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 .СОМ, 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 Ъ
Size of extended memory:
                               15360 КЪ
List of memory control blocks:
MCB type: 4Dh
                PSP adress: 0008h
                                         Size:
                                                      16 b
MCB type: 4Dh
                PSP adress: 0000h
                                         Size:
                                                     64 h
MCB type: 4Dh
                PSP adress: 0040h
                                         Size:
                                                     256 Ъ
MCB type: 4Dh
                PSP adress: 0192h
                                         Size:
                                                     144 Ь
MCB type: 5Ah
                PSP adress: 0192h
                                                 648912 Ь
                                                                  LAB3
                                         Size:
```

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 Ь
Size of extended memory:
                                  15360 КЪ
List of memory control blocks:
MCB type: 4Dh
MCB type: 4Dh
MCB type: 4Dh
MCB type: 4Dh
                 PSP adress: 0008h
                                             Size:
                                                           16 b
                PSP adress: 0000h
                                                           64 b
                                             Size:
                  PSP adress: 0040h
                                                          256 Ъ
                                             Size:
                  PSP adress: 0192h
                                             Size:
                                                          144 Ь
 CB type: 4Dh
                  PSP adress: 0192h
                                             Size:
                                                          816 Ъ
                                                                         LAB3_2
 KB type: 5Ah
                  PSP adress: 0000h
                                             Size:
                                                       648080 Ъ
```

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

```
F:\>lab3_3.com
Amount of available memory:
                                      648912 Ъ
Size of extended memory:
                                    15360 КЪ
List of memory control blocks:
MCB type: 4Dh
                   PSP adress: 0008h
                                                Size:
                                                              16 b
MCB type: 4Dh
                   PSP adress: 0000h
                                                Size:
                                                              64 b
                   PSP adress: 0040h
                                                             256 Ъ
                                                Size:
                   PSP adress: 0192h
                                                             144 Ь
                                                Size:
                   PSP adress: 0192h
                                                Size:
                                                             864 Ъ
                                                                             LAB3 3
                   PSP adress: 0192h
                                                Size:
                                                           65536 Ъ
                                                                             LAB3 3
 1CB type: 5Ah
                   PSP adress: 0000h
                                                Size:
                                                         582480 Ъ
```

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

```
F:\>lab3_4.com
Amount of available memory:
                                648912 Ъ
ERROR! Memory can not be allocated!
                              15360 КЪ
Size of extended memory:
List of memory control blocks:
MCB type: 4Dh
                PSP adress: 0008h
                                        Size:
                                                     16 b
MCB type: 4Dh
                PSP adress: 0000h
                                        Size:
                                                     64 b
MCB type: 4Dh
                PSP adress: 0040h
                                        Size:
                                                    256 Ъ
MCB type: 4Dh
                PSP adress: 0192h
                                        Size:
                                                    144 Ь
MCB type: 4Dh
                PSP adress: 0192h
                                        Size:
                                                    864 Ъ
                                                                 LAB3_4
1CB type: 5Ah
                PSP adress: 0000h
                                                 648032 Ъ
                                                                 LAB3_3
                                        Size:
```

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 ах.

WRD_TO_HEX	осуществляет перевод числового значения, помещенного
	в регистр АХ, 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 АХ, 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 (номер 5), т.к. программа была последней загружена в память и обладает всем объемом свободной ранее памяти.

Во второй версии, МСВ блок – предпоследний (номер 5). 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 версии, МСВ блок – предпоследний (номер 5).

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
          db 'Amount of available memory:
db 'Size of extended memory:
 av mem
                                                       b$'
 ex mem
                                                     Kb$'
 mcb db 'List of memory control blocks:$'
 typeMCB db 'MCB type: 00h$'
 adrPSP db 'PSP adress: 0000h$'
 size s
           db 'Size:
   endl db 13, 10, '$'
           db 9,'$'
    tab
tetr to hex proc near
    and al, OFh
           al, 09
    cmp
           next
    jbe
           al, 07
    add
next:
           al, 30h
   add
   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
           al, ah
   xchq
           cl, 4
   mov
           al, cl
           tetr to hex ;В al старшая цифра, в ah младшая
   call
   pop
   ret
   byte to hex endp
;Перевод в 16 сс 16-ти разрядного числа
; ах - число, di - адрес последнего символа
wrd to hex proc near
    push
           bx
             bh, ah
    mov
    call
            byte to hex
            [di], ah
    mov
            di
    dec
            [di], al
    mov
    dec
            di
            al, bh
    mov
           byte_to_hex
    call
    mov
            [di], ah
    dec
            di
            [di], al
    mov
            bx
    pop
    ret
```

