*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана***  ***(национальный исследовательский университет)»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ \_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Отчет**

**по домашнему заданию № \_1\_\_\_\_**

**Дисциплина: \_**Машинно-зависимые языки и основы компиляции**\_\_\_\_**

**Название лабораторной работы: \_**Обработка символьной информации**\_**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент гр. **\_**ИУ6-42б**\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**И.С. Марчук**\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Москва, 2021

**Задание:**

Разработать программу на языке ассемблера, которая решает поставленную задачу.

**Вариант 16.**

Дан текст 8 слов, разделенных пробелом. Определить количество повторений буквы Е в каждом слове.

**Цель работы:**

Изучение команд обработки цепочек и приемов обработки символьной информации

**Схема алгоритма для решения поставленной задачи представлена на рисунке 1:**

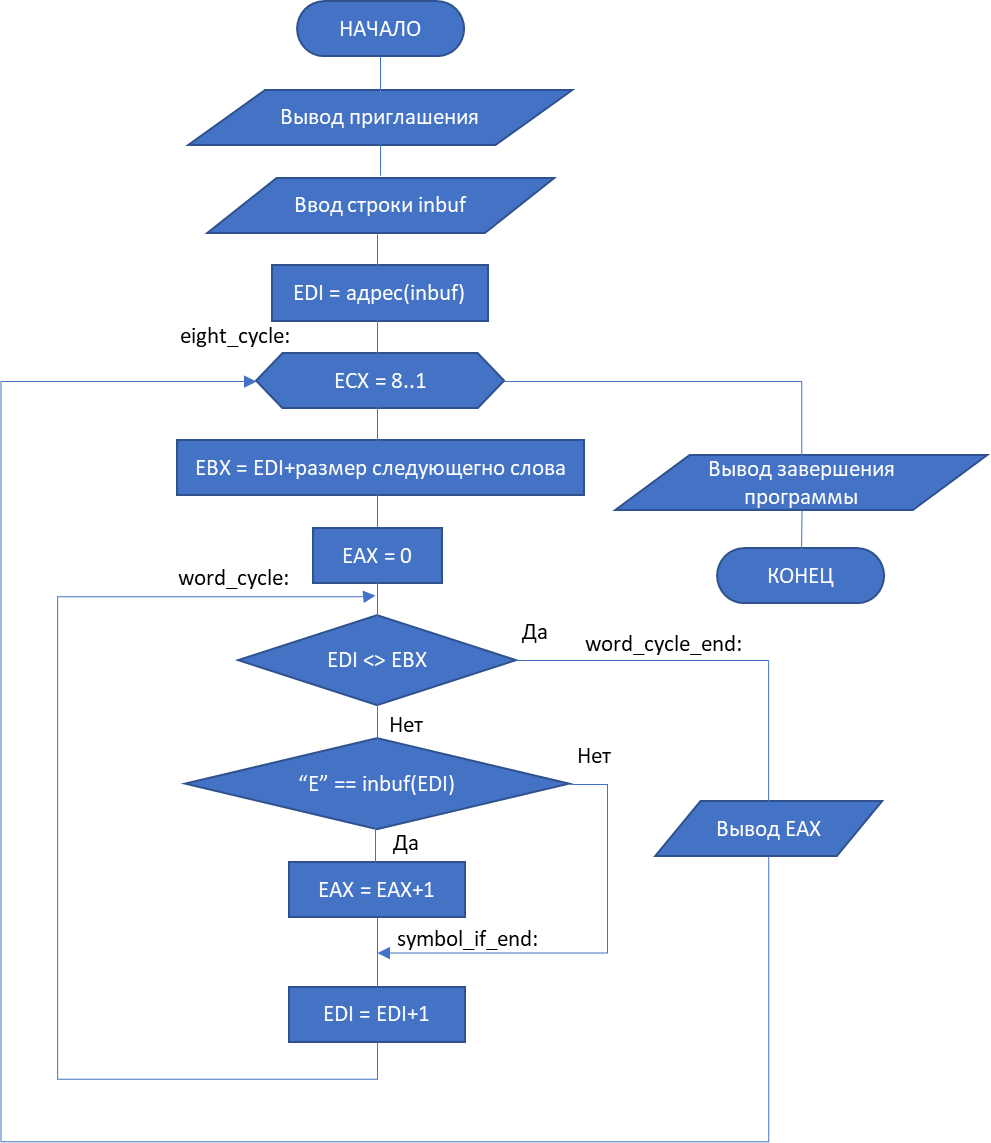


Рисунок 1 – Схема алгоритма

**Текст программы с комментариями:**

**.586** ; подключение набора команд Реntium

**.MODEL flat, stdcall** ; модель памяти и

; конвенция о передаче параметров

**OPTION CASEMAP:NONE** ; опция различия строчных

; и прописных букв

**Include kernel32.inc** ; подключение описаний процедур и

**Include masm32.inc** ; констант

**IncludeLib kernel32.lib** ; подключение библиотек

**IncludeLib masm32.lib**

**.CONST**

**msgEnter DB "Enter eight words separated by spaces and no more than 50 characters:",0AH,0DH,0**

**MsgExit DB 0AH,0DH,"Press Enter to Exit",0AH,0DH,0**

**nextS DB " ",0**

**.DATA**

**separate DB ' '**; разделитель слов

**symbol DB 'E'**; искомый символ

**.DATA?**

**inbuf DB 100 DUP (?)**

**ans DWORD ?**

**.CODE**

**Start:**

;

; Вариант 16.

; Дан текст 8 слов, разделенных пробелом.

; Определить количество повторений буквы Е в каждом слове.

