

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

ОТЧЕТ
по лабораторной работе №3
по дисциплине «Операционные системы»
ТЕМА: ИССЛЕДОВАНИЕ ОРГАНИЗАЦИИ УПРАВЛЕНИЯ ОСНОВНОЙ ПАМЯТЬЮ.

Студент гр. 9381

Шахин Н.С

Преподаватель

Ефремов М. А.

Санкт-Петербург

2021

Цель работы.

Для исследования организации управления памятью. Необходимо ориентироваться на тип основной памяти, реализованный в компьютере и способ организации, принятый в ОС. В лабораторной работе рассматривается не страничная память и способ управления динамическими разделами. Для реализации управления памятью в этом случае строится список занятых и свободных участков памяти. Функции ядра, обеспечивающие управление основной памятью, просматривают и преобразуют этот список.

В лабораторной работе исследуются структуры данных и работа функций управления памятью ядра операционной системы.

Ход работы.

1. Написан и отлажен программный модуль типа .COM, который выбирает и распечатывает следующую информацию:

Количество доступной памяти

Размер расширенной памяти

Выводит цепочку битов управления памятью

```
F:\>lab3.com
Amount of available memory:    648912 b
Size of extended memory:      15360 Kb
List of memory control blocks:
MCB type: 4Dh  PSP address: 0008h      Size:    16 b
MCB type: 4Dh  PSP address: 0000h      Size:     64 b
MCB type: 4Dh  PSP address: 0040h      Size:   256 b
MCB type: 4Dh  PSP address: 0192h      Size:   144 b
MCB type: 5Ah  PSP address: 0192h      Size: 648912 b      LAB3
```

2. Программа изменена таким образом, что память, которая не использована освобождается с помощью функции 4Ah прерывания 21H.

```

F:\>lab3_2.com
Amount of available memory:      648912 b
Size of extended memory:        15360 Kb
List of memory control blocks:
MCB type: 4Dh   PSP address: 0008h   Size:      16 b
MCB type: 4Dh   PSP address: 0000h   Size:       64 b
MCB type: 4Dh   PSP address: 0040h   Size:      256 b
MCB type: 4Dh   PSP address: 0192h   Size:      144 b
MCB type: 4Dh   PSP address: 0192h   Size:      816 b      LAB3_2
MCB type: 5Ah   PSP address: 0000h   Size:    648080 b

```

3. Программа изменена. После освобождения памяти, программа запрашивает 64Кб памяти функцией 48Н прерывания 21Н.

```

F:\>lab3_3.com
Amount of available memory:      648912 b
Size of extended memory:        15360 Kb
List of memory control blocks:
MCB type: 4Dh   PSP address: 0008h   Size:      16 b
MCB type: 4Dh   PSP address: 0000h   Size:       64 b
MCB type: 4Dh   PSP address: 0040h   Size:      256 b
MCB type: 4Dh   PSP address: 0192h   Size:      144 b
MCB type: 4Dh   PSP address: 0192h   Size:      864 b      LAB3_3
MCB type: 4Dh   PSP address: 0192h   Size:    65536 b      LAB3_3
MCB type: 5Ah   PSP address: 0000h   Size:    582480 b

```

4. Изменен первоначальный вариант программы. Запрашивается 64Кб памяти функцией 48Н прерывания 21Н до освобождения памяти.

```

F:\>lab3_4.com
Amount of available memory:      648912 b
ERROR! Memory can not be allocated!
Size of extended memory:        15360 Kb
List of memory control blocks:
MCB type: 4Dh   PSP address: 0008h   Size:      16 b
MCB type: 4Dh   PSP address: 0000h   Size:       64 b
MCB type: 4Dh   PSP address: 0040h   Size:      256 b
MCB type: 4Dh   PSP address: 0192h   Size:      144 b
MCB type: 4Dh   PSP address: 0192h   Size:      864 b      LAB3_4
MCB type: 5Ah   PSP address: 0000h   Size:    648032 b      LAB3_3

```

Описание функций.

