|  |
| --- |
| **ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**  **НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**  **«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**  Факультет компьютерных наук  Департамент программной инженерии  **КОНСОЛЬНОЕ ПРИЛОЖЕНИЕ, ЗАМЕНЯЮЩЕЕ ГЛАСНЫЕ БУКВЫ НА СООТВЕТСТВУЮЩИЕ ЗАГЛАВНЫЕ**  Пояснительная записка  Исполнитель:  Студент группы БПИ191  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Дондик Я.В. /  «1» ноября 2020 г. |

Содержание

[1. Текст задания 2](#_Toc55162260)

[2. Применяемые расчетные методы 3](#_Toc55162261)

[2.1. Теория решения задания 3](#_Toc55162262)

[2.1.1. Общая идея решения задания 3](#_Toc55162263)

[2.1.2. Входные и выходные данные 3](#_Toc55162264)

[2.1.3. Особенности при решении задания в FASM 3](#_Toc55162265)

[2.2. Дополнительный функционал программы 3](#_Toc55162266)

[3. Тестирование программы 4](#_Toc55162267)

[3.1.1. Тест 1 4](#_Toc55162268)

[3.1.2. Тест 2 4](#_Toc55162269)

[3.1.3. Тест 3 5](#_Toc55162270)

[3.1.4. Тест 4 5](#_Toc55162271)

[3.2. Проверка программы на некорректных данных 6](#_Toc55162272)

[4. Список источников. 7](#_Toc55162273)

[ПРИЛОЖЕНИЕ 1 8](#_Toc55162274)

[ПРИЛОЖЕНИЕ 2 9](#_Toc55162275)

1. Текст задания

Разработать программу, заменяющую все гласные буквы в заданной ASCII-строке заглавными

2. Применяемые расчетные методы

2.1. Теория решения задания

2.1.1. Общая идея решения задания

После считывания строки проходимся по каждому символу и проверяем, является ли символ строчной латинской гласной буквой. Если да, то заменяем её на соответствующую заглавную букву.

2.1.2. Входные и выходные данные

Входные данные: строка, состоящая из символов из таблицы ASCII. Ограничение по количеству символов - 1000 символов.

Выходные данные: преобразованная строка, в которой все строчные латинские гласные буквы заменены на соответствующие заглавные.

2.1.3. Особенности при решении задания в FASM

1. Для работы со строками используются регистры ESI (источник данных) и EDI (приемник данных)
2. Для выделения символа исходной строки используется команда lodsb (выделяет байт в регистр AL)
3. Для проверки символа на то, является ли он строчной латинской гласной буквой, используется префикс repne в сочетании с командой scas byte (идем до конца строки со всеми гласными буквами или пока не встретим ту, что находится в регистре AL после п.2.)
4. Для записи символа в регистр EDI используется команда stosb (помещает байт AL в байт, на который указывает EDI)
5. Для работы с циклами используется команда LOOP
6. Код программы разбит на файл компиляции и файл с макросами

2.2. Дополнительный функционал программы

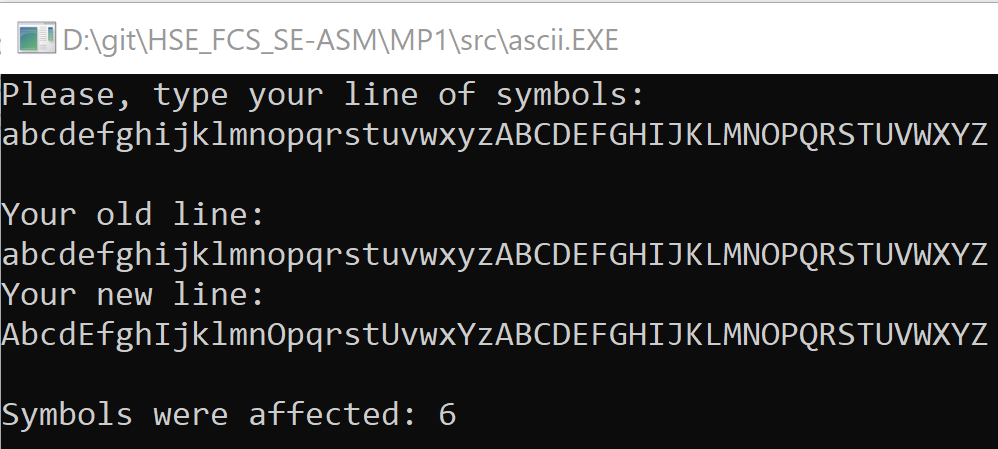
Программа подсчитывает количество измененных символов в строке (то есть количество строчных гласных букв, который далее были заменены заглавными)

