**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №7

дисциплина: Архитектура вычислительных систем

тема: «Цепочечные команды»

Выполнил: ст. группы ПВ-31

Донцов А.А.

Проверил: Осипов О.В.

Белгород 2019

**Цель работы**: изучение цепочечных команд для работы с одномерными массивами и строками

**Задания для выполнения к работе:**



.486

.model flat, stdcall

option casemap: none

include c:\masm32\include\kernel32.inc

include c:\masm32\include\msvcrt.inc

includelib c:\masm32\lib\kernel32.lib

includelib c:\masm32\lib\msvcrt.lib

.data

str1 db 1024 dup(?)

swap db 20 dup(?)

format\_str db "%s", 0

.code

comparison\_word proc ; bool comparison\_word(char\* str1, char\* str2, int n)

PUSH ESI

PUSH EDI

PUSH ECX

MOV ESI, [ESP+6\*4]

MOV EDI, [ESP+5\*4]

MOV ECX, [ESP+4\*4]

REPE CMPSB

JA comp1; array\_1 > array\_2

JBE comp2; array\_1 <= array\_2

comp1:

MOV EAX, 1

jmp compend

comp2:

MOV EAX, 0

jmp compend

compend:

POP ECX

POP EDI

POP ESI

ret 12

comparison\_word endp

word\_symbol\_count proc

PUSH ESI

PUSH EDI

PUSH ECX

MOV EDI, [ESP+16]

MOV AL, ' ' ; завершающий байт

MOV ECX, 1024

REPNE SCASB

MOV EAX, EDI

SUB EAX, [ESP+16]

POP ECX

POP EDI

POP ESI

DEC EAX

ret 4

word\_symbol\_count endp

swap\_word proc ; swap(char\* a, char\* b, int n)

PUSH ESI

PUSH EDI

PUSH ECX

PUSH EBX

MOV EAX, [ESP+5\*4]

MOV EBX, [ESP+6\*4]

CLD

MOV ESI, EAX ; В ESI - адрес источника a

MOV EDI, offset swap ; В EDI - адрес приёмника

MOV ECX, [ESP+7\*4] ; Количество пересылок

REP MOVSB ; Копирование массива слов

MOV ESI, EBX ; В ESI - адрес источника b

MOV EDI, EAX ; В EDI - адрес приёмника a

MOV ECX, [ESP+7\*4] ; Количество пересылок

REP MOVSB ; Копирование массива слов

MOV ESI, offset swap ; В ESI - адрес источника swap

MOV EDI, EBX ; В EDI - адрес приёмника b

MOV ECX, [ESP+7\*4] ; Количество пересылок

REP MOVSB ; Копирование массива слов

POP EBX

POP ECX

POP EDI

POP ESI

ret 12

swap\_word endp

get\_word\_count proc

; Сохранить значения используемых регистров в стеке

PUSH EBX

PUSH ECX

PUSH EDI

; Поместить в EDI адрес обрабатываемой строки

MOV EDI, [ESP+16]

; Сначала нужно найти длину строки, поэтому нужно загрузить в AL искомый символ. Признак окончания строки - нулевой символ

MOV AL, 0

; Поместить в ECX -1, чтобы инициализировать его максимальным значением, т.к. число итераций не известно, а условием остановки цепочечной команды является нахождение первого нулевого байта

MOV ECX, -1

; Выполнить поиск символа, помещённого в AL

REPNE SCASB

; Теперь нужно вычислить длину строки

MOV ECX, EDI

; Адрес нулевого байта известен. Он находится в EDI. Адрес начала строки - в стеке

; Разница этих адресов и есть длина строки

SUB ECX, [ESP+16]

; EBX - счётчик количества слов

XOR EBX, EBX

; Поместить в EDI адрес начала строки

MOV EDI, [ESP+16]

; Поиск пробела

MOV AL, ' '

; Пропустить все пробелы в начале строки, т.е. выполнять команду, пока в ячейках памяти по адресу EDI пробелы

REPE SCASB

j1:

; Если ECX = 0, то конец алгоритма

JECXZ j\_end

; Условие остановки команды - ECX = 0 или найден пробел (разделитель слов)

REPNE SCASB

; Если между словами несколько пробелов, то нужно их все пропустить

REPE SCASB

; Увеличить количество слов на единицу

INC EBX

; Переход в начало цикла к следующему слову

JMP j1

j\_end:

; Поместить результат - количество слов в EAX

MOV EAX, EBX

; Восстановление регистров из стека

POP EDI

POP ECX

POP EBX

; Возврат из подпрограммы

RET 4

get\_word\_count endp

sort\_bubble proc ; sort(str, countword, sizeword)

PUSH EAX

PUSH EBX

PUSH ECX

PUSH EDI

XOR EBX,EBX ; Счётчик

sortj1:

MOV EDI, [ESP+5\*4] ; первое слово

MOV ESI, EDI

ADD ESI, [ESP+7\*4] ; второе слово

INC ESI

MOV ECX, [ESP+6\*4] ; Счётчик

DEC ECX

sortj2:

PUSH EDI

PUSH ESI

PUSH [ESP+7\*4] ; размер слов

call comparison\_word ; сравнение двух слов

CMP EAX, 1

JE swapj ; передаем на обмен

MOV EDI, ESI ; следующее слово

MOV ESI, EDI

ADD ESI, [ESP+7\*4] ; следующее слово

INC ESI

LOOP sortj2

INC EBX

CMP EBX,[ESP+6\*4]

JNE sortj1

jmp jmpend

swapj:

PUSH [ESP+7\*4] ; размер слов

PUSH ESI

PUSH EDI

call swap\_word

MOV EDI, ESI ; следующее слово

MOV ESI, EDI

ADD ESI, [ESP+7\*4] ; следующее слово

INC ESI

LOOP sortj2

INC EBX

CMP EBX,[ESP+6\*4]

JNE sortj1

jmp jmpend

jmpend:

POP EDI

POP ECX

POP EBX

POP EAX

ret 12

sort\_bubble endp

start:

PUSH offset str1

call crt\_gets

ADD ESP, 8

SUB ESP, 8

MOV EDI, ESP

PUSH offset str1

call get\_word\_count

MOV [EDI], EAX ; сохраняем количество слов

PUSH offset str1

call word\_symbol\_count

MOV [EDI+4], EAX ; сохраняем размер слов

PUSH [EDI+4]

PUSH [EDI]

PUSH offset str1

call sort\_bubble

ADD ESP, 8

PUSH offset str1

PUSH offset format\_str

call crt\_printf

ADD ESP, 16

call crt\_\_getch ; Задержка ввода с клавиатуры

push 0

call ExitProcess ; Выход из программы

end start