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 ah.

WRD_TO_HEX	осуществляет перевод числового значения, помещенного в регистр AX, в символьную строку в шестнадцатеричной системе счисления, помещая результат в регистр di.
BYTE_TO_DEC	осуществляет перевод байта, помещенного в AL, в два символа в десятичной системе счисления, помещая результат в SI.
WRD_TO_DEC	осуществляет перевод слова, помещенного в AX, в последовательность символов в десятичной системе счисления, помещая результат в SI. (по аналогии с BYTE_TO_DEC)
PRINT	осуществляет вывод строки на экран.
PRINT_SYMB	осуществляет вывод символа на экран (используя функцию DOS 02h, прерывания 21).

Ответы на контрольные вопросы.

1. Что означает “доступный объем памяти”?

Доступный объем памяти – область основной памяти, выделенная программе.

2. Где МСВ блок вашей программы в списке?

В первой версии программы он расположен в конце списка, т.к. программа была последней загружена в память и обладает всем объемом свободной ранее памяти.

Во второй версии, МСВ блок - предпоследний. Последним является блок, освобожденной программой памяти.

В третьей версии, блок так же пятый в списке, но после него располагаются блок, памяти в 64Кб, выделенный по запросу программы и после – блок свободной памяти.

В четвертой ситуация как во второй версии, так как запрос выделения памяти завершился неудачей.

3.Какой размер памяти занимает программа в каждом случае?

В первой программе, она занимает всю выделенную ей память: 648912б.

Во второй – только объем, занимаемый самой программой: 800 б.

В третьей – объем, занимаемый самой программой и 64 Кб, выделенные ей по требованию: 65536 б.

В четвертой – только объем, занимаемый самой программой, т. к. выделить 64кБ было невозможно: 864 б.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПОГРАММЫ

Файл lab3.asm

```
PCinfo segment
    assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
    org 100h

start:
    jmp begin

;data
av_mem db 'Amount of available memory:          b$'
ex_mem db 'Size of extended memory:            Kb$'
mcb db 'List of memory control blocks:$'
typeMCB db 'MCB type: 00h$'
adrPSP db 'PSP address: 0000h$'
size_s db 'Size:          b$'
    endl db 13, 10, '$'
    tab db 9, '$'

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

;Байт в al переводится в два символа 16-ричного числа в ah
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 ;В al старшая цифра, в ah младшая
    pop cx
    ret
byte_to_hex endp

;Перевод в 16 ss 16-ти разрядного числа
;ah - число, di - адрес последнего символа
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

;Перевод в 10 сс, si - адрес поля младшей цифры
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

wrd_to_dec proc near
        push    cx
        push    dx
        mov     cx, 10
wloop_bd:
        div     cx
        or      dl, 30h
        mov     [si], dl
        dec     si
        xor     dx, dx
        cmp     ax, 10
        jae     wloop_bd
        cmp     al, 00h
        je      wend_l
        or      al, 30h
        mov     [si], al
wend_l:
        pop     dx
        pop     cx
        ret
        wrd_to_dec endp

;Вывод строки
print proc near
        push    ax
        push    dx
        mov     ah, 09h
        int     21h
        pop     dx
        pop     ax
        ret
        print endp

;Вывод символа
print_symb proc near
        push    ax

