**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 гр.8382 _____ Ершов М.И.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2020

**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 стоит префикс сегмента программы (PSP) и помещает его адрес в сегментный регистр. Исследование префикса сегмента программы (PSP) и среды, передаваемой программе.

**Ход работы**

Был написан и отлажен программный модуль типа **.COM**, который выбирает и распечатывает следующую информацию:

1. Сегментный адрес недоступной памяти
2. Сегментный адрес среды, передаваемой программе
3. Хвост командной строки
4. Содержимое области среды
5. Путь загружаемого модуля

Результат выполнения программы продемонстрирован на рисунке 1.


Рисунок 1

**Ответы на контрольные вопросы**

**Сегментный адрес недоступной памяти**

1) На какую область памяти указывает адрес недоступной памяти?

На область памяти после программы.

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

За областью, выделенной программе.

3) Можно ли в эту область памяти писать?

Да, ограничений нет.

**Среда, передаваемая программе**

1) Что такое среда?

Набор из именованных переменных.

2) Когда создаётся среда? Перед запуском приложения или в другой момент?

Нет, в другой момент (во время загрузки операционной системы MS-DOS).

3) Откуда берётся информация, записываемая в среду?

Информация берется из родительской среды (создается при помощи файла AUTOEXEC.BAT).

**Вывод**

В ходе выполнения лабораторной работы был изучен интерфейс управляющей программы и интерфейс загрузочных модулей, а так-же исследованы префикс сегмента программы и среда, передаваемая программе.

# ПРИЛОЖЕНИЕ А
# ИСХОДНЫЙ КОД CODE.ASM

```
TESTPSP SEGMENT
         ASSUME CS:TESTPSP, DS:TESTPSP, ES:NOTHING, SS:NOTHING
         ORG 100H
START: JMP MAIN

INACCESSIBLE_MEMORY_title db 'INACCESSIBLE MEMORY: ', 10, 13, '$'
INACCESSIBLE_MEMORY db '0000', 10, 13, '$'
ENV_ADDRESS_title db 'ENV ADDRESS: ', 10, 13, '$'
PATH_title db 'PATH: ', 10, 13, '$'
ARGUMENTS_title db 'ARGUMENTS: ', 10, 13, '$'
ENV_ADDRESS db '0000', 10, 13, '$'
CRLF db 10, 13, '$'

EXIT PROC near
      xor AL, AL
      mov AH, 4ch
      int 21h
      ret
EXIT ENDP

PRINT_STR PROC near
      push ax
      mov ah, 09h
      int 21h
      pop ax
      ret
PRINT_STR 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

BYTE_TO_DEC PROC near
	push cx
	push dx
	xor ah, ah
	xor dx, dx
	mov cx, 10
	loop_bd:
		div cx
		or dl, 30h
		mov [si], dl
		dec si
		xor dx, dx
		cmp ax, 10
		jae loop_bd
		cmp al, 00h
		je end_l
		or al, 30h
		mov [si], al
	end_l:
		pop dx
		pop cx
	ret
BYTE_TO_DEC ENDP

MAIN:

	mov dx, offset INACCESSIBLE_MEMORY_title
	call PRINT_STR

	mov dx, es:[2h];es указывает на PSP, 2h - смещение для первого байта недоступной памяти
```

```
mov al, dh
mov si, offset INACCESSIBLE_MEMORY
call BYTE_TO_HEX
mov [si], ax
mov al, dl
call BYTE_TO_HEX
mov [si+2], ax

mov dx, offset INACCESSIBLE_MEMORY
call PRINT_STR

mov dx, offset ENV_ADDRESS_title
call PRINT_STR

mov dx, es:[2ch];2ch - смещение сегментного адреса среды
mov al, dh
call BYTE_TO_HEX
mov si, offset ENV_ADDRESS
mov [si], ax
mov al, dl
call BYTE_TO_HEX
mov [si+2], ax

mov dx, offset ENV_ADDRESS
call PRINT_STR

xor cx, cx
mov cl, es:[80h];число символов в хвосте командной строки
mov si, 81h;хвост командной строки

mov ah, 2h
cmp cl, 0
je print_env
mov dx, offset ARGUMENTS_title
call PRINT_STR

print_CLtail:;посимвольная печать хвоста командной строки
      mov dl, [si]
      int 21h
      inc si
      loop print_CLtail
```

```
      mov dx, offset CRLF;перевод строки
      call PRINT_STR

print_env:

      mov es, es:[2ch]
      mov dl, es:[0]
      mov si, 0
      ;int 21h
      mov cx, 1
      print_env_area:;посимвольная печать содержимого области среды
            ;inc si
            mov dl, es:[si]
            cmp dl, 0
            je stop_print
            int 21h
            inc si

            jmp print_env_area

    stop_print:
      mov dx, offset CRLF
      call PRINT_STR



      add si, cx;инкрементировали si
      cmp cx, 0
      je print_blaster
      cmp cx, 2
      je print_path
      cmp cx, 3
      je finish
      mov cx, 0
      add si, cx
      jmp print_env_area

      print_blaster:

      mov dl, es:[si+1]
```

```
	int 21h
	mov cx, 2
	add si, cx
	jmp print_env_area

	print_path:

	mov cx, 3
	add si, 2

	jmp print_env_area

	finish:
		call EXIT
TESTPSP ENDS
END START
```