3. Тестирование программы

**3.1. Проверка программы на корректных данных**

3.1.1. Тест 1

Проверим программу на латинском алфавите (включая строчные и заглавные буквы):

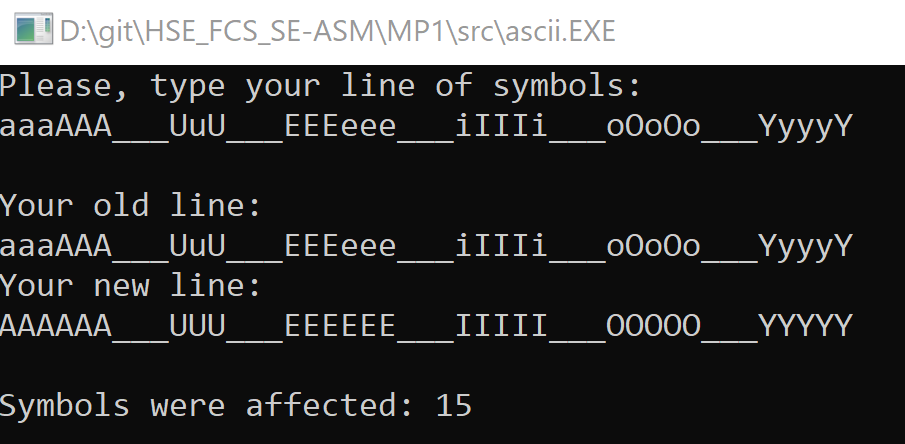


*Рисунок 1.* Тестирование программы на латинском алфавите

Несложно заметить, что были заменены на заглавные только 6 символов (столько гласных в латинском алфавите). Программа отработала успешно.

3.1.2. Тест 2

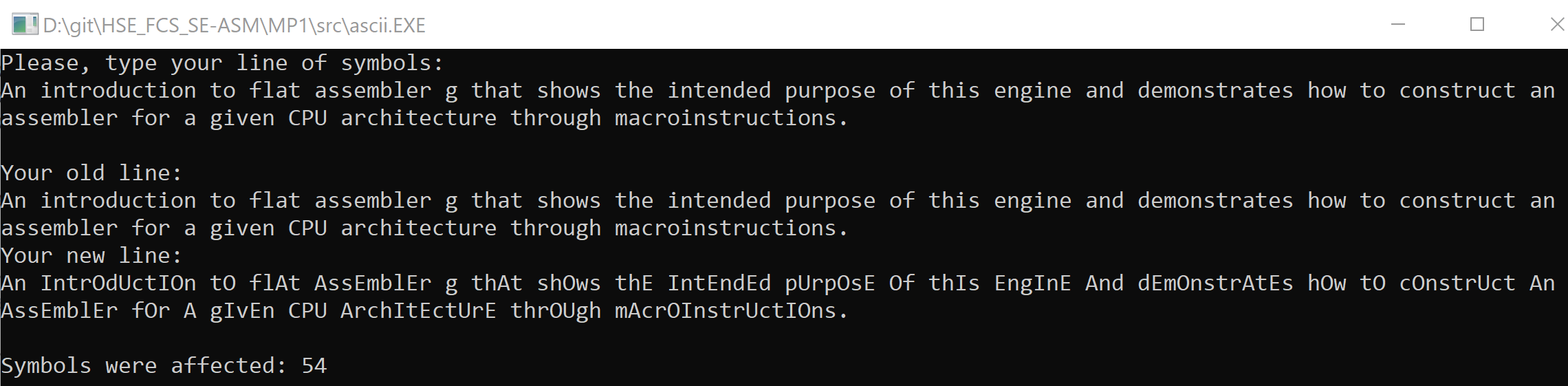
Проверим программу на тесте с повторениями гласных букв:



*Рисунок 2.* Тестирование программы на строке с повторением гласных букв

Несложно заметить, что все строчные гласные были заменены на заглавные, а их количество действительно было 15 штук. Программа отработала успешно.