```

```

push    dx
mov     ah, 02h
int     21h
pop     dx
pop     ax
ret
    print_symb endp

begin:

;количество доступной памяти
mov     ah, 4Ah
mov     bx, 0ffffh
int     21h

xor     dx, dx
mov     ax, bx
mov     cx, 10h
mul     cx

mov     si, offset av_mem+37
call    wrd_to_dec

mov     dx, offset av_mem
call    print
mov     dx, offset endl
call    print

;размер расширенной памяти
mov     al, 30h
out     70h, al
in      al, 71h
mov     bl, al ;младший байт
mov     al, 31h
out     70h, al
in      al, 71h ;старший байт
mov     ah, al
mov     al, bl

mov     si, offset ex_mem+34
xor     dx, dx
call    wrd_to_dec

mov     dx, offset ex_mem
call    print
mov     dx, offset endl
call    print

;цепочка блоков управления памятью
    mov     dx, offset mcb
    call    print
    mov     dx, offset endl
    call    print

    mov     ah, 52h
    int     21h
    mov     ax, es:[bx-2]
    mov     es, ax

;тип MCB
tag1:

```



```

mov     al, es:[0000h]
call    byte_to_hex
mov     di, offset typeMCB+10
mov     [di], ax

mov     dx, offset typeMCB
call    print
mov     dx, offset tab
call    print

; сегментный адрес PSP владельца участка памяти
mov     ax, es:[0001h]
mov     di, offset adrPSP+15
call    wrd_to_hex

mov     dx, offset adrPSP
call    print
mov     dx, offset tab
call    print

; размер участка в параграфах
mov     ax, es:[0003h]
mov     cx, 10h
mul     cx

mov     si, offset size_s+13
call    wrd_to_dec
mov     dx, offset size_s
call    print
mov     dx, offset tab
call    print

; последние 8 байт
push    ds
push    es
pop     ds

mov     dx, 08h
mov     di, dx
mov     cx, 8
tag2:
cmp     cx, 0
je      tag3
mov     dl, byte PTR [di]
call    print_symb
dec     cx
inc     di
jmp     tag2
tag3:
pop     ds
mov     dx, offset endl
call    print

; проверка, последний блок или нет
cmp     byte ptr es:[0000h], 5ah
je      quit

; адрес следующего блока
mov     ax, es
add     ax, es:[0003h]
inc     ax
mov     es, ax
jmp     tag1

```

```
quit:

        xor     ax, ax
        mov     ah, 4ch
        int     21h
PCinfo ENDS
        END     START
```

Файл lab3_2.asm

```
PCinfo segment
        assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
        org     100h

start:
        jmp     begin

;data
av_mem      db 'Amount of available memory:          b$'
ex_mem      db 'Size of extended memory:             Kb$'
mcb         db 'List of memory control blocks:$'
typeMCB     db 'MCB type: 00h$'
adrPSP      db 'PSP address: 0000h$'
size_s      db 'Size:                                b$'
        endl   db 13, 10, '$'
        tab    db 9, '$'

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

;Байт в al переводится в два символа 16-ричного числа в ax
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 ;В al старшая цифра, в ah младшая
        pop     cx
        ret
byte_to_hex endp

;Перевод в 16-ти разрядного числа
;ax - число, di - адрес последнего символа
word_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

;Перевод в 10 сс, si - адрес поля младшей цифры
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

wrd_to_dec proc near
    push    cx
    push    dx
    mov     cx, 10
wloop_bd:
    div     cx
    or      dl, 30h
    mov     [si], dl
    dec     si
    xor     dx, dx
    cmp     ax, 10
    jae     wloop_bd
    cmp     al, 00h
    je      wend_l
    or      al, 30h
    mov     [si], al
wend_l:
    pop     dx
    pop     cx
    ret
wrd_to_dec endp

;Вывод строки
print proc near
    push    ax
    push    dx
    mov     ah, 09h
    int     21h
    pop     dx
    pop     ax
    ret
print endp

