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. 7383 _____ Иолшина В.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2019

Цель работы.

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

Таблица 1 - Описание процедур и структур данных.

Название процедуры	**Назначение**
MEMORY_	печатает адрес недоступной памяти, взятый из PSP, в шестнадцатеричном виде
SEGMENT_	печатает сегментный адрес среды, передаваемой программе, в шестнадцатеричном виде
SPACE	перенос содержимого на новую строку
PRINT	вызывает функцию печати строки
TAIL_	печатает хвост командной строки в символьном виде
PATH_	печатает путь загружаемого модуля
BYTE_TO_HEX	переводит байтовое число из регистра AL в шестнадцатеричную систему счисления, записывая получившееся в al и ah
TETR_TO_HEX	вспомогательная функция для работы функции BYTE_TO_HEX, которая переводит из двоичной в шестнадцатеричную систему счисления
WRD_TO_HEX	переводит число из регистра AX в строку в шестнадцатеричной системе счисления с, записывая получившееся в di

BYTE_TO_DEC	переводит байт из регистра AL в десятичную систему счисления и записывает получившееся число по адресу si, начиная с младшей цифры

Действия, выполняемые программой.

1. Печатать сегментного адреса первого байта недоступной памяти
2. Печатать сегментного адрес среды, передаваемого программе
3. Печатать хвоста командной строки
4. Печатать содержимого области среды в символьном виде
5. Печатать пути загружаемого модуля
6. Выход в DOS

Результат выполнения программы lr_2 представлен на рис. 1-2.

```
K:\>lr2
Address of unavailable memory:9FFF
Address of environment: 0188
Tail:
Content of the environment:
PATH=Z:\
COMSPEC=Z:\COMMAND.COM
BLASTER=A220 I7 D1 H5 T6
PATH of the loadable module:
K:\LR2.COM
```

Рисунок 1 – Результат выполнения программы lr2

```
K:\>lr2 Valeriya
Address of unavailable memory:9FFF
Address of environment: 0188
Tail: Valeriya
Content of the environment:
PATH=Z:\
COMSPEC=Z:\COMMAND.COM
BLASTER=A220 I7 D1 H5 T6
PATH of the loadable module:
K:\LR2.COM
```

Рисунок 2 – Результат выполнения программы lr2 Valeriya

Выводы.

В ходе данной лабораторной работы были исследованы интерфейс управляющей программы и загрузочного модуля. Был написан текст

исходного .COM файла, который выводит на экран сегментный адрес недоступной памяти, сегментный адрес среды, хвост командной строки, содержимое области и путь загружаемого модуля.

Ответы на контрольные вопросы.

Сегментный адрес недоступной памяти:

1) На какую область памяти указывает адрес недоступной памяти?

   На область, которая является доступной для загрузки программы.

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

   Этот адрес расположен сразу после области памяти, которая выделена программе, начиная с 9FFFh.

3) Можно ли в эту область памяти писать?

В эту область памяти можно писать, так как в MS-DOS отсутствует защита памяти.

Среда, передаваемая программе:

1) Что такое среда?

Среда представляет собой область памяти, в которой в виде символьных строк записаны значения переменных, называемых переменными среды, и некоторые данные об операционной системе, в виде последовательности символьных строк (имя = параметр), где каждая строка завершается байтом нулей.

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

Среда создается при загрузке DOS, при запуске приложения копируется в новую область памяти.

3) Откуда берется информация, записываемая в среду?

При работе с операционной системой MS – DOS, информация, которая заносится в среду, берётся из системного файла autoexec.bat.

ПРИЛОЖЕНИЕ А

LAB2_OS.ASM

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

ADDofMEM                  db           'Address of unavailable memory:    ',0dh,0ah,'$'
ADDofENV                  db           'Address of environment:     ',0dh,0ah,'$'
TAIL                      db           'Tail:','$'
CONTofENV                 db           'Content of the environment: ' , '$'
PATHofMOD                 db           'PATH of the loadable module: ' , '$'
ENDL                      db           0dh,0ah,'$'

SPACE        PROC   near
             lea          dx,ENDL
             call   PRINT
             ret
SPACE        ENDP

PRINT        PROC   near
             mov          ah,09h
             int          21h
             ret
PRINT        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
; segment address of unavailable memory
MEMORY_     PROC    near
            push    ax
            mov     ax,es:[2]
            lea         di,ADDofMEM
            add     di,33
            call    WRD_TO_HEX
            pop         ax
            ret
MEMORY_             ENDP
;  segment address of environment
SEGMENT_            PROC    near
            push    ax
            mov     ax,es:[2Ch]
            lea         di,ADDofENV
            add     di,27
            call    WRD_TO_HEX
            pop         ax
            ret
SEGMENT_            ENDP
;  finding tail
TAIL_       PROC    near
      push  ax
            push    cx
      xor   ax, ax
      mov   al, es:[80h]
      add   al, 81h
      mov   si, ax
      push  es:[si]
      mov   byte ptr es:[si+1], '$'
      push  ds
      mov   cx, es
      mov   ds, cx
      mov   dx, 81h
      call  PRINT
            pop     ds
      pop   es:[si]
      pop         cx
      pop         ax
            ret
TAIL_       ENDP
;  path of module
PATH_  PROC   near
            push    es
            push    ax
            push    bx
            push    cx
            mov         bx,1
            mov         es,es:[2ch]
            mov         si,0
      POINT:
            call    SPACE
            mov         ax,si
      POINT_:
            cmp     byte ptr es:[si], 0
            je          POINT1
```

```
            inc         si
            jmp   POINT_
      POINT1:
            push  es:[si]
            mov         byte ptr es:[si], '$'
            push  ds
            mov         cx,es
            mov         ds,cx
            mov         dx,ax
            call  PRINT
            pop         ds
            pop         es:[si]
            cmp         bx,0
            jz          POINT1_
            inc         si
            cmp   byte ptr es:[si], 01h
      jne   POINT
      lea         dx,PATHofMOD
      call  PRINT
      mov         bx,0
      add   si,2
      jmp   POINT
    POINT1_:
            pop         cx
            pop         bx
            pop         ax
            pop         es
            ret
PATH_ ENDP
;----------------------------
Write       PROC  near
            mov         ah,09h
            int         21h
            ret
Write       ENDP
BEGIN:
            call  MEMORY_
            call  SEGMENT_
            lea         dx,ADDofMEM
            call  PRINT
            lea         dx,ADDofENV
            call  PRINT
            lea   dx, TAIL
            call  PRINT
            call  TAIL_
            call  SPACE
            lea         dx,CONTofENV
            call  PRINT
            call  PATH_
            xor         al,al
            mov   ah, 04Ch
            int   21h
            ret
TESTPC ENDS
            END   START
```