Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Конструирование программ»

ОТЧЕТ

к лабораторной работе №6

на тему:

«РАБОТА С ФАЙЛАМИ.»

БГУИР 1-40-04-01

Выполнил студент группы 253504
ЖГУТОВ Евгений Дмитриевич

_____
(дата, подпись студента)

Проверил ассистент кафедры информатики
РОМАНЮК Максим Валерьевич

_____
(дата, подпись преподавателя)

Минск 2023

Цель работы: Задание 1. Вариант 7. Подсчитать число строк в файле, в которых есть указанное слово.

Ход работы: на рисунке 1 представлены изначальные вводимые данные, на рисунке 2 – результат работы программы.

Листинг 1 – Исходный код программы задания 1

```
.model small
.stack 100h
.data
    crlf db 0Dh, 0Ah, '$'
    word db 10, 0, 15 dup('$'), '$'
    buf db 2, 0, 2 dup('$'), '$'
    msg_prompt_to_input db 'Enter the word to
search', 0Dh, 0Ah, '$'
    msg_result db 0Dh, 0Ah, 'Count of strings: $'
    msg_bad db 0Dh, 0Ah, 'Buffer overflow$'
    msg_bad_args db 'Error: Unable to parse command
line arguments', 0Dh, 0Ah, '$'
    msg_empty_args db 'Error: No command line
arguments provided', 0Dh, 0Ah, '$'
    msg_error db 0Dh, 0Ah, 'Error', 0Dh, 0Ah, '$'
    word_capacity equ 50
    word_buffer db word_capacity + 2 dup(0)
    cmd_capacity equ 127
    cmd_length db ?
    cmd_text db cmd_capacity dup('$')
    file_path db cmd_capacity dup('$')

.code


is_empty macro str, is_0
    push si
    lea si, str
    call strlen
    pop si

    cmp ax, 0
    je is_0
endm


output macro str
    push ax
```

```
    push dx
    lea dx, str
    mov ah, 9
    int 21h
    pop dx
    pop ax
endm


input macro str
    push bx
    push cx
    push dx

again:
    mov ah, 0Ah
    lea dx, str
    int 21h

    xor ax, ax
    xor cx, cx

    mov cl, [word + 1]
    cmp cl, 0
    je again

    pop dx
    pop cx
    pop bx
endm

puti macro
    local put1
    local put2
    local ex

    push ax
    push cx
    push -1
    mov cx, 10
put1:
    xor dx, dx
    xor ah, ah
    div cl
    mov dl, ah
```

```
    push dx
    cmp al, 0
    jne put1

    mov ah, 2

put2:
    pop dx
    cmp dx, -1
    je ex
    add dl, '0'
    int 21h
    jmp put2
ex:
    mov dl, ' '
    int 21h
    pop cx
    pop ax
endm

fopen macro
    lea dx, file_path
    mov ah, 3Dh
    mov al, 00h
    int 21h
    jc exit

    mov bx, ax
endm

fclose macro
    mov ah, 3Eh
    int 21h
endm

fread macro
    local continue
    push ax
    push cx
    push dx

    mov cx, 1
    lea dx, buf

    mov ah, 3Fh
```

```
    int 21h
    jc exit

    mov cx, ax
    test cx, cx
    jnz continue
    fclose

    jmp good_exit

continue:
    pop dx
    pop cx
    pop ax
endm


parse_cmd_text proc
    push bx
    push cx
    push dx

    mov cl, cmd_length
    xor ch, ch

    lea si, cmd_text
    lea di, file_path
    call to_asciiz

    is_empty file_path, bad_cmd_args

    lea di, word_buffer
    call to_asciiz

    is_empty word_buffer, good_cmd_args

bad_cmd_args:
    output msg_bad_args
    mov ax, 1
    jmp end_parse_cmd_text

good_cmd_args:
    mov ax, 0

end_parse_cmd_text:
```

```
    pop bx
    pop cx
    pop bx
    ret
parse_cmd_text endp

;
to_asciiz proc
    push ax
    push cx
    push di

    parse_to_asciiz:
        mov al, ds:[si]
        cmp al, ' '
        je is_delimeter
        cmp al, 0Dh
        je is_delimeter
        cmp al, 09h
        je is_delimeter
        cmp al, 0Ah
        je is_delimeter
        cmp al, 00h
        je is_delimeter
        cmp al, '$'
        je is_delimeter

        mov es:[di], al
        inc di
        inc si
    loop parse_to_asciiz

is_delimeter:
    mov al, 00h
    mov es:[di], al
    mov al, '$'
    inc di
    mov es:[di], al
    inc si


    pop di
    pop cx
    pop ax
    ret
```

```
to_asciiz endp

strlen proc
    push bx
    push si

    xor ax, ax
start_strlen:
    mov bl, ds:[si]
    cmp bl, 00h
    je end_strlen
    inc si
    inc ax
    jmp start_strlen
end_strlen:
    pop si
    pop bx
    ret
strlen endp

count_raws:
    xor dx, dx

    search:
        fread
        mov al, [word+2]
        mov cl, [buf]
        cmp cl, al
        je check_word
    jmp search

check_word:
    lea si, word+2
    mov al, [si]
    mov ah, 1

    while:
        inc ah
        inc si
        mov al, [si]
        mov cl, [word+1]
        cmp ah, cl
        jg success
        fread
        mov cl, [buf]
```

```
        cmp al, cl
        jne search
    je while

success:
    inc dx

skip:
    fread
    mov al, 13
    mov cl, [buf]
    cmp al, cl
    jne skip
    fread
    jmp search

    jmp count_raws_end

start:
    mov ax, @data
    mov es, ax
    xor ch, ch
    mov cl, ds:[80h]
    mov cmd_length, cl
    mov si, 82h
    lea di, cmd_text
    rep movsb
    mov ds, ax

    call parse_cmd_text
    test ax, ax
    jne exit

    output msg_prompt_to_input
    input word
    fopen
    output msg_result
    jmp count_raws

count_raws_end:

exit:
    output msg_error
    pop dx
    pop cx
```

```
    pop ax
    mov ax, 4c00h
    int 21h
good_exit:
    output crlf
    pop dx
    pop cx
    pop ax
    mov ax, dx
    puti
    mov ax, 4c00h
    int 21h
end start
```

