

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по лабораторной работе №6
по дисциплине «Операционные системы»
Тема: Построение модуля динамической структуры

Студент гр.7383

Ханова Ю.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2019

Цель работы.

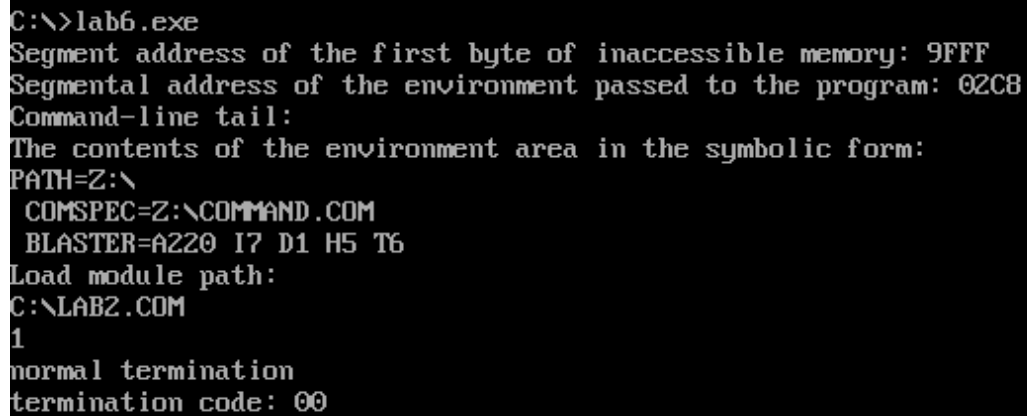
Исследование возможности построения загрузочного модуля динамической структуры.

Описание функций и структур данных.

Название функции	Назначение
BYTE_TO_HEX	переводит число из AL в 2 16-ых символа и помещает их в AL и BH
PRINT	вызывает функцию печати строки
PATH	выполняет подготовку параметров для запуска загрузочного модуля
PATH_PAR	выполняет создание блок параметров
LOAD	выполняет запуск загрузочного модуля
ERRORS	функция обработки ошибок
CODE_ERR	функция вывода причины и кода завершения загрузочного модуля

Ход работы.

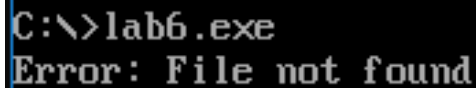
- Запуск программы, когда она находится в текущем каталоге с разработанным модулем lab2.exe и последующий ввод случайного буквенного символа на рис.1:



```
C:\>lab6.exe
Segment address of the first byte of inaccessible memory: 9FFF
Segmental address of the environment passed to the program: 02C8
Command-line tail:
The contents of the environment area in the symbolic form:
PATH=Z:\
COMSPEC=Z:\COMMAND.COM
BLASTER=A220 I7 D1 H5 T6
Load module path:
C:\LAB2.COM
1
normal termination
termination code: 00
```

Рис.1

- Пример запуска программы, когда модуль и программа находятся в разных каталогах на рис.2:



```
C:\>lab6.exe
Error: File not found
```

Рис.2

Выводы.

В процессе выполнения данной лабораторной работы была исследована возможность построения загрузочного модуля динамической структуры.

Ответы на контрольные вопросы.

1. Как реализовано прерывание CTRL+C?

Вектор прерывания 23h, находящийся по адресу 0000:008Ch, содержит адрес, по которому DOS передает управление после обнаружения нажатия пользователем клавиш Ctrl-C.

Обычная системная обработка Ctrl-C сводится к немедленному снятию программы.

2. В какой точке заканчивается вызываемая программа, если код завершения 0?

При выполнении функции 4Ch прерывания int 21h;

3. В какой точке заканчивается вызываемая программа по прерыванию Ctrl+C?

В точке вызова функции 01h прерывания int 21h.

ПРИЛОЖЕНИЕ А

lab6.asm

```
ODE SEGMENT
    ASSUME CS:CODE, DS:DATA, ES:DATA, SS:ASTACK
START: JMP BEGIN

PRINT PROC
    push ax
    mov AH,09h
    int 21h
    pop ax
    ret
PRINT ENDP

;-----
TETR_TO_HEX PROC near
    and AL,0Fh
    cmp AL,09
    jbe NEXT
    add AL,07
NEXT: add AL,30h
    ret
TETR_TO_HEX ENDP

;-----
BYTE_TO_HEX PROC near
    push CX
    mov AH,AL
    call TETR_TO_HEX
    xchg AL,AH
    mov CL,4
    shr AL,CL
    call TETR_TO_HEX
    pop CX
    ret
BYTE_TO_HEX ENDP

;-----
PATH PROC
    mov ax,ASTACK
    sub ax,CODE
    add ax,100h
    mov bx,ax
    mov ah,4ah
    int 21h
    jnc PATH_skip1
        call ERRORS
PATH_skip1:

    ; подготавливаем блок параметров:
    call PATH_PAR
```

```

; определяем путь до программы:
push es
push bx
push si
push ax
mov es,es:[2ch] ; в es сегментный адрес среды
mov bx,-1
ENV:
    add bx,1
    cmp word ptr es:[bx],0000h
    jne ENV
add bx,4
mov si,-1
PUT_ZIKL:
    add si,1
    mov al,es:[bx+si]
    mov PROGR[si],al
    cmp byte ptr es:[bx+si],00h
    jne PUT_ZIKL

add si,1
PUT_ZIKL2:
    mov PROGR[si],0
    sub si,1
    cmp byte ptr es:[bx+si],'\ '
    jne PUT_ZIKL2
add si,1
mov PROGR[si],'l'
add si,1
mov PROGR[si],'a'
add si,1
mov PROGR[si],'b'
add si,1
mov PROGR[si],'2'
add si,1
mov PROGR[si], '.'
add si,1
mov PROGR[si], 'c'
add si,1
mov PROGR[si], 'o'
add si,1
mov PROGR[si], 'm'
pop ax
pop si
pop bx
pop es

ret