```
wrd to hex endp
;Перевод в 10 cc, si - адрес поля младшей цифры
byte to dec proc near
    push
             CX
    push
              dx
             ah, ah
    xor
              dx, dx
    xor
              cx, 10
    mov
loop_bd:
    div
              CX
              dl, 30h
    or
    mov
              [si], dl
    dec
              si
    xor
              dx, dx
              ax, 10
    cmp
              loop bd
    jae
              al, \overline{00}
    cmp
              end 1
    jе
    or
              al, 30h
    mov
              [si], al
end 1:
    pop
              dx
    pop
              CX
    ret
   byte_to_dec endp
wrd_to_dec proc near
    push
            CX
              dx
    push
             cx, 10
    mov
wloop_bd:
    div
              CX
              dl, 30h
    or
             [si], dl
    mov
    dec
             si
 xor dx, dx
             ax, 10
    cmp
              wloop bd
    jae
              al, 00h
    cmp
              wend 1
    jе
              al, \overline{30h}
    or
              [si], al
    {\tt mov}
wend_1:
    pop
              dx
    pop
              CX
    ret
   wrd_to_dec endp
;вывод строки
print proc near
    push
           ax
    push
             dx
    mov
              ah, 09h
    int
              21h
              dx
    pop
    pop
              ax
    ret
   print endp
;вывод символа
print_symb proc near
 push ax
```

```
push dx
 mov
           ah, 02h
           21h
 int
           dx
 pop
            ax
 pop
 ret
 print_symb endp
begin:
; количество доступной памяти
mov ah, 4Ah
     bx, Offffh
 mov
     21h
 int
 xor dx, dx
 mov ax, bx
 mov cx, 10h
 mul cx
 mov si, offset av mem+37
 call wrd_to_dec
      dx, offset av_mem
 mov
 call print
 mov dx, offset endl
 call print
;размер расширенной памяти
mov al, 30h
      70h, al
 out
      al, 71h
 in
 mov bl, al ;младший байт
 mov al, 31h
 out 70h, al
      al, 71h ;старший байт
 in
 mov ah, al
 mov al, bl
 mov si, offset ex mem+34
      dx, dx
 xor
 call wrd_to_dec
     dx, offset ex mem
 mov
 call print
      dx, offset endl
call print
;цепочка блоков управления памятью
   mov dx, offset mcb
         print
   call
 mov
            dx, offset endl
 call print
           ah, 52h
   mov
            21h
   int
            ax, es:[bx-2]
   mov
   mov
            es, ax
   ;тип МСВ
tag1:
```

```
mov al, es:[0000h]
   call byte_to_hex
          di, offset typeMCB+10
   mov
           [di], ax
   mov
   mov
          dx, offset typeMCB
          print
   call
   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
          dx, offset adrPSP
   mov
   call
          print
           dx, offset tab
   mov
   call
          print
   ;размер участка в параграфах
   mov ax, es:[0003h]
   mov
          cx, 10h
   mul
          CX
          si, offset size_s+13
   call
          wrd to dec
          dx, offset size s
   mov
          print
   call
          dx, offset tab
   mov
   call
          print
   ;последние 8 байт
   push ds
   push
          es
          ds
   pop
          dx, 08h
   mov
          di, dx
   mov
           cx, 8
   mov
tag2:
           cx,0
 cmp
 jе
           tag3
           dl, byte PTR [di]
   mov
          print_symb
   call
   dec
           CX
   inc
           di
   jmp
           tag2
tag3:
 pop ds
          dx, offset endl
mov
   call
          print
   ;проверка, последний блок или нет
   cmp byte ptr es:[0000h], 5ah
           quit
   jе
   ;адрес следующего блока
   mov ax, es
   add
          ax, es:[0003h]
   inc
          ax
   mov
jmp
          es, ax
          tag1
```

```
quit:
    xor ax, ax
           ah, 4ch
   mov
             21h
    int
PCinfo ENDS
       END
             START
Файл lab3 2.asm
PCinfo segment
      assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
 org
start:
 jmp
            begin
 ;data
          db 'Amount of available memory:
db 'Size of extended memory:
                                                        b$'
 av mem
                                                    Kb$'
 ex mem
 mcb db 'List of memory control blocks:$'
 typeMCB db 'MCB type: 00h$'
            db 'PSP adress: 0000h$'
 adrPSP
            db 'Size:
 size s
             db 13, 10, '$'
    endl
                9,'$'
             db
    tab
tetr to hex proc near
        al, OFh
    and
            al, 09
    cmp
    jbe
            next
            al, 07
    add
next:
            al, 30h
    add
   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
    ret
   byte to hex endp
;Перевод в 16 сс 16-ти разрядного числа
; ах - число, di - адрес последнего символа
wrd to hex proc near
    push
           bx
            bh, ah
    mov
    call
           byte to hex
            [di], ah
    mov
    dec
            di
    mov
            [di], al
    dec
            di
            al, bh
    call
           byte to hex
           [di], ah
    mov
```

