Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Лабораторная работа № 5

по дисциплине ПнаЯА

Вариант 1

Выполнил студент гр. 250502: Бекетова М. А.

Проверил: Туровец Н. О.

Минск 2023

1 ТЕОРИТИЧЕСКИЕ СВЕДИНИЯ

Работа с файлами выполняется через функции DOS в стандартном порядке:

1) создание или открытие существующего файла;

2) выполнение файловых операций чтения или записи данных;

3) закрытие файла.

Дополнительно доступны операции: удаление, поиск и управление

Передача параметров программы через командную строку при запуске программы – решение, которое позволяет построить гибкую по входным данным программу без дополнительных диалогов с пользователем. При запуске программы DOS помещает всю командную строку в блок PSP запущенной программы по смещению 81h и ее длину в байт 80h.

2 КОД

.model small

.stack 100h

.data

noDataStr db "No data!", '$'

noFileStr db "The file with this name doesn't exist!", '$'

procStartStr db "File processing has started...", '$'

amountStr db "The amount of lines is:", 9h, '$'

procFinStr db "File processing has finished.", '$'

finStr db "Program executed successfully.", '$'

addAmountStr db " additional amount:", 9h, '$'

endl db 0Dh, 0Ah, '$'

sucMessage1 db "FOPEN: Success!", '$'

sucMessage2 db "RFF: Success!", '$'

sucMessage3 db "CEL: SB: Success!", '$'

sucMessage4 db "CEL: Success!", '$'

errMessage1 db "FOPEN: File not found!", '$'

errMessage2 db "FOPEN: Path not found!", '$'

errMessage3 db "FOPEN: Too many files are opened!", '$'

errMessage4 db "FOPEN: Access denied!", '$'

errMessage5 db "FOPEN: Wrong access mode!", '$'

errMessage6 db "RFF: error!", '$'

maxCmdLen equ 126

cmdLen dw ?

cmdLine db 126 dup(?)

fileDescr dw ?

bufSize equ 80

buf db bufSize+2 dup('$')

addAmount dw 0

nonEmptyLines dw 0

chars dw 0

cf dw 0

eof dw 0

.code

display macro string

pusha

displaySingle string

displaySingle endl

popa

endm

displaySingle macro string

lea dx, string

mov ah, 09h

int 21h

endm

exit macro

mov ax,4C00h

int 21h

endm

main:

mov ax, @data

mov ds, ax

call getComArgs ;get arguments

mov ax, cmdLen

cmp cmdLen, 1 ;if cmdLen <= 1 nodata

jle noData

display cmdLine

display procStartStr

lea dx, cmdLine

call fopen ;open file

cmp ax, 1 ;if it was not suc.

je endMain

call countNonEmptyLines

displaySingle amountStr

mov ax, nonEmptyLines

call printDecNumber

displaySingle endl

jmp endMain

noData:

display noDataStr

endMain:

display procFinStr

display finStr

exit

getComArgs proc

;returns: operation code in ax (0 - ok, 1 - no args)

push ax

push cx

mov cx, 0

mov cl, es:[80h] ;80h - cmd line length

mov cmdLen, cx

cmp cx, 1 ;if length <=1

jle endGCA ;go to end

cld ;clear direction flag

mov di, 81h ;81h - cmd itself

mov al, ' '

rep scasb ;repeat send byte while not end (cmp es:[di] and al while == or end)

dec di ;for ptr to the last smbl in cmd line

lea si, cmdLine ;load adress

skip:

mov al,es:[di]

cmp al, 0dh ;/n?

je endSkip

cmp al, 20h ;space?

je endSkip

cmp al, 9h ;tab?

je endSkip

mov ds:[si], al ;save smbl

inc di

inc si

jmp skip ;while != tab \n space

endSkip:

inc si ;addres + 1

mov ds:[si], word ptr '$';==end of string

endGCA:

pop cx

pop ax

ret

endp getComArgs

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

fopen proc

; requires: ds:dx - file name

; returns: ax - operation code (0 - ok, 1 - error)

; fileDescr - file descr

push cx

mov ah, 3dh ;open file

mov al, 0h ;for reading

int 21h

jc errorHandling ;if errors occured

mov fileDescr, ax ;save desc

jmp fopenOk ;if ok

errorHandling:

cmp al, 02h

jne pathNotFound

display errMessage1

jmp fopenError

pathNotFound:

cmp al, 03h

jne tooManyFilesOpened

display errMessage2

jmp fopenError

tooManyFilesOpened:

cmp al, 04h

jne accessDenied

display errMessage3

jmp fopenError

accessDenied:

cmp al, 05h

jne wrongAccessMode

display errMessage4

jmp fopenError

wrongAccessMode:

cmp al, 0Ch

jne fopenError

display errMessage5

jmp fopenError

fopenOk:

mov ax, 0

display sucMessage1

jmp fopenEnd

fopenError:

mov ax, 1

fopenEnd:

pop cx

ret

fopen endp

countNonEmptyLines proc

pusha

mov nonEmptyLines, 0

mov chars, 0

mov addAmount, 0

mov cf, 0

mov eof, 0

mov bx, fileDescr

handling:

call readFromFile ; fill buffer

cmp ax, 1

je finCounting

cmp cx, bufSize ; cx<bufSize -> not enough chars

jb lastOperation ; to fill the buf -> end of file

call scanBuf ; scan buf

jmp handling ; continue reading till eof (loop)

lastOperation:

mov eof, 1

call scanBuf

display sucMessage4

jmp finCounting

finCounting:

popa

ret

countNonEmptyLines endp

readFromFile proc

;requires: bx - file descr

;returns: ax - operation code (0 - ok, 1 - error)

