Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Конструирование программ

Отчёт по лабораторной работе №8

РАБОТА С ФАЙЛАМИ

Студент Кончик Д.С.

Принимающий Романюк М.В.

Минск 2022

**СОДЕРЖАНИЕ**

[1 Цель работы 3](#_Toc118909172)

[2 Задание 4](#_Toc118909173)

[3 Результат выполнения 11](#_Toc118909174)

[4 Вывод 13](#_Toc118909175)

**1 ЦЕЛЬ РАБОТЫ**

Ознакомиться с основными операциями обработки файлов.

Получить понятие о работе с параметрами командной строки.

2 ЗАДАНИЕ

Вариант 12. В выходной файл поместить только те строки входного файла, которые не содержат все указанные символы.

Листинг 1 ­– Код программы

; Распечатать строку [Макрос]

printStr macro str

push ax

push dx

mov ah,09h ; Номер функции DOS

lea dx,str ; Смещение str в DX

int 21h ; 21 прерывание

pop dx

pop ax

endm

readChars MACRO id, number, buf

mov ah,3Fh ; Функция DOS (чтение из файла)

mov bx,id ; Загрузка дескриптора

mov cx,number ; Количество считываемых чисволов

lea dx,buf ; Загрузка буфера

int 21h ; Считать данные

jc readingErrorLabel

endm

writeChars MACRO id, number, buf

mov ah,40h ; Функция DOS (запись в файл)

mov bx,id ; Дескриптор

mov cx,number ; Количество записываемых символов

lea dx,buf ; Буфер

int 21h ; Записать данные

jc writingErrorLabel

endm

setCursor MACRO id, distance, type

mov ah,42h ; Функция DOS (переместить указатель чтения/записи)

mov bx,id ; Идентификатор

mov cx,0h ;

mov dx,distance ;

mov al,type ; Относительно чего

int 21h ; Переместить указатель

jc cursorErrorLabel

endm

closeFile MACRO id

mov bx,id ; Дескриптор

mov ah,3Eh ; Функция DOS 3Eh

int 21h ; Закрыть файл

jc closeErrorLabel

endm

deleteFile MACRO nameOffset, labelGood, labelBad

mov ah,41h ; Функция DOS 41h (удалить файл)

mov dx,nameOffset ; Адрес ASCIZ-строки с именем файла

int 21h ; Удалить файл

jc deleteErrorLabel

endm

createFile MACRO nameOffset, id

mov ah,3Ch ; Функция DOS 3Ch (создать файл)

mov cx,00100000b ; Атрибут файла

mov dx,nameOffset ; Адрес ASCIZ-строки с именем файла

int 21h ; Создать файл

mov id,ax ; Сохранить дескриптор

jc createErrorLabel

endm

openFile MACRO nameOffset, id

mov dx,nameOffset ; Адрес ASCIZ-строки с именем файла

mov ah,3Dh ; Функция DOS 3Dh (открыть существующий файл)

mov al,02h ; 02 - для чтения и записи

int 21h ; Открыть файл

mov id,ax ; Сохранить дескриптор

jc openErrorLabel

endm

.model tiny

.code

org 100h

START:

mov end\_SP,sp ; Сохранение SP

; Обработка параметров командной строки

printStr paramsMessage ; "Параметры командной строки:"

printStr toNewLine ; На новую строку

printStr paramsFormat ; Формат параметров

printStr toNewLine

printStr toNewLine

printStr paramsYourMessage ; Ваши параметры

printStr toNewLine

call processCommandLine ; Обработать параметры командной строки

cmp paramsNumber,03h ; Параметров должно быть три

je paramsAreNormal

; Проблемы с параметрами

printStr toNewLine

printStr paramsError ; Ошибка

jmp exit ; Выход из программы

; С параметрами всё нормально

paramsAreNormal:

openFile source\_file\_offset, source\_file\_ID ; Открыть исходный файл

lea dx, temp\_file\_name ; Задать смещение имени

mov temp\_file\_offset, dx ; временного файла

createFile temp\_file\_offset,temp\_file\_ID ; Создать временный файл

createFile destination\_file\_offset,destination\_file\_ID

; Главный цикл

mainWhile:

; Считывание порции символов

readChars source\_file\_ID, buffer\_dose, buffer

; Поиск симола 0Dh в считанном буфере

search0Dh:

mov charsRead,ax

mov cx,charsRead ; Количество считанных символов

mov al,0Dh ; Символ 0Dh

lea di,buffer ; Смещение буфера

repne scasb ; Сравниваем, пока не равны

jz Exists0Dh ; ZF = 1?

