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 СТРУКТУР ЗАГРУЗОЧНЫХ МОДУЛЕЙ.

Студент гр. 9381 Шахин Н.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.

Функции.

Название	Описание
TETR_TO_HEX	осуществляет перевод половины байта в символ.
BYTE_TO_HEX	осуществляет перевод байта, помещенного в al, в два символа в шестнадцатеричной системе счисления, помещая результат в ax.
WRD_TO_HEX	осуществляет перевод числового значения, помещенного в регистр AX, в символьную строку в шестнадцатеричной системе счисления, помещая результат в регистр di.
PRINT	Вызывает функцию 09h прерывания 21h

Ход работы.

Был написан и отлажен программный модуль типа .COM, который распечатывает и выводит информацию:

- Сегментный адрес недоступной памяти, взятый из PSP, в шестнадцатеричном виде.
- Сегментный адрес среды, передаваемой программе, в шестнадцатеричном виде.
- Хвост командной строки в символьном виде.
- Содержимое области среды в символьном виде.
- Путь загружаемого модуля.

Результаты работы программы:

Ответы на контрольные вопросы:

Сегментный адрес недоступной памяти

1. На какую область памяти указывает адрес недоступной памяти?

Адрес недоступной памяти указывает на значение сегментного адреса памяти, следующей за памятью, выделенной программе.

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

Сразу после памяти, выделенной программе. Расположен в сторону увеличения адресов.

3. Можно ли в эту область памяти писать?

Можно, так как контроля доступа к памяти нет.

Среда, передаваемая программе

1. Что такое среда?

Фактически, среда представляет собой текстовый массив, состоящий из строк вида:

"<переменная>=<значение>"

Где <переменная> и <значение> - текстовые величины и нулевой байт, обозначающий конец строки. Таким образом, среда – совокупность системных переменных среды, данные в которых могут быть необходимы программе во время ее работы.

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

Среда создаётся при загрузке операционной системы. Эта среда копируется в адресное пространство запущенной программы, и может быть изменена по требованию программы.

3. Откуда берется информация, записываемая в среду?

Из системного пакетного файла AUTOEXEC.BAT.

Вывод.

Был изучен интерфейс управляющей программы и загрузочных модулей. Был изучен PSP и среда, передаваемая программе.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Файл lb2.asm

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



in_memory   db    'Inaccessible memory adress:     $'
adress                db    'Environment adress:     $'
tail                  db    'Tail of command line:  $'
content     db    'Environment area contents:  $'
path                  db    'Loadable path:  $'
endl                  db  13, 10, '$'

;ПРОЦЕДУРЫ
;перевод десятичной цифры в код символа
;-------------------------------------------------------------------------
----
TETR_TO_HEX       PROC  near
            and         al, 0fh ;логическое умножение всех пар битов
            cmp         al, 09
            jbe         NEXT ;Переход если ниже или равно
            add         al, 07
NEXT: add         al, 30h
            ret
TETR_TO_HEX       ENDP


;перевод байта 16 с.с в символьный код
;байт в al переводится в 2 символа шестнадцетиричного числа в ax
;-------------------------------------------------------------------------
----
BYTE_TO_HEX       PROC near
            push  cx
            mov         ah, al
            call  TETR_TO_HEX
            xchg  al, ah ;обмен местами регистра/памяти и регистра
            mov         cl, 4
            shr         al, cl ;логический сдвиг вправо
            call  TETR_TO_HEX
            pop         cx
            ret
BYTE_TO_HEX       ENDP
;-------------------------------------------------------------------------
----
;Перевод в 16 сс 16-ти разрядного числа
;ax - число, di - адрес последнего символа
WRD_TO_HEX        PROC  near
            push  bx
            mov         bh, ah
            call  BYTE_TO_HEX
            mov         [di], ah
            dec         di ;вычитает 1 из операнда
            mov         [di], al
            dec         di
```

```
            mov         al, bh
            xor         ah, ah
            call  BYTE_TO_HEX
            mov         [di], ah
            dec         di
            mov         [di], al
            pop         bx
            ret
WRD_TO_HEX        ENDP

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

PRINT PROC NEAR
      push ax
      mov ah, 09h
      int 21h
      pop ax
      ret
PRINT ENDP

BEGIN:
            ;сегментные адреса недоступной памяти
            mov   ax, es:[0002h]
            mov   di, offset in_memory+31
            call  WRD_TO_HEX
            mov   dx, offset in_memory
            call PRINT
            mov         dx, offset endl
            call PRINT

            ;сегментный адрес среды
            mov   ax, es:[002Ch]
            mov   di, offset adress+23
            call  WRD_TO_HEX
            mov   dx, offset adress
            call PRINT
            mov         dx, offset endl
            call PRINT

            ;хвост командной строки в символьном виде
            mov   dx, offset tail
            call PRINT
            xor   cx, cx
            xor   bx, bx
            mov   cl, byte PTR es:[80h]
            mov   bx, 81h
cycle1:
            cmp         cx, 0h
            je          continue1
            mov   dl, byte PTR es:[bx]
            mov         ah, 02h; вывод символа на экран
            int         21h
            inc         bx
            dec         cx
            jmp         cycle1
continue1:
            mov         dx, offset endl
            call PRINT

            ;содержимое области среды в символьном виде
            push  es
            mov         dx, offset content
```

```
            call PRINT
            mov         dx, offset endl
            call PRINT
            mov         bx, es:[002Ch]
            mov         es, bx
            xor   bx, bx

continue2:
            mov   dl, byte PTR es:[bx]
            cmp   dl, 0h
            je    cycle2
            mov   ah, 02h
            int   21h
            inc   bx
            jmp   continue2
cycle2:
            mov   dx, offset endl
            call PRINT
            inc   bx
            mov   dl, byte PTR es:[bx]
            cmp   dl, 0h
            je          quit2
            jmp   continue2
quit2:

;Путь загружаемого модуля
            add   bx, 3
            mov         dx, offset endl
            mov   ah, 09h
            int   21h
            mov         dx, offset path
            call PRINT
cycle3:
            mov   dl, byte PTR es:[bx]
            cmp   dl, 0h
            je          quit3
            mov         ah, 02h
            int         21h
            inc         bx
            jmp   cycle3
quit3:
            mov   dx, offset endl
            call PRINT

            ;выход в dos
            xor         al, al
            mov   ah, 4ch
            int         21h
            ret
TESTPC      ENDS
            END   START
```