```

```
;ВЫВОД СИМВОЛА
print_symb proc near
    push    ax
    push    dx
    mov     ah, 02h
    int     21h
    pop     dx
    pop     ax
    ret
    print_symb endp
```

```
begin:
```

```
;количество доступной памяти
mov     ah, 4Ah
mov     bx, 0ffffh
int     21h
```

```
xor     dx, dx
mov     ax, bx
mov     cx, 10h
mul     cx
```

```
mov     si, offset av_mem+37
call    wrd_to_dec
```

```
mov     dx, offset av_mem
call    print
mov     dx, offset endl
call    print
```

```
;освобождение памяти
mov     ax, offset SegEnd
mov     bx, 10h
xor     dx, dx
div     bx
inc     ax
mov     bx, ax
mov     al, 0
mov     ah, 4Ah
int     21h
```

```
;размер расширенной памяти
mov     al, 30h
out     70h, al
in      al, 71h
mov     bl, al ;младший байт
mov     al, 31h
out     70h, al
in      al, 71h ;старший байт
mov     ah, al
mov     al, bl
```

```
mov     si, offset ex_mem+34
xor     dx, dx
call    wrd_to_dec
```

```
mov     dx, offset ex_mem
call    print
mov     dx, offset endl
```

```

call print

;цепочка блоков управления памятью
mov     dx, offset mcb
call    print
mov     dx, offset endl
call    print

mov     ah, 52h
int     21h
mov     ax, es:[bx-2]
mov     es, ax

;тип MCB
tag1:
mov     al, es:[0000h]
call    byte_to_hex
mov     di, offset typeMCB+10
mov     [di], ax

mov     dx, offset typeMCB
call    print
mov     dx, offset tab
call    print

;сегментный адрес PSP владельца участка памяти
mov     ax, es:[0001h]
mov     di, offset adrPSP+15
call    wrd_to_hex

mov     dx, offset adrPSP
call    print
mov     dx, offset tab
call    print

;размер участка в параграфах
mov     ax, es:[0003h]
mov     cx, 10h
mul     cx

mov     si, offset size_s+13
call    wrd_to_dec
mov     dx, offset size_s
call    print
mov     dx, offset tab
call    print

;последние 8 байт
push    ds
push    es
pop     ds

mov     dx, 08h
mov     di, dx
mov     cx, 8
tag2:
cmp     cx, 0
je      tag3
mov     dl, byte PTR [di]
call    print_symb
dec     cx
inc     di
jmp     tag2

```

```

tag3:
    pop    ds
    mov     dx, offset endl
    call    print

    ;проверка, последний блок или нет
    cmp     byte ptr es:[0000h], 5ah
    je      quit

    ;адрес следующего блока
    mov     ax, es
    add     ax, es:[0003h]
    inc     ax
    mov     es, ax
    jmp     tag1

quit:

    xor     ax, ax
    mov     ah, 4ch
    int     21h
SegEnd:
PCinfo ENDS
        END     START

```

Файл lab3_3.asm

```

PCinfo segment
    assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
    org    100h

start:
    jmp     begin

;data
av_mem      db 'Amount of available memory:           b$'
ex_mem      db 'Size of extended memory:              Kb$'
mcb         db 'List of memory control blocks:$'
typeMCB     db 'MCB type: 00h$'
adrPSP      db 'PSP adress: 0000h$'
size_s      db 'Size:                                b$'
    endl    db 13, 10, '$'
    tab     db 9, '$'
    error   db 'ERROR! Memory can not be allocated!$'

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

;Байт в al переводится в два символа 16-ричного числа в ax
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 ;В al старшая цифра, в ah младшая

```

```

        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
    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

;Перевод в 10 сс, si - адрес поля младшей цифры
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

wrd_to_dec proc near
    push        cx
    push        dx
    mov         cx, 10
wloop_bd:
    div         cx
    or          dl, 30h
    mov         [si], dl
    dec         si
    xor         dx, dx
    cmp         ax, 10
    jae         wloop_bd
    cmp         al, 00h
    je          wend_l
    or          al, 30h