; Действие, если нет символа 0Dh

NotExists0Dh:

mov dx,charsRead ; Число символов в порции =

mov charsOfStringInDose,dx ; = число считанных символов

mov isEndOfString,00h ; Конца строки нет

; Поиск символа NULL в считанном буфере

mov cx,charsRead ; Количество считанных символов

mov al,0h ; Символ NULL

lea di,buffer ; Смещение буфера

repne scasb ; Сравниваем, пока не равны

jnz writeCharsInTempFile ; ZF <> 0?

; Действие, если есть символ NULL

dec charsOfStringInDose ; charsOfStringInDose--

mov isEndOfString,01h ; Конец строки

mov isEndOfFile,01h ; Конец файла

jmp writeCharsInTempFile

; Действие, если есть символ 0Dh

Exists0Dh:

; Определения числа символов строки в порции

; charsOfStringInDose = charsRead - CX + 1

mov dx,charsRead

mov charsOfStringInDose,dx

sub charsOfStringInDose,cx

inc charsOfStringInDose

mov isEndOfString,01h

; Кейс когда .. .. .. .. 0Dh, порция закончилась

; то есть в порцию не влезло 0Ah

mov bx,charsOfStringInDose

mov buffer[bx-1],0Ah

; Запись символов во временный файл

writeCharsInTempFile:

writeChars temp\_file\_ID,charsOfStringInDose,buffer ; Записать символы

mov dx,charsOfStringInDose

add charsInString,dx

; Проверка, конец ли строки

cmp isEndOfString,0h

je notEndOfString

; Действие, если в порции есть конец строки

endOfString:

; Установка курсора в source\_file

mov bx,charsRead

mov cursorOffset,bx

mov bx,charsOfStringInDose

sub cursorOffset,bx

neg cursorOffset

setCursor source\_file\_ID,cursorOffset,01h ; Установить курсор

; Проверка, подходит ли строка из файла под условие

; "не содержать ни одного символа из заданных"

setCursor temp\_file\_ID,00h,00h ; Курсор на начало файла

mov dx,charsInString ; charsInStringCopy =

mov charsInStringCopy,dx ; charsInString

mov dx,buffer\_dose ; Сравнить

cmp charsInStringCopy,dx ; charsInStringCopy и buffer\_dose

jbe skipCheckCycle ; if charsInStringCopy <= buffer\_dose

; Цикл сравнения порциями

checkCycle:

readChars temp\_file\_ID buffer\_dose, buffer ; Считать символы

checkBufferMacro buffer\_dose ; Проверить

cmp contains,01h

je resetTempfile ; Сбросить файл

mov dx,buffer\_dose ;

sub charsInStringCopy,dx ; charsInStringCopy -= buffer\_dose

mov dx,buffer\_dose ; Сравнение charsInStringCopy

cmp charsInStringCopy,dx ; и buffer\_dose

ja checkCycle ; charsInStringCopy > buffer\_dose?

; charsInStringCopy <= buffer\_dose

skipCheckCycle:

readChars temp\_file\_ID charsInStringCopy, buffer ; Считать символы

checkBufferMacro charsInStringCopy ; Проверить

cmp contains,01h

je resetTempfile ; Сбросить файл

; Пересылка строки из temp\_file в destination\_file

; Проверка на условие "не содержать символы" пройдена

transferString:

setCursor temp\_file\_ID,00h,00h ; Курсор на начало файла

mov dx,buffer\_dose ;

cmp charsInString,dx ;

jbe skipTransferCycle ; charsInString <= buffer\_dose?

; Цикл пересылки строки порциями

transferCycle:

readChars temp\_file\_ID buffer\_dose, buffer

writeChars destination\_file\_ID buffer\_dose, buffer

mov dx,buffer\_dose ;

sub charsInString,dx ; charsInString -= buffer\_dose

mov dx,buffer\_dose ; Сравнение charsInString

cmp charsInString,dx ; и buffer\_dose

ja transferCycle ; charsInString > buffer\_dose?

