МИНОБРНАУКИ РОССИИ

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

Дисциплина: Архитектура ВС

Лабораторная работа № 7

Тема: «Цепочечные команды»

Выполнил:

ст. группы ПВ–31

Зановская А.И.

Проверил: Осипов О.В.

Белгород

2018

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

Вариант 12

Вывести все пары слов текста, которые отличаются только одной буквой.

.486

.386

.model flat, stdcall

option casemap: none

include C:\Users\Acer\Desktop\LABS\ABC\masm32\include\windows.inc

include C:\Users\Acer\Desktop\LABS\ABC\masm32\include\kernel32.inc

include C:\Users\Acer\Desktop\LABS\ABC\masm32\include\user32.inc

include C:\Users\Acer\Desktop\LABS\ABC\masm32\include\msvcrt.inc

includelib C:\Users\Acer\Desktop\LABS\ABC\masm32\lib\user32.lib

includelib C:\Users\Acer\Desktop\LABS\ABC\masm32\lib\kernel32.lib

includelib C:\Users\Acer\Desktop\LABS\ABC\masm32\lib\msvcrt.lib

.data

NAMEFLD db 1024 dup(?) ; введенные символы.

ACTLEN DD ? ; длина строки

COUNTWORD DD ? ; количество слов в строке

COUNTDUO DD 0 ; количество пар выведенных слов

n DD 0

npos DD 0 ; позиция первого считанного слова

rpos DD 0 ; позиция второго считанного слова

len1 DD ?

len2 DD ?

len DD 31

cm2\_bf db 31 dup(?) ; буфер для слова 1

cm1\_bf db 31 dup(?) ; буфер для слова 2

buf db 31 dup(?)

format\_str db "%d:%s %s", 13, 10, 0

.code

;----------------Функция для подсчёта длины строки

get\_str\_size proc

PUSH ECX

PUSH EDI

MOV EDI, [ESP+12] ; Поместить в EDI адрес обрабатываемой строки

MOV AL, 0 ; Найдем длину строки

MOV ECX, -1

REPNE SCASB ; Поиск символа, помещённого в AL

MOV ECX, EDI ;

SUB ECX, [ESP+12] ; Разница этих адресов и есть длина строки

MOV EAX, ECX

POP EDI

POP ECX

RET 4

get\_str\_size endp

;----------------Функция для подсчёта количества слов в строке, разделённых пробелами

get\_word\_count proc

PUSH EBX

PUSH ECX

PUSH EDI

mov ecx, ACTLEN

XOR EBX, EBX ; EBX - счётчик количества слов

MOV EDI, [ESP+16] ; Поместить в EDI адрес начала строки

MOV AL, ' ' ; Поиск пробела

REPE SCASB ; Пропустить все пробелы в начале строки

j1:

JECXZ j\_end ; Если ECX = 0, то конец алгоритма

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

REPE SCASB

INC EBX ; Увеличить количество слов на единицу

JMP j1 ; Переход в начало цикла к следующему слову

j\_end:

MOV EAX, EBX ; Поместить результат - количество слов в EAX

POP EDI

POP ECX

POP EBX

RET 4

get\_word\_count endp

;----------------Функция get word

get\_word proc

push ESI

push EDI

push EBX

MOV EAX, [ESP+4\*4]

MOV ESI, offset NAMEFLD ; В ESI - адрес источника

mov EBX, ESI

ADD ESI, EAX

MOV EDI, [ESP+4\*5] ; В EDI - адрес приёмника

MOV AL, ' '

MOV n, 0

cmp [ESI], AL

jbe repeatedSpace1

jmp repeatedSym1

repeatedSpace1:

inc ESI

cmp [ESI], AL

jbe repeatedSpace1

repeatedSym1:

MOVSB

ADD n, 1

cmp [ESI], AL

jg repeatedSym1

MOV byte ptr [EDI], 0

mov EAX, ESI

sub EAX, EBX ;запомним позицию после слова

pop EBX

pop EDI

pop ESI

ret 8

get\_word endp

;----------------Функция cmp\_word

cmp\_word proc

push esi

push edi

push eax

xor eax, eax

mov esi, offset cm1\_bf

mov edi, offset cm2\_bf

mov ecx, [ESP+4\*4]

pl:

cmpsb

jne incEbx

loop pl

jmp cmpRez

incEbx:

inc eax

jmp pl

je j\_end

cmpRez:

cmp eax, 1

je outWord

jne j\_end

outWord:

INC COUNTDUO

PUSH offset cm2\_bf

PUSH offset cm1\_bf

PUSH COUNTDUO

PUSH offset format\_str

CALL crt\_printf

ADD ESP, 16

jmp j\_end

j\_end:

pop eax

pop edi

pop esi

ret 4

cmp\_word endp

;----------------Функция clean\_word

clean\_word proc

PUSH ECX

PUSH EAX

MOV ecx, [ESP + 4\*3]

XOR eax, eax

MOV edi, [ESP + 4\*4]

REP stosd

POP EAX

POP ECX

RET 8

clean\_word endp

;----------------Функция main

main proc

push eax

push ebx

mov ecx, COUNTWORD ;цикл по всей строке

xor eax, eax

MOV npos, EAX ; начальная позиция, сначала равны нулю

MOV rpos, EAX ; для внутр. цикла, сначала равны нулю

DEC ECX

loop\_1:

mov n, 0

PUSH offset cm1\_bf

PUSH npos

CALL get\_word

MOV npos, EAX ; позиция пробела после первого считанного слова

MOV EBX, n

MOV len1, EBX

mov npos, eax

mov rpos, eax

PUSH ECX

loop\_2:

mov n, 0

PUSH offset cm2\_bf

PUSH rpos

CALL get\_word

MOV rpos, EAX

MOV EBX, n

MOV len2, EBX

mov eax, n

mov ebx, len1

cmp eax, ebx

je check

jmp next\_iter

check:

push ecx

PUSH len1

CALL cmp\_word

pop ecx

jmp next\_iter

next\_iter:

PUSH ECX

PUSH offset cm2\_bf

PUSH len2

CALL clean\_word

POP ECX

loop loop\_2

PUSH offset cm1\_bf

PUSH len1

CALL clean\_word

POP ECX;

DEC ecx

CMP ECX, 0

jne loop\_1

pop ebx

pop eax

ret

main endp

START:

;--------------Ввод строки с клавиатуры

PUSH offset NAMEFLD

CALL crt\_gets

ADD ESP, 4 ; Очистка стека от аргумента

;--------------Подсчёт длины исходной строки

PUSH offset NAMEFLD

CALL get\_str\_size

MOV ACTLEN, EAX

;--------------Вызов функции для подсчёта количества слов в строке

PUSH offset NAMEFLD

CALL get\_word\_count

MOV COUNTWORD, EAX

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

CMP EAX, 1

je exit

CALL main

exit:

call crt\_\_getch

push 0

call ExitProcess

end START