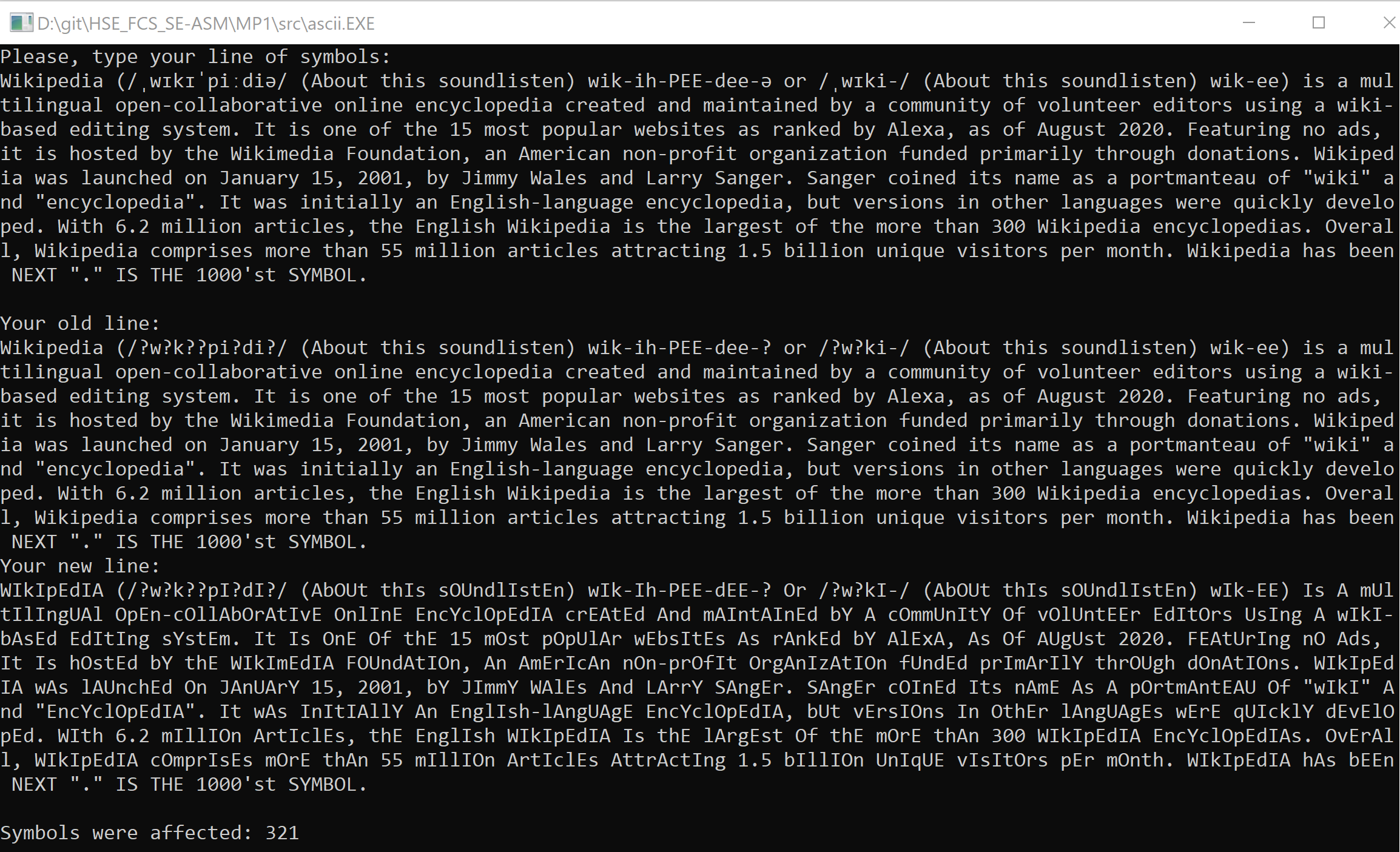
3.1.3. Тест 3

Проверим программу на строку с текстом (то есть включая символы пробела):

*Рисунок 3.* Тестирование программы на тексте (с пробелами)

Все строчные гласные были заменены на заглавные. Программа отработала успешно.

