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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Операционные системы»

Тема: Исследование интерфейсов программных модулей

Студент(ка) гр. 9382 _____ Голубева В.П.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Исследование интерфейса управляющей программы и загрузочных модулей. Этот интерфейс состоит в передаче запускаемой программе управляющего блока, содержащего адреса и системные данные. Так загрузчик строит префикс сегмента программы (PSP) и помещает его адрес в сегментный регистр. Исследование префикса сегмента программы (PSP) и среды, передаваемой программе.

Задание.

Шаг 1. Для выполнения лабораторной работы необходимо написать и отладить программный модуль типа .COM, который выбирает и распечатывает следующую информацию:

1) Сегментный адрес недоступной памяти, взятый из РЭР, в шестнадцатеричном виде.

2) Сегментный адрес среды, передаваемой программе, в шестнадцатеричном виде.

3) Хвост командной строки в символьном виде.

4) Содержимое области среды в символьном виде.

5) Путь загружаемого модуля.

Сохраните результаты, полученные программой, и включите их В отчет.

Шаг 2. Оформление отчета В соответствии с требованиями. В отчет включите скриншот с запуском программы и результатами.

Необходимые сведения для составления программы

При начальной загрузке программы формируется PSP, который размещается в начале первого сегмента программы. PSP занимает 256 байт и располагается с адреса, кратного границе сегмента. При загрузке модулей типа .COM все сегментные регистры указывают на адрес РЭР. При загрузке модуля типа .EXE сегментные регистры DS и ES указывают на РЭР. Именно

по этой причине значения этих регистров в модуле .EXE следует переопределять.

Формат PSP:

Смещение	Длина поля(байт)	Содержимое поля
0	2	int 20h
2	2	Сегментный адрес первого байта недоступной памяти. Программа не должна модифицировать содержимое памяти за этим адресом.
4	6	Зарезервировано
0Ah (10)	4	Вектор прерывания 22h (IP,CS)
0Eh (14)	4	Вектор прерывания 23h (IP,CS)
12h (18)	4	Вектор прерывания 24h (IP,CS)
2Ch (44)	2	Сегментный адрес среды, передаваемой программе.
5Ch		Область форматируется как стандартный неоткрытый блок управления файлом (FCB)
6Ch		Область форматируется как стандартный неоткрытый блок управления файлом (FCB). Перекрывается, если FCB с адреса 5Ch открыт.
80h	1	Число символов в хвосте командной строки.
81h		Хвост командной строки - последовательность символов после имени вызываемого модуля.

Область среды содержит последовательность символьных строк вида: имя=параметр. Каждая строка завершается байтом нулей.

В первой строке указывается имя COMSPEC, которая определяет используемый командный процессор и путь к COMMANDCOM. Следующие строки содержат информацию, задаваемую командами PATH, PROMPT, SET.

Среда заканчивается также байтом нулей. Таким образом, два нулевых байта являются признаком конца переменных среды. Затем идут два байта, содержащих 00h, 01h, после которых располагается маршрут загруженной программы. Маршрут также заканчивается байтом 00h.

Выполнение работы.

В ходе выполнения работы была выполнена последовательность действий, описанная в задании, в результате чего была написана программа, извлекающая из PSP нужную информацию. В работоспособности программы можно убедиться, взглянув на Рис. 1 и Рис. 2.

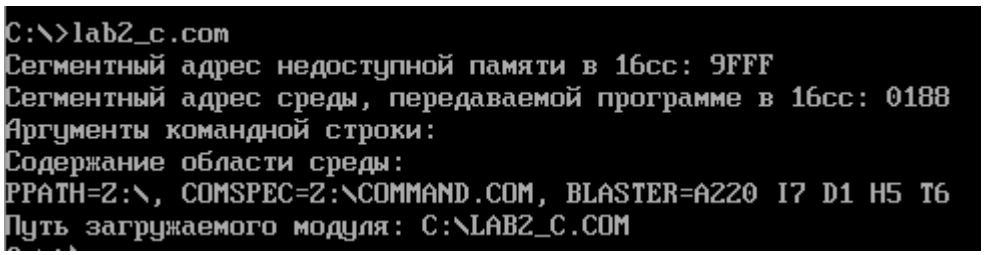

Рис. 1. Вывод программы без аргументов командной строки

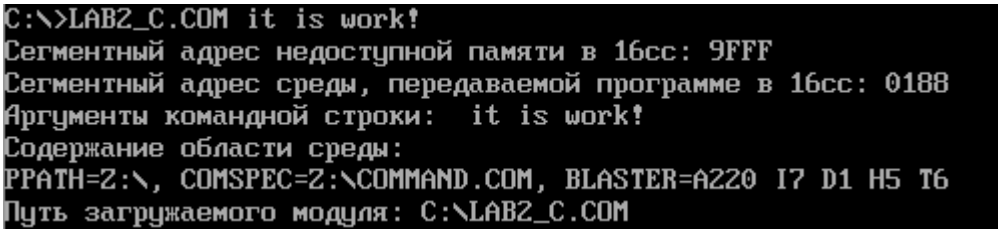

Рис. 2. Вывод программы с аргументами командной строки

Ответы на контрольные вопросы

Контрольные вопросы по лабораторной работе № 2.