; charsInString <= buffer\_dose

skipTransferCycle:

readChars temp\_file\_ID charsInString, buffer

writeChars destination\_file\_ID charsInString, buffer

; Сбросить временный файл

resetTempfile:

mov charsInString,0h ; charsInString = 0

setCursor temp\_file\_ID,00h,00h ; Курсор на начало файла

; Действие, если в порции нет конца строки

notEndOfString:

mov isEndOfString,0h ; Не конец строки

cmp isEndOfFile,01h ; Сравнение

jne mainWhile ; Конец файла?

closeFile temp\_file\_ID ; Закрыть временный файл

deleteFile temp\_file\_offset ; Удалить временный файл

jmp exit

openErrorLabel:

printStr openError

jmp exit

createErrorLabel:

printStr createError

jmp exit

readingErrorLabel:

printStr readingError

jmp exit

writingErrorLabel:

printStr writingError

jmp exit

cursorErrorLabel:

printStr cursorError

jmp exit

closeErrorLabel:

printStr closeError

jmp exit

deleteErrorLabel:

printStr deleteError

jmp exit

; Завершение программы

exit:

mov sp,end\_SP ; Восстановить SP

ret ; Завершение COM-программы

; [Процедура] – результат в contains

; Есть ли символы в порции строки

checkBuffer PROC near

mov contains,0h

mov si,0h ; SI = 0 [счетчик]

for1:

mov dx,0h ; DX = 0

mov dl,buffer[si] ; DX = текущий символ из порции

mov si,0h ; SI = 0 [счетчик]

mov di,symbolsOffset ; DI = смещение на начало символов

for2:

cmp dl,byte ptr [di] ; Сравнение символа из порции с

jne not\_equals ; символом из набора символов

mov contains,01h ; contains = 1

pop si ; Восстановление SP

ret ; Выход из процедуры

; Не равен

not\_equals:

; Условие выхода из цикла

inc di ; DI++

cmp byte ptr [di],0 ; Если не конец массива

jne for2

pop si ; Восстановление SI

; Условие выхода из цикла

inc si ; SI++

cmp si,bufferNumberCheck ; Сравнение

jb for1 ; if si < bufferNumberCheck

ret

checkBuffer endp

; [Макрос] – есть ли символы в порции строки

checkBufferMacro MACRO number

mov dx,number

mov bufferNumberCheck,dx

call checkBuffer

endm

; Распечатать строку по смещению [Макрос]

printStrByOffset macro offset

push ax

push dx

mov ah,09h ; Номер функции DOS

mov dx,offset ; Смещение в DX

int 21h ; 21 прерывание

pop dx

pop ax

endm

; [Процедура] – обработка командной строки

processCommandLine PROC near

cld ; Для команд строковой обработки

mov bp,sp ; Сохранить текущую вершину стека в BP

mov cl,[80h] ; Размер командной строки

cmp cl,1 ; Проверка длины командной строки

jle end ; выход из программы

mov cx,0FFFFh ; Для команд работы со строками

mov di,81h ; Начало командной строки в ES:DI

find\_param:

mov al,' ' ; AL = пробел

repz scasb ; Искать не пробел

dec di ; DI = адрес начала параметра

push di ; Сохранить адрес в стек

mov paramOffset,di ; Для будущей операции вывода параметра

inc paramsNumber ; Увеличить paramsNumber на 1

mov si,di ; SI = DI для следующей команды lodsb

scan\_params:

lodsb ; Прочитать символ из параметра

cmp al,0Dh ; Если 0Dh – последний параметр

je params\_ended ; Параметры закончились

cmp al,20h ; Сравнение с пробелом

jne scan\_params ; Текущий параметр не закончился

dec si ; SI = первый байт после параметра

mov [si],'$' ; Записать в него $

printStrByOffset paramOffset ; Распечатать параметр

printStr toNewLine ; Переход на новую строку

mov [si],0 ; Записать в него 0

mov di,si ; DI = SI для команды scasb

inc di ; DI = следующий после нуля символ

jmp find\_param ; Продолжить разбор

params\_ended:

dec si ; SI = первый байт после конца

mov [si],'$' ; Записать в него $

printStrByOffset paramOffset ; Распечатать параметр

printStr toNewLine ; Переход на новую строку

mov [si],0 ; Записать в него 0

; Достаем смещения из стека

pop\_from\_stack:

pop symbolsOffset

pop destination\_file\_offset

pop source\_file\_offset

end:

mov sp,bp ; Восстановить SP

ret ; Выход из процедуры

processCommandLine endp

; Данные программы

end\_SP dw 0 ; Адрес SP для ret в конце

source\_file\_ID dw 0

source\_file\_offset dw 0

source\_file\_error db 0Ah, 0Dh,'Error opening the source file$'

