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.

Студентка гр. 8382 _____ Наконечная А. Ю.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург

2020

Цель работы.

Исследовать интерфейс управляющей программы и загрузочных модулей.

Постановка задачи.

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 стоки в символьном виде.
4. Содержимое области среды в символьном виде.
5. Путь загружаемого модуля.

Выполнение работы.

Была создана функция MAIN, которая позволяет распечатать информацию о: сегментном адресе недоступной памяти, сегментном адресе среды, хвосте командной строки, содержимом области среды, пути загружаемого модуля.

Для вывода информации были объявлены следующие строки:

```
1)MEM_STR db 'Inaccessible memory     ', 0DH,0AH,'$'
2)ENV_ADRESS_STR db 'Environment segment adress     ', 0DH,0AH,'$'
3)TAIL_STR db 'Command line tail        ',0DH,0AH,'$'
4)EMPTY_TAIL_STR db 'Command line tail is empty', 0DH,0AH,'$'
5)ENV_CONTENT_STR db 'Symbolic content of the environment area ',
0DH,0AH,'$'
6)END_PRINT db 0DH,0AH,'$'
7)PATH_STR db 'Module path ', 0DH,0AH,'$'
```

В результате выполнения была получена информация, представленная на рисунке 1:

Рисунок 1 — результат выполнения программы

Ответы на контрольный вопросы.

Сегментный адрес недоступной памяти

1) На какую область памяти указывает адрес недоступной памяти?

Адрес указывает на последний параграф доступной памяти.

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

После области памяти, которая выделена программе.

3) Можно ли в эту область памяти писать?

Да, так как программа имеет доступ ко всей адресуемой памяти.

Среда передаваемая программе

1) Что такое среда?

Это область памяти, которая хранит информацию в виде символьных строк (ИМЯ = ПАРАМЕТР). Например, среди строк можно найти переменную PATH, которая определяет путь к каталогу, в котором хранится исполняемый файл.

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

Когда одна программа запускает другую программу или когда COMMAND.COM запускает программу, то запущенная программа получает свой собственный экземпляр блока среды, который, по умолчанию, является точной копией среды родителя. При этом в процессе начальной загрузки DOS будет создана корневая среда.

3) Откуда берётся информация, записываемая в среду?

Из AUTOEXEC.BAT, который является системным пакетным файлом, расположенным в корневом каталоге загрузочного устройства. Как правило, он используется в DOS для установки ключевых переменных окружения.

Выводы.

В ходе лабораторной работы был исследован интерфейс управляющей программы и загрузочных модулей, префикс сегмента программы (PSP) и среды, передаваемой программе.

ПРИЛОЖЕНИЕ А

Исходный код программы

lr2.asm

```
TESTPC SEGMENT
ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
ORG 100H
START: JMP BEGIN

; Данные
MEM_STR db 'Inaccessible memory     ', 0DH,0AH,'$'
ENV_ADRESS_STR db 'Environment segment adress     ', 0DH,0AH,'$'
EMPTY_TAIL_STR db 'Command line tail is empty', 0DH,0AH,'$'
TAIL_STR db 'Command line tail        ',0DH,0AH,'$'
ENV_CONTENT_STR db 'Symbolic content of the environment area ', 0DH,0AH,'$'
END_PRINT db 0DH,0AH,'$'
PATH_STR db 'Module 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 шестн. 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
;--------------------------------------------------

BYTE_TO_DEC PROC near
; перевод в 10с/с, SI - адрес поля младшей цифры
      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
;--------------------------------

; Код
BEGIN:

MAIN PROC near
; Сегментный адрес недоступной памяти
      mov AX,ES:[02h]
```

```
      mov DI,offset MEM_STR
      add DI,23
      call WRD_TO_HEX
      mov DX,offset MEM_STR
      mov AH,09h
      int 21h
      mov DX, offset END_PRINT
      mov AH,09h
      int 21h

; Сегментный адрес среды
      mov AX,ES:[2Ch]
      mov DI,offset ENV_ADRESS_STR
      add DI,30
      call WRD_TO_HEX
      mov DX,offset ENV_ADRESS_STR
      mov AH,09h
      int 21h
      mov DX, offset END_PRINT
      mov AH,09h
      int 21h

; Хвост командной строки
    xor CX,CX
      mov CL,DS:[80h]
      mov SI,offset TAIL_STR
      add SI,17
    cmp CL,0
           je EMPTY
      xor DI,DI
      xor AX,AX
      jmp READ_TAIL
EMPTY:
      mov DX, offset EMPTY_TAIL_STR
      jmp TAIL_ENDING
READ_TAIL:
      mov AL,DS:[81h+DI]
    inc DI
    mov [SI],Al
      inc SI
      loop READ_TAIL
      mov DX, offset TAIL_STR
      jmp TAIL_ENDING
TAIL_ENDING:
      mov AH,09h
      int 21h
      mov DX, offset END_PRINT
      mov AH,09h
      int 21h
```

```
; Содержимое области среды
      mov DX, offset ENV_CONTENT_STR
      mov AH,09h
      int 21h
      xor DI, DI
      mov DS, ES:[2Ch]
READ_CONTENT:
      cmp byte ptr [DI], 00
            je END_STR
      mov DL, [DI]
      mov AH, 02h
      int 21h
      jmp ENDING
END_STR:
      push DS
      mov CX, CS
      mov DS, CX
      mov DX, offset END_PRINT
      mov AH,09h
      int 21h
      pop DS
ENDING:
      inc DI
      cmp word ptr [DI], 0001
            jne READ_CONTENT

; Путь загружаемого модуля
      push DS
      mov CX, CS
      mov DS, CX
      mov DX, offset PATH_STR
      mov AH,09h
    int 21h
      pop DS
      add DI, 2
READ_PATH:
      cmp byte ptr [DI], 00
            je STOP
      mov DL, [DI]
      mov AH, 02h
      int 21h
      inc DI
      jmp READ_PATH
STOP:
      ret

MAIN ENDP
```

```
call MAIN

; Выход в DOS
xor AL,AL
mov AH,4Ch
int 21H

TESTPC ENDS
END START
```