Сегментный адрес недоступной памяти

1) На какую область памяти указывает адрес недоступной памяти?

Ответ: на первый байт в опреативной памяти, определенный под программу.

2) Где расположен этот адрес по отношению области памяти, отведенной программе?

Ответ: расположен сразу за концом памяти, выделенной программе.

3) Можно ли в эту область памяти писать?

Ответ: можно, но так как в DOS это не запрещено, но потребуется использовать аресацию ддля сегментного регистра.

Среда передаваемая программе

1) Что такое среда?

Ответ: это набор переменных, в которых хранятся некоторые настройки операционной системы.

2) Когда создается среда? Перед запуском приложения или в другое время?

Ответ: во время загрузки DOS.

3) Откуда берется информация, записываемая в среду?

Ответ: копируется из файла autoexec.bat, который расположен в корневом каталоге загрузочного устройства

Выводы.

Был исследован интерфейс управляющей программы и загрузочных модулей. Написана программа, извлекающая информацию из недоступной памяти и выводящая её на экран.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
LAB2 SEGMENT
    ASSUME CS:LAB2, DS:LAB2, ES:NOTHING, SS:NOTHING
ORG 100H
    START: JMP BEGIN
s_address db 'Сегментный адрес недоступной памяти в 16cc:     ',
0DH, 0AH, '$';43
s_envir db 'Сегментный адрес среды, передаваемой программе в
16cc:     ',  0DH, 0AH, '$';54
s_tail db 'Аргументы командной строки: ', '$';28
s_contain db 'Содержание области среды: ', 0DH, 0AH, '$'
s_path db 'Путь загружаемого модуля: ', '$'
s_ent db ' ', 0DH, 0AH, '$'

;procedures
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
;byte AL translate in two symbols on 16cc numbers in AX
    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
;-------------------------------
WRD_TO_HEX PROC near
;translate in 16cc a 16 discharge number
;in AL - number, DI - the address of the last symbol
    push BX
    mov BH,AH
    call BYTE_TO_HEX
    mov [DI],AH
    dec DI
    mov [DI],AL
    dec DI
    mov AL,BH
    call BYTE_TO_HEX
    mov [DI],AH
    dec DI
    mov [DI],AL
    pop BX
ret
WRD_TO_HEX ENDP

enter proc near
      mov DX, offset s_ent
    mov ah, 09h
    int 21h
    ret
enter endp

BEGIN:

address:
    mov ax, ds:[2h]
    mov di, offset s_envir-4
    call WRD_TO_HEX
```

```
    mov dx, offset s_address
    mov ah, 09h
    int 21h

environment:
    mov ax, ds:[2Ch]
    push ax

    mov di, offset s_tail-4
    call WRD_TO_HEX

    mov dx, offset s_envir
    mov ah, 09h
    int 21h

tail_command_line:
    mov dx, offset s_tail
    mov ah, 09h
    int 21h

    mov ch, ds:[80h]
    mov cl, 0h
    xor di, di

loop_str:
    cmp ch, cl
    jle contain_environment_area


    mov dl, ds:[81h+di]
    inc di
    mov ah, 02h
    int 21h
    dec ch

    jmp loop_str
```

```
contain_environment_area:
    call enter
    mov dx, offset s_contain
    mov ah, 09h
    int 21h

    mov bx, ds


    pop ax
    push bx

    mov ds, ax

    xor di, di

    mov dl, [di]
    mov ah, 02h
    int 21h

    jmp loop_area

e:
    inc di
    mov dl, 2Ch
    mov ah, 02h
    int 21h

    mov dl, 20h
    mov ah, 02h
    int 21h

loop_area:

    cmp word ptr [di], 0000h
    je path
    cmp byte ptr [di], 00h
```

```
    je e

    mov dl, [di]
    mov ah, 02h
    int 21h
    inc di

    jmp loop_area

path:

    pop bx
    mov ax, ds
    push ax
    mov ds, bx

    add di, 4h
    call enter
    mov dx, offset s_path
    mov ah, 09h
    int 21h

    pop bx
    mov ds, bx

loop_path:

    cmp byte ptr [di], 00h
    je end_pro

    mov dl, [di]
    mov ah, 02h
    int 21h

    inc di
    jmp loop_path
```

```
end_pro:
    xor AL,AL
    mov AH,4Ch
    int 21H
LAB2    ENDS
          END START
```