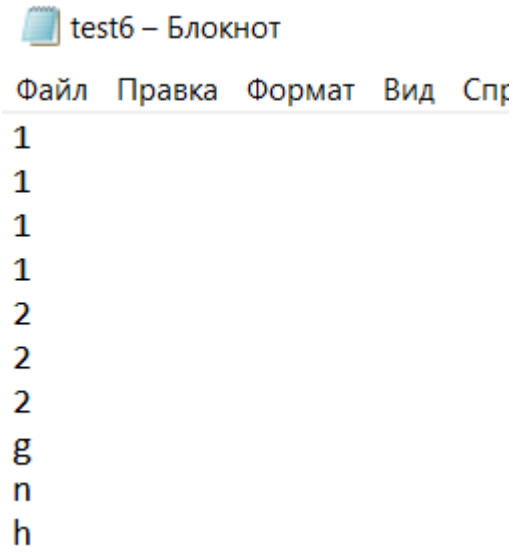

Рисунок 2 – Изначальные вводимые данные

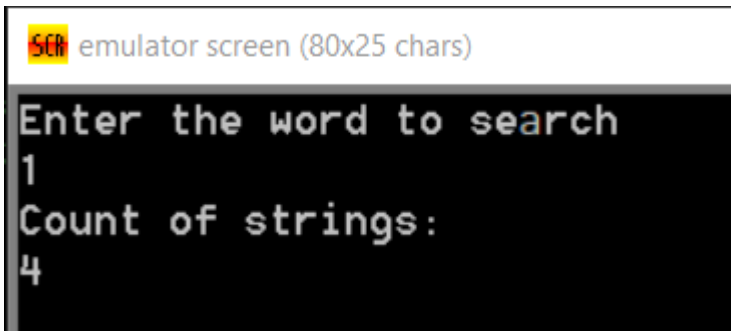

Рисунок 1 – Результат работы программы

Выводы: В результате лабораторной работы была выполнена одна задача с использованием основными операциями обработки файлов.