**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Операционные системы»**

Тема: Исследование интерфейсов программных модулей

|  |  |  |
| --- | --- | --- |
| Студент гр. 9383 |  | Орлов Д.С. |
| 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.

**Порядок выполнения работы**

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 PSP, 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.

**Выполнение работы**

В процессе выполнения лабораторной работы была разработана .COM-программа, в которой:

* MEMORY\_S db 'Memory segment: ',0DH,0AH,'$' – сегментный адрес недоступной памяти.
* ENV\_ADRESS db 'Environment address: ',0DH,0AH,'$' – сегментный адрес среды.
* TAIL db 'Tail of command line: ','$' – хвост командной строки.
* EMPTY\_T db 'Tail of command line: EMPTY','$' – пустой хвост.
* ENV\_CONTENT db 0DH,0AH,'Environment scope content:',0DH,0AH, '$' – содержимое области среды.
* PATH db 'Path: ',0DH,0AH, '$' – путь загружаемого модуля.

Изображение выглядит как текст

Автоматически созданное описание

Рис. 1. Пример работы программы без аргументов

Изображение выглядит как текст

Автоматически созданное описание

Рис. 2. Пример работы программы с аргументами

**Ответы на вопросы**

**Сегментный адрес недоступной памяти**

1. На какую область памяти указывает адрес недоступной памяти?

Адрес недоступной памяти указывает на основную оперативную память, то есть на первый байт после памяти, выделенной программе.

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

Адрес расположен сразу после области памяти, отведенной программе (начиная с адреса 9FFF).

3. Можно ли в эту область памяти писать?

Можно, т. к. DOS не имеет механизмов защиты от перезаписи памяти различными программами, которая им не отведена.

**Среда, передаваемая программе**

1. Что такое среда?

Среда — область памяти, хранящая информацию о состоянии системы. Например, настройки, путь к домашней директории компьютера и т.д.

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

Изначальная среда создается при запуске ОС. Эта среда копируется (и после чего может измениться) в адресное пространство запущенной программы.

3. Откуда берется информация, записываемая в среду?

Информация берется из системного файла AUTOEXEC.BAT, который расположен в корневом каталоге загрузочного устройства.

**Вывод.**

В ходе лабораторной работы были получены навыки работы с PSP, а также изучены интерфейсы программных модулей DOS.

**Приложение А**

TESTPC SEGMENT

ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING

ORG 100H

START: JMP BEGIN

; Данные

MEMORY\_S db 'Memory segment: ',0DH,0AH,'$'

ENV\_ADRESS db 'Environment address: ',0DH,0AH,'$'

TAIL db 'Tail of command line: ','$'

EMPTY\_T db 'Tail of command line: EMPTY','$'

ENV\_CONTENT db 0DH,0AH,'Environment scope content:',0DH,0AH, '$'

END\_STRING db 0DH,0AH, '$'

PATH db 'Path: ',0DH,0AH, '$'

; Процедуры

;-----------------------------------------------------

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

;байт в AL переводится в два символа шест. числа в AX

push CX

mov AH,AL

call TETR\_TO\_HEX

xchg AL,AH

mov CL,4

shr AL,CL

call TETR\_TO\_HEX ;в AL старшая цифра

pop CX ;в AH младшая

ret

BYTE\_TO\_HEX ENDP

;-------------------------------

WRD\_TO\_HEX PROC near

;перевод в 16 с/с 16-ти разрядного числа

; в AX - число, DI - адрес последнего символа

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

;-------------------------------

MEMORY\_DEFINITION PROC near

mov ax, ds:[02h]

mov di, offset MEMORY\_S

add di, 19

call WRD\_TO\_HEX

mov dx, offset MEMORY\_S

mov AH,09h

int 21h

ret

MEMORY\_DEFINITION ENDP

ENV\_ADRESS\_DEFINITION PROC near

mov ax, ds:[2Ch]

mov di, offset ENV\_ADRESS

add di, 24

call WRD\_TO\_HEX

mov dx, offset ENV\_ADRESS

mov AH,09h

int 21h

ret

ENV\_ADRESS\_DEFINITION ENDP

TAIL\_DEFINITION PROC near

mov cx, 0

mov cl, ds:[80h]

cmp cl, 0 ;если пусто

je empty\_tail

mov dx, offset TAIL

mov AH,09h

int 21h

mov di, 0

mov ax, 0

read\_tail:

mov dl, ds:[81h+di]

mov ah, 02h

int 21h

inc di

loop read\_tail

ret

empty\_tail:

mov dx, offset EMPTY\_T

mov AH,09h

int 21h

ret

TAIL\_DEFINITION ENDP

CONTENT\_DEFINITION PROC near

mov dx, offset ENV\_CONTENT

mov AH,09h

int 21h

mov di, 0

mov ds, ds:[2Ch]

read\_str:

cmp byte ptr [di], 0

je end\_str

mov dl, [di]

mov ah, 02h

int 21h

jmp find\_end

end\_str:

cmp byte ptr [di+1],00h

je find\_end

push ds

mov cx, cs

mov ds, cx

mov dx, offset END\_STRING

mov AH,09h

int 21h

pop ds

find\_end:

inc di

cmp word ptr [di], 0001h

je read\_path

jmp read\_str

read\_path:

push ds

mov ax, cs

mov ds, ax

mov dx, offset PATH

mov AH,09h

int 21h

pop ds

add di, 2

loop2:

cmp byte ptr [di], 0

je break

mov dl, [di]

mov ah, 02h

int 21h

inc di

jmp loop2

break:

ret

CONTENT\_DEFINITION ENDP

; Код

BEGIN:

call MEMORY\_DEFINITION

call ENV\_ADRESS\_DEFINITION

call TAIL\_DEFINITION

call CONTENT\_DEFINITION

xor AL,AL

mov AH,4Ch

int 21H

TESTPC ENDS

END START; конец модуля, START - точка выхода