**Invoke StdOut, ADDR msgEnter** ; Приглашение

**Invoke StdIn,ADDR inbuf,LengthOf inbuf; вводим строку**

**LEA EDI, inbuf ; получаем адрес начала строки**

**MOV ECX, 8**

**eight\_cycle:** ; цикл из восьми слов

**PUSH ECX**

**PUSH EDI** ; запоминаем адрес начала слова

**MOV ECX, 50** ; слово не может быть больше 50

**MOV AL, separate**; записываем ссылку на разделитель в al

**CLD** ; обработака от начала к концу

**REPNE SCASB** ; ищем разделитель(конец строки) попутно увеличивая счетчик EDI в итоге получая длинну слова

**MOV EBX, EDI** ; запоминаем адрес конца слова

**POP EDI** ; возвращаем обратно адрес начала слова

**MOV EAX, 0**; Инициализируем количество E в слове

**word\_cycle:** ; цикл while(поисковый) по длинне слова

**CMP EDI, EBX; пока EDI <> EBX**

**JE word\_cycle\_end**

**MOV CL, symbol**  ; сравниваем текущий символ с искомым, и если находим

**CMP CL, DS:[EDI]**

**JNE symbol\_if\_end**

**INC EAX** ; увеличиваем количество E на 1

**symbol\_if\_end:** ; если не находим, то ничего не делаем

**INC EDI**; переходим к следующему элементу

**JMP word\_cycle**

**word\_cycle\_end:**

; выводим количество искомых символов

**MOV ans, EAX**

**Invoke dwtoa, ans, ADDR inbuf** ; прреодразование ответа-числа в строку

**Invoke StdOut, ADDR inbuf** ; вывод

**Invoke StdOut, ADDR nextS** ; вывод разделителя

**POP ECX**

**LOOP eight\_cycle**; переход к следующему слову

**XOR EAX,EAX**

**Invoke StdOut,ADDR MsgExit**

**Invoke StdIn,ADDR inbuf,LengthOf inbuf**

**Invoke ExitProcess,0**

**End Start**

**Работа программы с тестовыми даннымипроиллюстрирована на рисунке 2, а результаты представлена в таблице 1:**

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Полученный результат |
| **EEEE EEE** test**E**r CHETSR**E** PYAT Shest S**E**m VOSeM | 4 3 1 2 0 0 1 0 | 4 3 1 2 0 0 1 0 |
| IT IS MY T**E**ST T**E**XT WITH **EEEEEEEE** YAY | 0 0 0 1 1 0 8 0 | 0 0 0 1 1 0 8 0 |
| ASS**E**MBL**E**R IS TH**E** B**E**ST LANG I LOV**E** IU6 | 2 0 1 1 0 0 1 0 | 2 0 1 1 0 0 1 0 |

Таблица 1 – Отладка программы

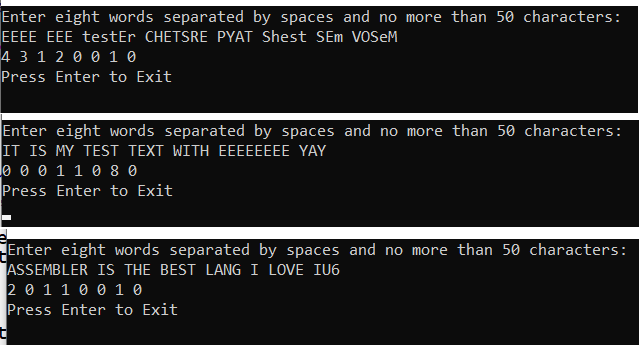


Рисунок 2 – Вывод программы при различных входных данных

**Контрольные вопросы:**

1) Дайте определение символьной строки.

Под строкой символов понимается последовательность байт идущих в памяти друг за другом.

2) Назовите основные команды обработки цепочек?

- пересылка цепочки:

movs адрес\_приемника,адрес\_источника

movsb

movsw

movsd

- сравнение цепочек:

cmps адрес\_приемника,адрес\_источника

cmpsb

cmpsw

cmpsd

- сканирование цепочки:

scas адрес\_приемника

scasb

scasw

scasd

- загрузка элемента из цепочки:

lods адрес\_источника

lodsb

lodsw

lodsd

- сохранение элемента в цепочке:

stos адрес\_приемника

stosb

stows

stosd

Какие операции выполняют строковые команды MOVS? Какие особенности характерны для этих команд?

Эта команда выполняет примитивный перенос элемента из цепочки источника в цепочку приемник и уменьшает значение ECX на 1 и увеличивает/уменьшает значение адреса источника и приемника на 1

3) Какие операции выполняют строковые команды CMPS, SCAS? Какие особенности характерны для этих команд?

CMPS – сравнение двух цепочек, поиск несовпадающих/совпадающих элементов

SCAS – сканирование цепочки, поиск нужного элемента

4) Как обеспечить циклическую обработку строк?

Поставить REP/REPZ/REPNZ перед оператором обработки строки

5) Какова роль флага DF во флаговом регистре при выполнении команд обработки строк?

Направление обработки, 0 – вперед, 1 – назад.

6) Какие макрокоманды используются в среде RADASM для ввода и вывода строк?

- movs адрес\_приемника,адрес\_источника

- cmps адрес\_приемника,адрес\_источника

- scas адрес\_приемника

- lods адрес\_источника

- stos адрес\_приемника

В зависимости от типа адреса заменяются на одну из команд указанных выше

1. Как правильно выбрать тестовые данные для проверки алгоритма обработки строки?

Тестовые данные для тестирования и отладки программы необходимо выбрать в соответствии с ее особенностями и ключевыми моментами. Также необходимо учесть все важные аспекты работы программы и граничные значения.

**Вывод:**

Я изучил команды обработки цепочек и приемы обработки символьной информации.