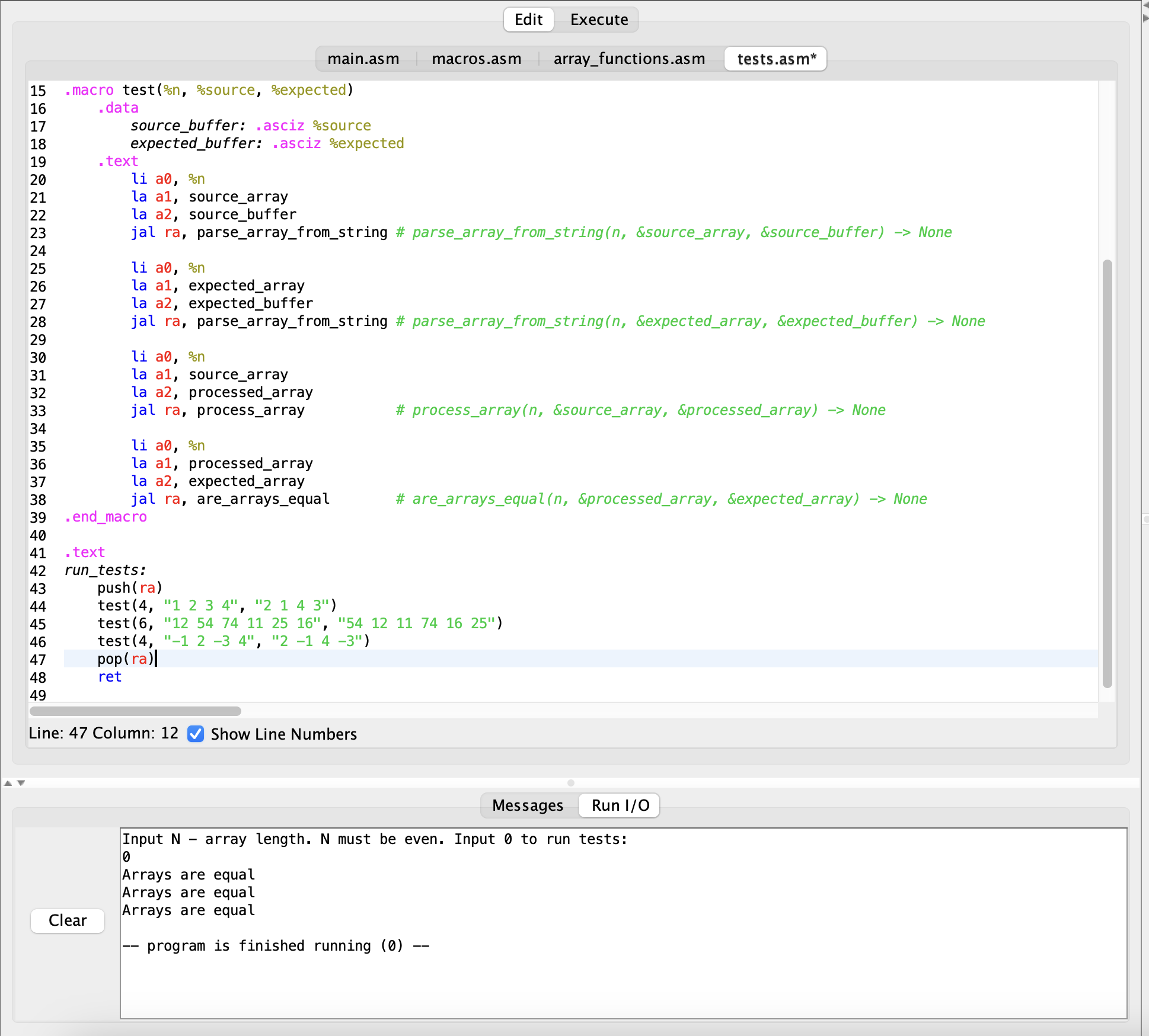
6-7 баллов

1. Все подпрограммы принимают аргументы в региcтрах a0, a1 и a2 (больше не нужно). Подпрограммы ничего не возвращают (как void в C и C++).  
   Регистры s0, s1, s2 итд сохраняют свои значения по итогу работы программы – конвенции не нарушены.
2. Локальные переменные хранятся в свободных регистрах.
3. В местах вызова функции добавлены комментарии, описывающие передачу фактических параметров. Аргументы хранятся в a0, a1, и иногда a2 – я не стал описывать это в комментариях к коду поскольку это избыточно, а код отформатирован максимально наглядным образом (пример ниже) 
4. Эту информацию в отчете отразил (странный критерий)

8 баллов

1. Разработанные подпрограммы написаны гибко и потому поддерживают многократное использование с различными наборами исходных данных.
2. Тестовая программа написана отдельным файлом. Чтобы обеспечить единую точку входа в программу и для упрощения работы с ней добавлена возможность запуска тестов из основной программы (при введении нулевой длины массива)
3. Изменения в отчет добавлены (еще один странный критерий)

9 баллов

1. Макросы на ввод вывод информации написаны. Не стал писать только макрос на чтение строки в буфер, поскольку это происходит один раз во всем коде и я решил что это не целесообразно.  
   Также я написал макрос для тестирования. Тесты выглядят так.   
   

10 баллов

1. Программа разбита на несколько файлов (4 файла).

- Библиотека макросов macros.asm  
- Функции для работы с массивами, не специфичные для данной задачи вынесены в отдельный файл array\_functions.asm   
- Функции main и process\_array оставлены в файле main.asm

- Макрос для тестов и сами тесты вынесены в файл tests.asm  
- Функции для работы с массивами используются как внутри тестовой программы, так и в основной

1. Макросы выделены в отдельную автономную библиотеку
2. Отчет расширен