```

```

        mov     [si], al
wend_l:
        pop     dx
        pop     cx
        ret
        wrd_to_dec endp

;Вывод строки
print proc near
        push    ax
        push    dx
        mov     ah, 09h
        int     21h
        pop     dx
        pop     ax
        ret
print endp

;Вывод символа
print_symb proc near
        push    ax
        push    dx
        mov     ah, 02h
        int     21h
        pop     dx
        pop     ax
        ret
        print_symb endp

begin:

;количество доступной памяти
        mov     ah, 4Ah
        mov     bx, 0ffffh
        int     21h

        xor     dx, dx
        mov     ax, bx
        mov     cx, 10h
        mul     cx

        mov     si, offset av_mem+37
        call    wrd_to_dec

        mov     dx, offset av_mem
        call    print
        mov     dx, offset endl
        call    print

;освобождение памяти
        mov     ax, offset SegEnd
        mov     bx, 10h
        xor     dx, dx
        div     bx
        inc     ax
        mov     bx, ax
        mov     al, 0
        mov     ah, 4Ah
        int     21h

;запрос памяти

```



```

xor        ax, ax
mov        ah, 48h
mov        bx, 1000h
int        21h
jnc        mem_ok
mov        dx, offset error
call print
mov        dx, offset endl
call print
mem_ok:

;размер расширенной памяти
mov        al, 30h
out        70h, al
in         al, 71h
mov        bl, al ;младший байт
mov        al, 31h
out        70h, al
in         al, 71h ;старший байт
mov        ah, al
mov        al, bl

mov        si, offset ex_mem+34
xor        dx, dx
call wrd_to_dec

mov        dx, offset ex_mem
call print
mov        dx, offset endl
call print

;цепочка блоков управления памятью
mov        dx, offset mcb
call print
mov        dx, offset endl
call print

mov        ah, 52h
int        21h
mov        ax, es:[bx-2]
mov        es, ax

;тип MCB
tag1:
mov        al, es:[0000h]
call byte_to_hex
mov        di, offset typeMCB+10
mov        [di], ax

mov        dx, offset typeMCB
call print
mov        dx, offset tab
call print

;сегментный адрес PSP владельца участка памяти
mov        ax, es:[0001h]
mov        di, offset adrPSP+15
call wrd_to_hex

mov        dx, offset adrPSP
call print
mov        dx, offset tab
call print

```

```

;размер участка в параграфах
mov     ax, es:[0003h]
mov     cx, 10h
mul     cx

mov     si, offset size_s+13
call    wrd_to_dec
mov     dx, offset size_s
call    print
mov     dx, offset tab
call    print

;последние 8 байт
push    ds
push    es
pop     ds

mov     dx, 08h
mov     di, dx
mov     cx, 8
tag2:
cmp     cx, 0
je      tag3
mov     dl, byte PTR [di]
call    print_symb
dec     cx
inc     di
jmp     tag2
tag3:
pop     ds
mov     dx, offset endl
call    print

;проверка, последний блок или нет
cmp     byte ptr es:[0000h], 5ah
je      quit

;адрес следующего блока
mov     ax, es
add     ax, es:[0003h]
inc     ax
mov     es, ax
jmp     tag1

quit:

xor     ax, ax
mov     ah, 4ch
int     21h
SegEnd:
PCinfo ENDS
        END      START

```

Файл lab3_4.asm

```

PCinfo segment
        assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
org     100h

start:
jmp     begin

```

```

;data
av_mem      db 'Amount of available memory:          b$'
ex_mem      db 'Size of extended memory:            Kb$'
mcb         db 'List of memory control blocks:$'
typeMCB     db 'MCB type: 00h$'
adrPSP      db 'PSP address: 0000h$'
size_s      db 'Size:          b$'
        endl      db 13, 10, '$'
        tab       db 9, '$'
        error     db 'ERROR! Memory can not be allocated!$'

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

;Байт в al переводится в два символа 16-ричного числа в ax
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 ;В al старшая цифра, в ah младшая
        pop       cx
        ret
byte_to_hex endp

;Перевод в 16 cc 16-ти разрядного числа
;ax - число, di - адрес последнего символа
wrд_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
wrд_to_hex endp

;Перевод в 10 cc, si - адрес поля младшей цифры
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