; cx - amount of read bytes

mov ax, 0

mov ah, 3fh ;read from file

mov cx, 80 ;read 80 bytes if possible

lea dx, buf

int 21h

jc FRRerror ;if error

mov cx, ax

mov ax, 0

display sucMessage2

jmp FRRend

FRRerror:

display errMessage6

mov ax, 1

FRRend:

ret

readFromFile endp

scanBuf proc

lea si, buf

push ax

scan:

lodsb ; load character(mov al, ds:si; inc si)

cmp al, 13 ; \r?

je setCF

cmp al, 10 ; \n?

je handlEndl

cmp al, 20 ; space?

je nextB

inc chars

jmp nextB

setCF:

mov cf, 1

jmp nextB

handlEndl: ; (possible)

cmp cf, 1

jne nextB

mov cf, 0

;cmp chars, 0

;je nextB

mov chars, 0

inc nonEmptyLines

jo handleOverflow

jmp nextB

handleOverflow:

inc addAmount

mov ax, addAmount

call printDecNumber

displaySingle addAmountStr

dec nonEmptyLines

mov ax, nonEmptyLines

call printDecNumber

displaySingle endl

mov nonEmptyLines, 0

inc nonEmptyLines

nextB:

loop scan

cmp chars, 0

je endSB

cmp eof, 0

je endSB

mov chars, 0

inc nonEmptyLines

jo handleOverflow

endSB:

display sucMessage3

pop ax

ret

scanBuf endp

printDecNumber proc

;ax contains result

pusha

xor cx, cx

mov bx, 10

loop1:

xor dx, dx

div bx

push dx

inc cx

cmp ax, 0

ja loop1

loop2:

pop dx

add dx, 30h

mov ah, 2

int 21h

loop loop2

displaySingle endl

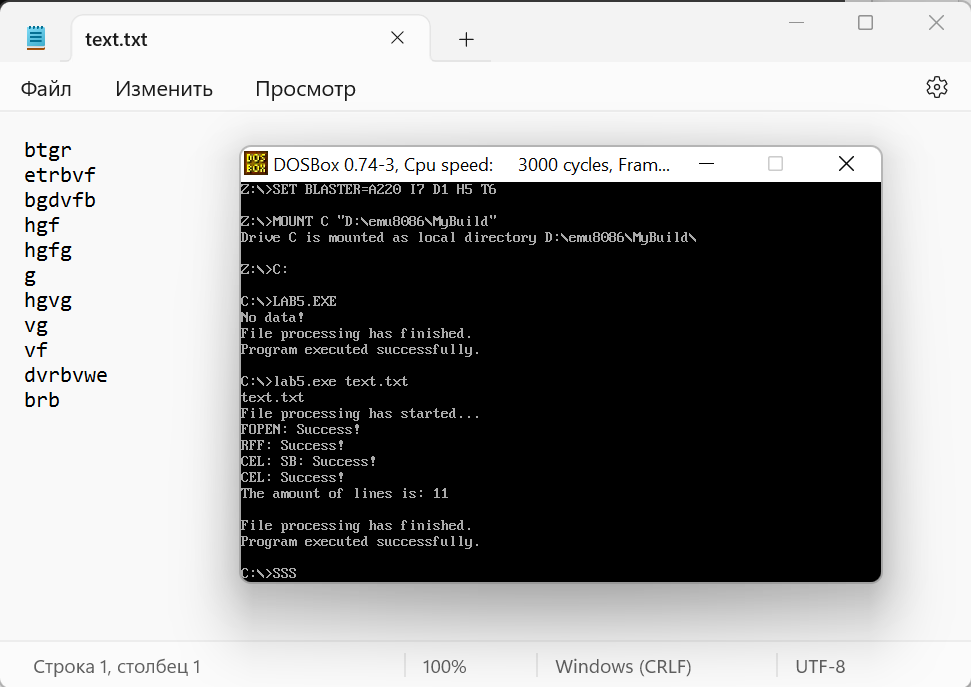
popa

ret

printDecNumber endp

end main

3 ДЕМОНСТРАЦИЯ РАБОТЫ



4 ВЫВОДЫ

Таким образом была реализована программа, которая считает число строк в файле, в которых нет указанного слова. В результате удалось ознакомиться с основными операциями обработки файлов, получить понятие о работе с параметрами командной строки.