destination\_file\_ID dw 0

destination\_file\_offset dw 0

temp\_file\_name db 'd:\temp.txt',0 ; Временный файл

temp\_file\_ID dw 0 ; Дескриптор

temp\_file\_offset dw 0

readingError db 0Ah, 0Dh,'Reading error$'

writingError db 0Ah, 0Dh,'Writing error$'

cursorError db 0Ah, 0Dh,'Cursor installation error$'

closeError db 0Ah, 0Dh,'File closing error$'

deleteError db 0Ah, 0Dh,'File deletion error$'

createError db 0Ah, 0Dh,'File creation error$'

openError db 0Ah, 0Dh,'File opening error$'

buffer db 100 DUP(0) ; Буфер данных

buffer\_dose dw 50d ; Размер порции

isEndOfFile db 0 ; [bool] – конец ли файла

isEndOfString db 0 ; [bool] – конец ли строки

charsRead dw 0 ; Считано символов

charsOfStringInDose dw 0 ; Символов строки в порции

charsInString dw 0 ; Символов в строке

cursorOffset dw 0 ; Смещение курсора

symbolsOffset dw 0 ; Смещение на символы, которых быть не должно

bufferNumberCheck dw 0 ; Количество символов из буфера для проверки

charsInStringCopy dw 0 ; Символов в строке (копия)

contains db 0

paramsNumber db 0 ; Число параметров командной стркои

paramOffset dw 0 ; Смещение параметра

toNewLine db 0Ah, 0Dh, '$'

paramsMessage db 'Command line parameters:$'

paramsYourMessage db 'Your parameters:$'

paramsError db 'You need to enter three parameters$' ;

