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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Операционные системы»

Тема: Исследование интерфейсов командных модулей

Студент гр. 8383 _____ Дейнега В.Е.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург

2020

Цель работы.

Исследование интерфейса управляющей программы и загрузочных модулей. Исследование префикса сегмента программы (PSP) и среды, передаваемой программе.

Ход работы.

В ходе выполнения лабораторной работы был написан и отлажен программный модуль типа **.COM**, который выбирает и распечатывает на экран следующую информацию:

1. Сегментный адрес недоступной памяти 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.

Результат работы представлен на рис. 1 и 2. Исходный код программы содержится в Приложении А.

```
C:\>l2.com

Address of locked memory: 9FFF
Address of enviroment: 0188
Tail comand_line:  no command line
Enviroment: PATH=Z:\
 COMSPEC=Z:\COMMAND.COM
 BLASTER=A220 I7 D1 H5 T6

Path: C:\L2.COM
```

Рисунок 1 – Работа без входных данных

Рисунок 2 – Работа программы с входными данными

Контрольные вопросы

Ниже приведены ответы на контрольные вопросы:

Сегментный адрес недоступной памяти

1. На какую область памяти указывает адрес недоступной памяти? Адрес недоступной памяти указывает на служебную часть памяти, программа не может ее использовать.
2. Где расположен адрес по отношению к памяти, выделенной программе? После памяти, отведенной программе.
3. Можно ли в эту область памяти писать? Да, DOS не может запретить делать этого.

Среда, передаваемая программе

1. Что такое среда? Среда – последовательность строк вида параметр = значение, хранящие какую-либо информацию, например, данные о настройках системы.
2. Когда создается среда? Среда создается при запуске программы.
3. Откуда берется информация, записываемая в среду? Из файла AUTOEXEC.BAT.

Выводы.

В ходе выполнения работы был исследован интерфейс управляющей программы и загрузочных модулей, а так же интерфейс префикса сегмента программы (PSP) и среды, передаваемой программе.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

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


LEN_LOCKED  EQU 31
LEN_ENV EQU 28
STR_LOCKED db 13,10, "Address of locked memory:             $"
STR_ENV db 13,10, "Address of enviroment:             $"
STR_TAIL db 13,10, "Tail comand_line: $"
STR_EMPTY_TAIL db " no command line $"
STR_ENVIROMENT_AREA db 13,10, "Enviroment: $"
STR_ENTER db 13,10, " $"
STR_PATH db 13,10, "Path: $"


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


WRD_TO_HEX   PROC  near
            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
            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
;-----------------------------------------------


PRINT_LOCKED_MEM PROC near
   push ax
   push dx
   mov ax, ds:[02h]
   mov di, offset STR_LOCKED
   add di, LEN_LOCKED
   call WRD_TO_HEX
   mov dx, offset STR_LOCKED
   call WRITE_STR
   pop dx
   pop ax
   ret
PRINT_LOCKED_MEM ENDP


PRINT_ADDRESS_ENVIROMENT  PROC near
   push ax
   push dx
   mov ax, ds:[2Ch]
   mov di, offset STR_ENV
   add di, LEN_ENV
   call WRD_TO_HEX
   mov dx, offset STR_ENV
   call WRITE_STR
   pop dx
```

```
   pop ax
   ret
PRINT_ADDRESS_ENVIROMENT ENDP


PRINT_TAIL PROC near
   push ax
   push dx
   mov dx, offset STR_TAIL
   call WRITE_STR
   mov cx, 0
   mov cl, ds:[80h]
   cmp cl, 0
   je tail_empty
   mov di, 0
   xor dx,dx
 print_tail_cycle:
   mov dl, ds:[81h+di]
   mov ah,02H
   int 21h
   inc di
   loop print_tail_cycle
   jmp end_print
tail_empty:
   mov dx, offset STR_EMPTY_TAIL
   call WRITE_STR
end_print:
   pop dx
   pop ax
   ret
PRINT_TAIL ENDP


PRINT_PATH PROC near
   push dx
   push ax
   push ds
   mov dx, offset STR_ENVIROMENT_AREA
   call WRITE_STR
   mov di, 0
   mov es, ds:[2Ch]
cycle_env:
   cmp byte ptr es:[di], 00h
   je enter_
   mov dl, es:[di]
   mov ah, 02h
   int 21h
   inc di
   jmp cycle_env
enter_:
   inc di
   cmp word ptr es:[di], 0001h
   je path_
   mov dx, offset STR_ENTER
   call WRITE_STR
   jmp cycle_env
path_:
   inc di
   inc di
   mov DX, offset STR_PATH
   call WRITE_STR
cycle_p:
   cmp byte ptr es:[di], 00h
```

```
   je end_print_p
   mov dl, es:[di]
   mov ah, 02h
   int 21h
   inc di
   jmp cycle_p
end_print_p:
   pop dx
   pop ax
   pop ds
   ret
PRINT_PATH ENDP



MAIN:
   call PRINT_LOCKED_MEM
   call PRINT_ADDRESS_ENVIROMENT
   call PRINT_TAIL
   call PRINT_PATH
   xor al, al
   mov AH,4Ch
   int 21H
TESTPC ENDS
END START
```