```

PATH ENDP

PATH_PAR PROC

```
    mov ax, es:[2ch]
    mov ARG, ax
    mov ARG+2, es ; Сегментный адрес параметров командной строки(PSP)
    mov ARG+4, 80h ; Смещение параметров командной строки
    ret
```

PATH_PAR ENDP

LOAD PROC

```
    mov ax, ds
    mov es, ax
    mov bx, offset ARG
```

```
    mov dx, offset PROGR
```

```
    mov KEEP_SS, SS
    mov KEEP_SP, SP
```

```
    mov ax, 4B00h
    int 21h
```

```
    push ax
    mov ax, DATA
    mov ds, ax
    pop ax
    mov SS, KEEP_SS
    mov SP, KEEP_SP
    jnc LOAD_skip1
        call ERRORS
        jmp LOAD_end
LOAD_skip1:
```

```
    call CODE_ERR
```

```
LOAD_end:
```

```
    ret
```

LOAD ENDP

ERRORS PROC

```
    mov dx, offset er
    call PRINT
```

```
    mov dx, offset er1
    cmp ax, 1
```

```

        je osh_pechat
        mov dx,offset er2
        cmp ax,2
        je osh_pechat
        mov dx,offset er7
        cmp ax,7
        je osh_pechat
        mov dx,offset er8
        cmp ax,8
        je osh_pechat
        mov dx,offset er9
        cmp ax,9
        je osh_pechat
        mov dx,offset er10
        cmp ax,10
        je osh_pechat
        mov dx,offset er11
        cmp ax,11
        je osh_pechat

osh_pechat:
        call PRINT
        mov dx,offset STRENDL
        call PRINT

        ret
ERRORS ENDP

CODE_ERR PROC
        ; получаем в al код завершения, в ah - причину:
        mov al,00h
        mov ah,4dh
        int 21h

        mov dx, offset end0
        cmp ah, 0
        je CODE_ERR_pech_1
        mov dx,offset end1
        cmp ah,1
        je CODE_ERR_pech
        mov dx,offset end2
        cmp ah,2
        je CODE_ERR_pech
        mov dx,offset end3
        cmp ah,3
        je CODE_ERR_pech

CODE_ERR_pech_1:
        call PRINT

```



```

    mov dx,offset STRENDL
    call PRINT
    mov dx, offset end_cod

CODE_ERR_pech:
    call PRINT

    cmp ah,0
    jne CODE_ERR_skip

; печать кода завершения:
    call BYTE_TO_HEX
    push ax
    mov ah,02h
    mov dl,al
    int 21h
    pop ax
    mov dl,ah
    mov ah,02h
    int 21h
    mov dx,offset STRENDL
    call PRINT
CODE_ERR_skip:

    ret
CODE_ERR ENDP
;-----
BEGIN:
    mov ax,data
    mov ds,ax
    call PATH
    call LOAD
    xor AL,AL
    mov AH,4Ch
    int 21H
CODE ENDS
; ДАННЫЕ
DATA SEGMENT
    er db 'Error: $'
    er1 db 'Function number is incorrect$'
    er2 db 'File not found$'
    er7 db 'Control memory block destroyed$'
    er8 db 'insufficient memory$'
    er9 db 'Invalid memory block address$'
    er10 db 'Invalid environment string$'
    er11 db 'incorrect format$'

; причины завершения
end0 db 'normal termination$'

```

```

end1 db 'termination by Ctrl-Break$'
end2 db 'termination by errors$'
end3 db 'termination by function 3lh$'
end_cod db 'termination code: $'

STRENDL db 0DH,0AH,'$'

; блок параметров
ARG    dw 0 ; сегментный адрес среды
        dd 0 ; сегмент и смещение командной строки
        dd 0 ; сегмент и смещение первого FCB
        dd 0 ; сегмент и смещение второго FCB

; путь и имя вызываемой программы
PROGR db 40h dup (0)

KEEP_SS dw 0
KEEP_SP dw 0
DATA ENDS

ASTACK SEGMENT STACK
        dw 100h dup (?)
ASTACK ENDS

END START

```