3.1.4. Тест 4

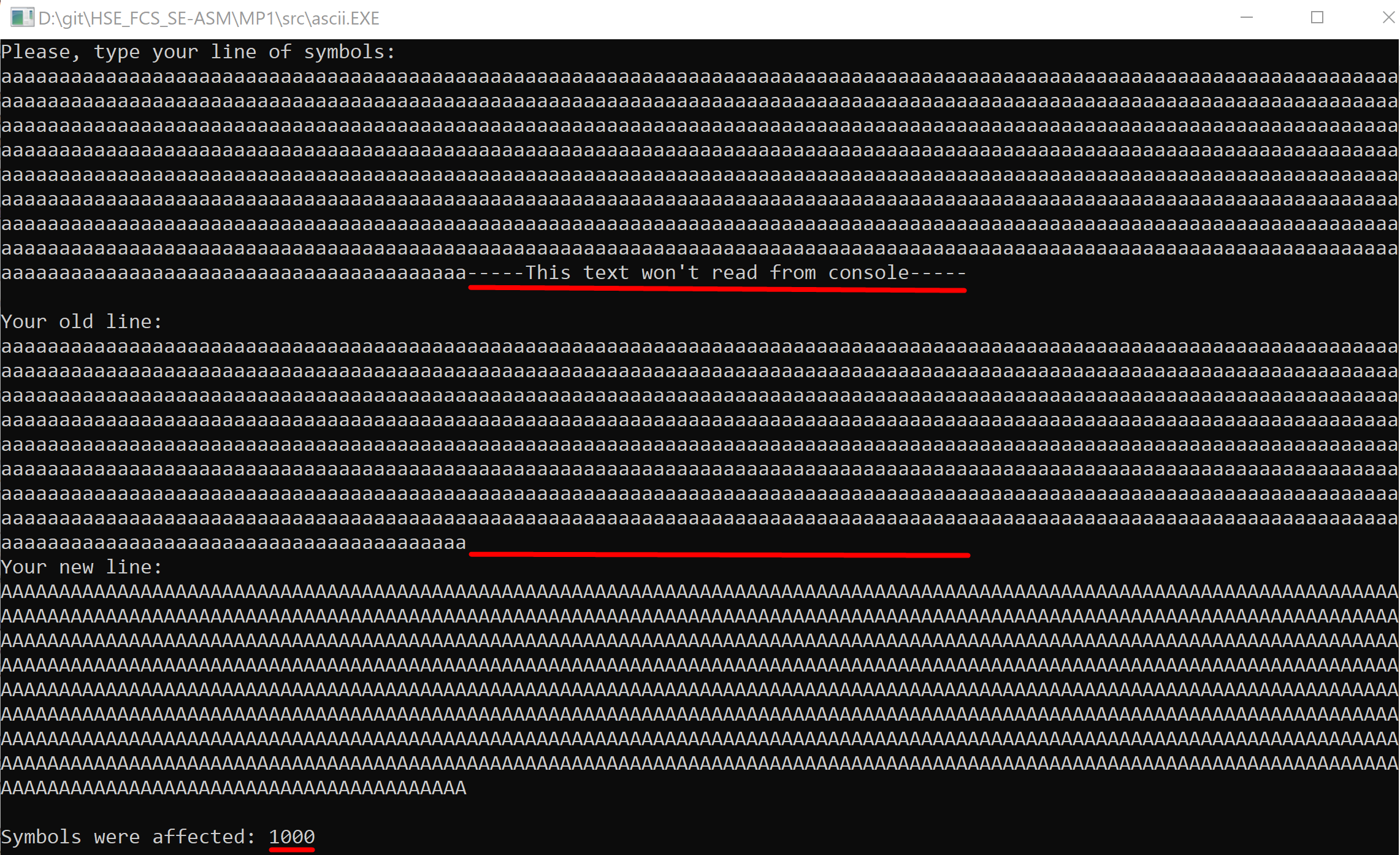
Проверим программу на работу со строкой в 1000 символов (ограничение входных данных):

*Рисунок 4.* Тестирование программы на строке c 1000 символами

Все строчные гласные были заменены на заглавные. Строка была успешно обработана.

3.2. Проверка программы на некорректных данных

Проверим программу на строке, имеющей больше 1000 символов:



*Рисунок 5.* Тестирование программы на строке с более, чем 1000 символами

Несложно заметить, что были считаны и заменены на заглавные все 1000 первых символов. Программа отработала успешно.