```

```

wrd_to_dec proc near
        push    cx
        push    dx
        mov     cx, 10
wloop_bd:
        div     cx
        or      dl, 30h
        mov     [si], dl
        dec     si
        xor     dx, dx
        cmp     ax, 10
        jae     wloop_bd
        cmp     al, 00h
        je      wend_l
        or      al, 30h
        mov     [si], al
wend_l:
        pop     dx
        pop     cx
        ret
wrd_to_dec endp

```

```

;ВЫВОД строки
print proc near
        push    ax
        push    dx
        mov     ah, 09h
        int     21h
        pop     dx
        pop     ax
        ret
print endp

```

```

;ВЫВОД СИМВОЛА
print_symb proc near
        push    ax
        push    dx
        mov     ah, 02h
        int     21h
        pop     dx
        pop     ax
        ret
print_symb endp

```

```

begin:

```

```

;количество доступной памяти
mov     ah, 4Ah
mov     bx, 0ffffh
int     21h

xor     dx, dx
mov     ax, bx
mov     cx, 10h
mul     cx

mov     si, offset av_mem+37
call    wrd_to_dec

mov     dx, offset av_mem
call    print
mov     dx, offset endl
call    print

;запрос памяти
xor     ax, ax
mov     ah, 48h
mov     bx, 1000h
int     21h
jnc     mem_ok
mov     dx, offset error
call    print
mov     dx, offset endl
call    print
mem_ok:

;освобождение памяти
mov     ax, offset SegEnd
mov     bx, 10h
xor     dx, dx
div     bx
inc     ax
mov     bx, ax
mov     al, 0
mov     ah, 4Ah
int     21h

;размер расширенной памяти
mov     al, 30h
out     70h, al
in      al, 71h
mov     bl, al ;младший байт
mov     al, 31h
out     70h, al
in      al, 71h ;старший байт
mov     ah, al
mov     al, bl

mov     si, offset ex_mem+34
xor     dx, dx
call    wrd_to_dec

mov     dx, offset ex_mem
call    print
mov     dx, offset endl
call    print

;цепочка блоков управления памятью

```

```

        mov     dx, offset mcb
        call    print
mov     dx, offset endl
call    print

        mov     ah, 52h
        int     21h
        mov     ax, es:[bx-2]
        mov     es, ax

        ;тип MCB
tag1:
        mov     al, es:[0000h]
        call    byte_to_hex
        mov     di, offset typeMCB+10
        mov     [di], ax

        mov     dx, offset typeMCB
        call    print
        mov     dx, offset tab
        call    print

        ;сегментный адрес PSP владельца участка памяти
        mov     ax, es:[0001h]
        mov     di, offset adrPSP+15
        call    wrd_to_hex

        mov     dx, offset adrPSP
        call    print
        mov     dx, offset tab
        call    print

        ;размер участка в параграфах
        mov     ax, es:[0003h]
        mov     cx, 10h
        mul     cx

        mov     si, offset size_s+13
        call    wrd_to_dec
        mov     dx, offset size_s
        call    print
        mov     dx, offset tab
        call    print

        ;последние 8 байт
        push    ds
        push    es
        pop     ds

        mov     dx, 08h
        mov     di, dx
        mov     cx, 8

tag2:
        cmp     cx, 0
        je      tag3
        mov     dl, byte PTR [di]
        call    print_symb
        dec     cx
        inc     di
        jmp     tag2
tag3:
        pop     ds
        mov     dx, offset endl

```

```

call    print

;проверка, последний блок или нет
cmp     byte ptr es:[0000h], 5ah
je      quit

;адрес следующего блока
mov     ax, es
add     ax, es:[0003h]
inc     ax
mov     es, ax
jmp     tag1

quit:

xor     ax, ax
mov     ah, 4ch
int     21h
SegEnd:
PCinfo ENDS
        END      START

```