Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

К ЗАЩИТЕ ДОПУСТИТЬ

\_\_\_\_\_\_\_\_\_\_ Н.О. Туровец

Отчет

Лабораторная работа №7

Выполнил студент гр. 150505: Горбачевский К.В

Проверил: Туровец Н.О.

Минск 2022

**СОДЕРЖАНИЕ**

[Введение 3](#_gjdgxs)

[Код программы 4](#_30j0zll)

[Вывод в консоли после компиляции 5](#_1fob9te)

[Заключение](#_tyjcwt) 6

# ВВЕДЕНИЕ

**Тема работы:** Загрузка и выполнение программ. Работа с памятью.

**Цель работы:** Ознакомиться с загрузкой и выполнением программ, рассмотреть работу с памятью.

**Вариант лабораторной работы:** Написать программу для вывода на экран нескольких строк символов.

**Код программы**

.MODEL tiny

.CODE

START:

mov ax, @data

mov ds, ax

mov es, ax

call PARSE\_CMD

xor cx, cx

mov cl, TIMES\_TO\_RUN

cmp cl, 0

je EXIT

mov sp, PROGRAMM\_LENGTH + 300h

mov ah, 4ah

STACK\_SHIFT = PROGRAMM\_LENGTH + 300h

mov bx, STACK\_SHIFT shr 5

int 21h

mov ax, cs

mov word ptr EPB + 4, ax

mov word ptr EPB + 8, ax

mov word ptr EPB + 0ch, ax

lea dx, TIMES\_TO\_RUN\_STR

mov ah, 9

int 21h

RUN\_PROGRAM\_CYCLE:

mov ax, 4b00h

lea dx, PROGRAM\_NAME

lea bx, EPB

int 21h

jc FATAL\_ERROR

loop RUN\_PROGRAM\_CYCLE

int 20h

EXIT:

mov ah, 4ch

int 21h

FATAL\_ERROR:

mov ah, 9

cmp al, 2

je SET\_FILE\_NOT\_FOUND

cmp al, 5

je SET\_ACCESS\_DENIED

cmp al, 0bh

je SET\_NOT\_VALID\_FORMAT

SHOW\_MESSAGE:

int 21h

jmp EXIT

SET\_FILE\_NOT\_FOUND:

lea dx, FILE\_NOT\_FOUND\_MESSAGE

jmp SHOW\_MESSAGE

SET\_ACCESS\_DENIED:

lea dx, ACCESS\_DENIED\_MESSAGE

jmp SHOW\_MESSAGE

SET\_NOT\_VALID\_FORMAT:

lea dx, NOT\_VALID\_FORMAT\_MESSAGE

jmp SHOW\_MESSAGE

PARSE\_CMD proc

mov cl, [80h]

cmp cl, 0

je EMPTY\_PARAMETER

mov di, 82h

call SKIP\_SPACES

lea si, PROGRAM\_NAME

call WRITE\_PARAMETR

call SKIP\_SPACES

lea si, TIMES\_TO\_RUN\_STR

call WRITE\_PARAMETR

call TO\_DIGIT

lea si, PARAMETER\_FOR\_RUN\_PROGRAM

call SKIP\_SPACES

dec di

WRITE\_PARAMETER\_FOR\_RUN\_PROGRAM:

cmp [di], 0dh

je EXIT\_FROM\_PARSE

mov al, [di]

mov [si], al

inc si

inc di

jmp WRITE\_PARAMETER\_FOR\_RUN\_PROGRAM

EXIT\_FROM\_PARSE:

ret

EMPTY\_PARAMETER:

mov ah, 9

lea dx, EMPTY\_PARAMETER\_MESSAGE

int 21h

mov ah, 4ch

int 21h

PARSE\_CMD endp

WRITE\_PARAMETR proc

WRITE\_CYCLE:

mov al, [di]

cmp al, 0

je END\_OF\_WRITE

cmp al, ' '

je END\_OF\_WRITE

cmp al, 9

je END\_OF\_WRITE

cmp al, 0dh

je END\_OF\_WRITE

mov [si], al

inc di

inc si

jmp WRITE\_CYCLE

END\_OF\_WRITE:

ret

WRITE\_PARAMETR endp

SKIP\_SPACES proc

dec di

CYCLE:

inc di

cmp [di], 0dh

je END\_OF\_SKIPPING

cmp [di], 0

je END\_OF\_SKIPPING

cmp [di], ' '

je CYCLE

cmp [di], 9

je CYCLE

END\_OF\_SKIPPING:

ret

SKIP\_SPACES endp

TO\_DIGIT proc

push di

call IS\_VALID\_DIGIT

lea di, TIMES\_TO\_RUN\_STR

xor bx, bx

xor ax, ax

mov bl, 10

TO\_DIGIT\_CYCLE:

mov cl, [di]

sub cl, '0'

mul bx

add al, cl

inc di

cmp [di], '$'

jne TO\_DIGIT\_CYCLE

pop di

mov TIMES\_TO\_RUN, al

ret

TO\_DIGIT endp

IS\_VALID\_DIGIT proc

lea di, TIMES\_TO\_RUN\_STR

xor cx, cx

mov cl, 4

FIND\_SIZE\_CYCLE:

cmp [di], '$'

je EXIT\_FROM\_CYCLE

cmp [di], '0'

jl NOT\_VALID\_DIGIT

cmp [di], '9'

jg NOT\_VALID\_DIGIT

inc di

loop FIND\_SIZE\_CYCLE

EXIT\_FROM\_CYCLE:

lea di, TIMES\_TO\_RUN\_STR

cmp cl, 4

je NOT\_VALID\_DIGIT

cmp cl, 1

jl NOT\_VALID\_DIGIT

je IS\_LOWER

ret

IS\_LOWER:

cmp [di], '2'

jg NOT\_VALID\_DIGIT

cmp [di + 1], '5'

jg NOT\_VALID\_DIGIT

cmp [di + 2], '5'

jg NOT\_VALID\_DIGIT

ret

NOT\_VALID\_DIGIT:

mov ah, 9

lea dx, NOT\_VALID\_DIGIT\_MESSAGE

int 21h

mov ah, 4ch

int 21h

IS\_VALID\_DIGIT endp

LINE db 128 dup ('$')

TIMES\_TO\_RUN db 0

TIMES\_TO\_RUN\_STR db 4 dup('$')

PROGRAM\_NAME db 128 dup(0)

PARAMETER\_FOR\_RUN\_PROGRAM 128 dup(0dh)

EPB dw 0000

dw offset PARAMETER\_FOR\_RUN\_PROGRAM

dw 005ch, 0, 006ch, 0

NOT\_VALID\_DIGIT\_MESSAGE db "Digit must be in [0, 255]$"

EMPTY\_PARAMETER\_MESSAGE db "You should enter name of program and quantity of starts$"

FILE\_NOT\_FOUND\_MESSAGE db "Executable file not found$"

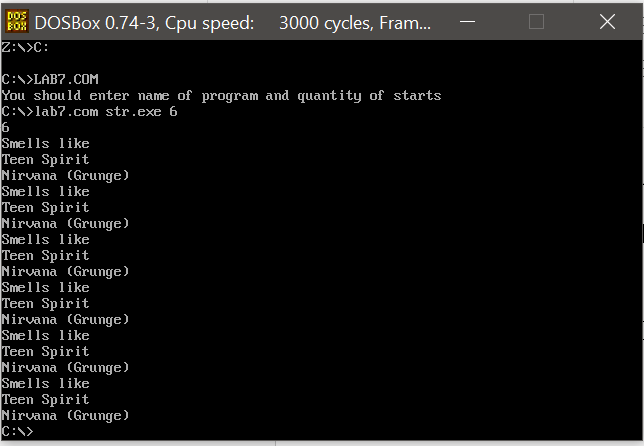
ACCESS\_DENIED\_MESSAGE db "Access denied$"

NOT\_VALID\_FORMAT\_MESSAGE db "Not valid format message$"

PROGRAMM\_LENGTH EQU $ - START

end START

**Вывод в консоли после компиляции**



**Заключение**

Цель лабораторной работы достигнута. Изучена структура программы на ЯП ассемблер и написана первая программа. Изучены два основные формата исполняемых файлов в DOS — com и exe. Отработана базовая команда пересылки данных команда MOV.