4. Список источников.

1. Документация по flat assembler [Электронный ресурс] // FASM: [сайт]. [2020]. URL: <https://flatassembler.net/docs.php?article=manual>, режим доступа: свободный, дата обращения: 01.11.2020

2. Условия выполнения задания, сайт дисциплины [Электронный ресурс] // SoftCraft: [сайт]. [2020]. URL: <http://softcraft.ru/edu/comparch/tasks/mp01/> режим доступа: свободный, дата обращения: 01.11.2020

3. Описание команды STOS [Электронный ресурс] // Уголок системного программиста: [сайт]. [2020]. URL: <http://sysprog.ru/post/60>, режим доступа: свободный, дата обращения: 01.11.2020

4. Описание команды LODS [Электронный ресурс] // Уголок системного программиста: [сайт]. [2020]. URL: <http://sysprog.ru/post/40>, режим доступа: свободный, дата обращения: 01.11.2020

5. Описание команды SCAS [Электронный ресурс] // Уголок системного программиста: [сайт]. [2020]. URL: <http://sysprog.ru/post/56>, режим доступа: свободный, дата обращения: 01.11.2020

ПРИЛОЖЕНИЕ 1

**Описание переменных**

|  |  |
| --- | --- |
| Название переменной | Описание |
| str1 | Исходная строка, считанная из консоли |
| str2 | Преобразованная строка, в которой строчные гласные буквы заменены на заглавные |
| numChg | Количество затронутых символов |
| tmpStack | Временная переменная для стека |
| tmpStack2 | Временная переменная для стека |
| ediTmp | Временная переменная для регистра EDI |

**Описание процедур**

|  |  |  |
| --- | --- | --- |
| Название процедуры | Описание | Примечание |
| IterateStringSymbols | Проходит по каждому символу в ESI. Проверяет, гласная ли буква. Записывает символ в EDI | До вызова процедуры следует передать в регистры ESI и EDI соответствующие адреса источника и приемника данных |
| CheckForSyllable | Проверяет, является ли регистр AL гласной или нет. После этого записывает символ в EDI | До вызова процедуры следует поместить проверяемый символ в регистр AL |
| numChg | Процедура вычисления длины строки | Строка должны быть передана в стек до вызова процедуры |

ПРИЛОЖЕНИЕ 2

**Код основной программы**

; Дондик Ярослав Витальевич

; Студент группы БПИ191

; Вариант 14

;

; Условие: Разработать программу, заменяющую все гласные буквы в заданной ASCII-строке заглавными

; Входные данные: строка, длинной не более 1 000 символов

; Выходные данные: строка, в которой все строчные гласные заменены соответствующими заглавными буквами

format PE console

entry \_start2

include 'win32a.inc'

include 'ascii\_macro.inc'

section '.data' data readable writable;

formatStr db '%s', 0

inputFormatStr db "%1000[^", 10,"]c", 0 ; считываем максимум 1000 символов

syllStr db 'aeouyi', 0

typeString db 'Please, type your line of symbols:', 10, 13, 0

yourOld db 10, 13, 'Your old line:', 10, 13, '%s', 10, 13, 0

yourNew db 'Your new line:', 10, 13, '%s', 10, 13, 10, 13, 0

symbolsCh db 'Symbols were affected: %d ', 10, 13, 0

newLine db 10, 13, 0

str1 rb 1001 ; исходная строка (на 1 больше, для признака конца строки)

str2 rb 1001 ; новая строка

numChg dd 0 ; количество измененных символов

tmpStack dd ? ; временная переменная для стека

tmpStack2 dd ? ; временная переменная для стека

ediTmp dd ? ; временная переменная для EDI

NULL = 0

section '.code' code readable executable

\_start2:

cinvoke printf, typeString ; просим ввести строку символов

cinvoke scanf, inputFormatStr, str1 ; считываем ASCII строку из консоли (максимум 1 000 символов)

mov esi, str1 ; указываем адрес источника как str1 (исходная строка)

mov edi, str2 ; указываем адрес приемника как str2 (новая строка)

call IterateStringSymbols ; проходимся по всем символам введенной строки

cinvoke printf, yourOld, str1 ; выводим исходную строку в консоль

cinvoke printf, yourNew, str2 ; выводим получившуюся строку в консоль

cinvoke printf, symbolsCh, [numChg] ; выводим количество затронутых символов

call [getch]

jmp ExitProgram

;------------------------------------

; Проходит по каждому символы в ESI. Проверяет, гласная ли буква. Записывает символ в EDI