paramsFormat db '1 - source file name (exists)',0Ah,0Dh,$

end START

3 РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ

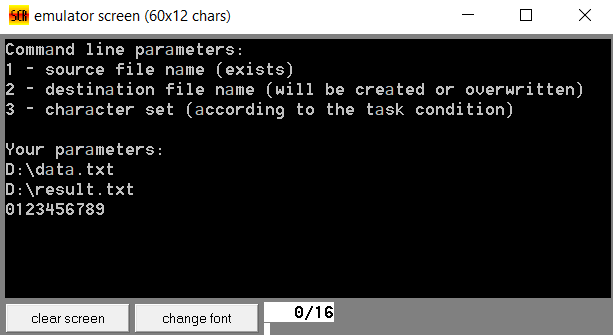


Рисунок 1 – Вывод параметров командной строки

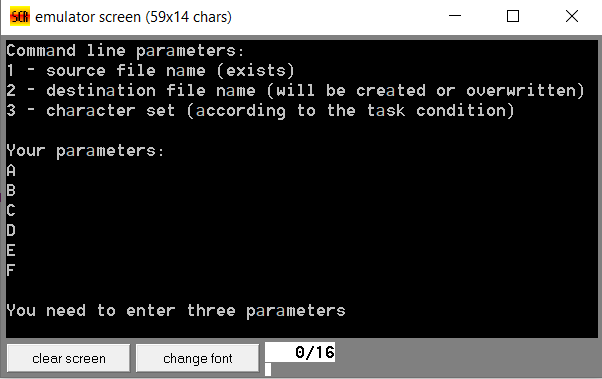


Рисунок 2 – Обработка некорректных параметров командной строки

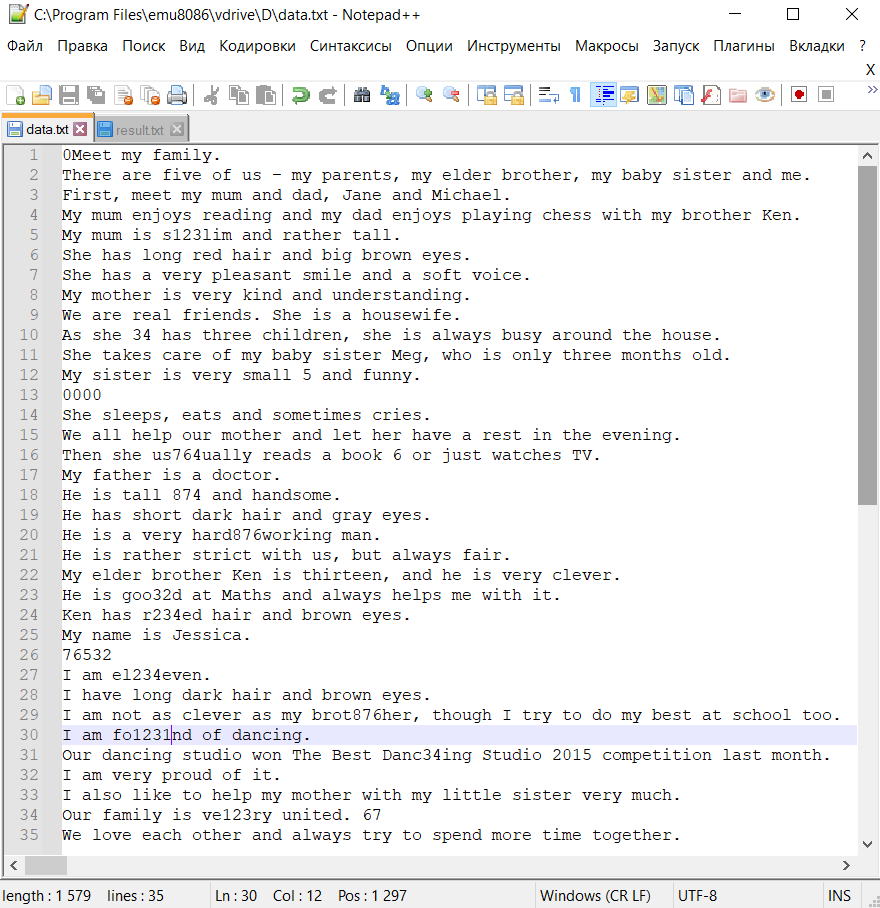


Рисунок 3 – Входной файл

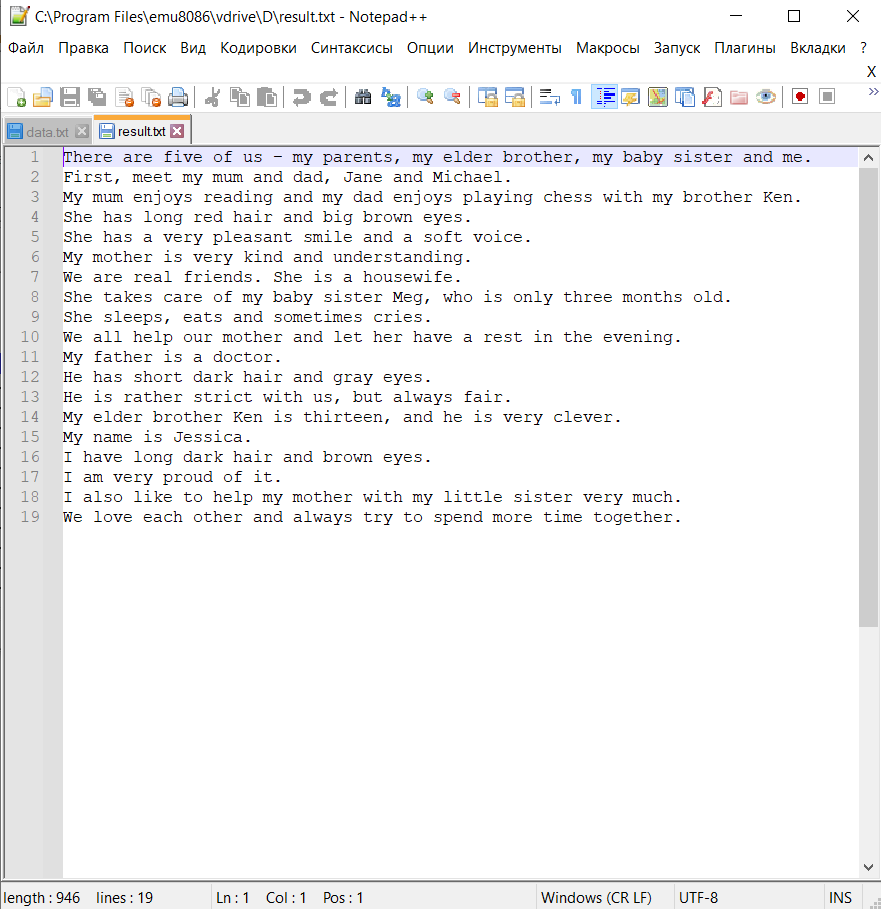


Рисунок 4 – Выходной файл (удалены строки, содержащие цифры)

4 ВЫВОД

Таким образом, была создана программа, которая помещает в выходной файл только те строки входного файла, которые не содержат все указанные символы. Параметры обработки (названия файлов и набор символов) принимаются через параметры командной строки. Также программа обрабатывает некорректный ввод параметров.