IterationOfStringSymbols

; Проверяет, является ли регистр AL гласной или нет. После этого записывает символ в EDI

CheckSymbolForBeingSyllable

; Процедура вычисления длины строки

StrlenProcedure

ExitProgram:

push NULL

call [ExitProcess]

section '.idata' import data readable

library kernel, 'kernel32.dll',\

msvcrt, 'msvcrt.dll'

import kernel,\

ExitProcess, 'ExitProcess',\

VirtualAlloc, 'VirtualAlloc',\ ; для резервирования памяти

VirtualFree, 'VirtualFree' ; для освобождения памяти

import msvcrt,\

printf, 'printf',\

scanf, 'scanf',\ ;добавим функцию для считывания

getch, '\_getch'

**Код макросов**

;--------------------------------------------------------------------------

macro IterationOfStringSymbols {

;--------------------------------------------------------------------------

; Проходит по каждому символы в ESI. Проверяет, гласная ли буква. Записывает символ в EDI

IterateStringSymbols:

mov [tmpStack], esp ; сохраняем стек вместе в адресом возврата

stdcall strlen, esi ; подсчитываем длину строки источника

mov ecx, eax ; установливаем счетчик цикла на EAX раз (длина строки)

\_loop:

push ecx ; сохраняем счетчик в стек

lodsb ; загружаем символ [esi] (символ, на который указывает esi) в регистр AL: { AL = [esi]; esi += 1; }

mov [ediTmp], edi ; сохраняем адрес приемника

call CheckForSyllable ; проверяем символ AL на то, является ли гласной буквой

pop ecx ; достаем счетчик из стека

LOOP \_loop

mov esp, [tmpStack] ; возвращаем адрес возврата в стек

ret

}

;--------------------------------------------------------------------------

macro CheckSymbolForBeingSyllable {

;--------------------------------------------------------------------------

; Проверяет, является ли регистр AL гласной или нет. После этого записывает символ в EDI

CheckForSyllable:

mov [tmpStack2], esp ; сохраняем стек вместе в адресом возврата

mov edi, syllStr ; указываем адрес приемника как строку с прописными гласными буквами

mov ecx, 6 ; устанавливаем максимум 6 итераций, т.к. всего 6 гласных букв

repne scas byte [edi] ; while ( ecx > 0 && syllStr[i] != AL) { ecx -= 1; edi += 1}

je ToUpper ; если символ гласной нашелся в строке, устанавливается флаг ZF = 1 и происходит прыжок je

jmp EndChecking ; иначе просто заканчиваем проверку

ToUpper:

sub al, 32 ; вычитаем 32 из кода ASCII прописной гласной (наш AL из [esi]) и получаем заглавную гласную

inc [numChg] ; увеличиваем количество затронутых символов

jmp EndChecking ; заканчиваем проверку

EndChecking:

mov edi, [ediTmp] ; возвращаем наш регистр-приемник (str2)

stosb ; устанавливаем AL в байт, на который указывает edi (регистр-приемник): {[edi] = AL; edi += 1; }

mov esp, [tmpStack2] ; возвращаем адрес возврата в стек

ret

}

;--------------------------------------------------------------------------

macro StrlenProcedure {

;--------------------------------------------------------------------------

; Процедура вычисления длины строки из стека

strlen:

mov [ediTmp], edi ; сохраняем предыдущий регистр EDI

mov edi, [esp+4] ; используем str как аргумент из стэка

mov ecx, -1 ; ecx < 0, чтобы цикл не закончился раньше времени

xor al, al ; последний элемент строки - 0

cld ; обнуляем флаг DF, т.к. прямо проход по строке

repne scasb ; while(str[edi] != al) {edi++; ecx--;}

neg ecx

sub ecx, 1 ; ecx = length(str)-1; count \0 symbol

mov eax, ecx

mov edi, [ediTmp] ; возвращаем предыдуший EDI назад

ret

}