```
dec
             di
    mov
             [di], al
    pop
             bх
    ret
   wrd to hex endp
;Перевод в 10 cc, si - адрес поля младшей цифры
byte_to_dec proc near
    push
             CX
    push
             dx
             ah, ah
    xor
             dx, dx
    xor
             cx, 10
    mov
loop bd:
    div
             CX
             dl, 30h
    or
             [si], dl
    mov
    dec
             si
             dx, dx
    xor
    cmp
             ax, 10
    jae
             loop bd
    cmp
             al, \overline{0}0h
    jе
             end 1
    or
             al, 30h
             [si], al
    mov
end 1:
             dx
    pop
             CX
    pop
    ret
   byte to dec endp
wrd to dec proc near
    push
            CX
             dx
    push
             cx, 10
    mov
wloop_bd:
             CX
    div
             dl, 30h
    or
             [si], dl
    mov
    dec
             si
 xor dx, dx
             ax, 10
    cmp
             wloop bd
    jae
             al, 00h
    cmp
             wend 1
    jе
             al, \overline{30}h
    or
             [si], al
    mov
wend 1:
    pop
             dx
    рор
             CX
    ret
   wrd_to_dec endp
;вывод строки
print proc near
    push ax
           dx
    push
             ah, 09h
    mov
    int
             21h
    pop
             dx
             ax
    pop
    ret
   print endp
```

```
;вывод символа
print_symb proc near push ax push dx
           ah, 02h
mov
            21h
 int
            dx
 pop
 pop
           ax
 ret
 print symb endp
begin:
; количество доступной памяти
 mov ah, 4Ah
     bx, Offffh
 mov
 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
       dx, offset endl
 mov
 call print
;освобождение памяти
   mov ax, offset SegEnd
           bx, 10h
   mov
            dx, dx
   xor
           bx
   div
   inc
            ax
           bx, ax
   mov
            al, 0
   mov
            ah, 4Ah
   mov
            21h
   int
;размер расширенной памяти
 mov al, 30h
 out
            70h, al
            al, 71h
bl, al ;младший байт
al, 31h
 in
 mov
            аl, 71h ;старший байт
ah, al
 out
 in
 mov
            al, bl
 mov
            si, offset ex_mem+34
 mov
 xor dx, dx
 call wrd to dec
 mov
           dx, offset ex_mem
 call print
```

dx, offset endl

mov

```
call print
;цепочка блоков управления памятью
  mov dx, offset mcb
call print mov dx, offset endl
 call print
           ah, 52h
   mov
           21h
   int
   mov
          ax, es:[bx-2]
   mov
          es, ax
   ;тип МСВ
tag1:
 mov al, es:[0000h]
   call byte_to_hex
           di, offset typeMCB+10
           [di], ax
          dx, offset typeMCB
   call
          print
   mov
           dx, offset tab
   call
           print
   ; сегментный адрес PSP владельца участка памяти
         ax, es:[0001h]
           di, offset adrPSP+15
   mov
           wrd to hex
   call
          dx, offset adrPSP
   mov
   call
           print
           dx, offset tab
   mov
   call
           print
   ;размер участка в параграфах
   mov ax, es:[0003h]
           cx, 10h
   mov
   mul
           CX
           si, offset size s+13
 mov
            wrd to dec
   call
           dx, offset size s
   mov
   call
           print
           dx, offset tab
   mov
   call
           print
   ;последние 8 байт
   push ds
   push es
           ds
   pop
         dx, 08h
   mov
            di, dx
   mov
            cx, 8
   mov
tag2:
           cx,0
 cmp
 jе
            tag3
   mov dl, byte PTR [di] call print_symb
   mov
   dec
           CX
   inc
            di
   jmp
            tag2
```

```
taq3:
 pop ds
           dx, offset endl
 mov
           print
   call
   ;проверка, последний блок или нет
          byte ptr es:[0000h], 5ah
   jе
            quit
   ;адрес следующего блока
   mov
         ax, es
           ax, es:[0003h]
   add
   inc
           ax
   mov
           es, ax
           tag1
   jmp
quit:
   xor
          ax, ax
           ah, 4ch
   int
           21h
SegEnd:
PCinfo ENDS
      END
            START
Файл lab3 3.asm
PCinfo segment
      assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
     100h
 org
start:
 jmp
            begin
 ;data
         db 'Amount of available memory:
                                                     b$'
 av mem
                                                  Kb$'
           db 'Size of extended memory:
 ex mem
 mcb db 'List of memory control blocks:$'
 typeMCB db 'MCB type: 00h$'
 adrPSP db 'PSP adress: 0000h$'
 size_s
           db 'Size:
          db 13, 10, '$'
   endl
           db 9,'$'
   tab
   error
           db 'ERROR! Memory can not be allocated!$'
tetr_to_hex proc near
   and al, OFh
   cmp
           al, 09
          next
   jbe
   add
           al, 07
next:
           al, 30h
   add
   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
   xchq
           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
             bh, ah
    mov
    call
             byte_to_hex
    mov
             [di], ah
    dec
             di
    mov
             [di], al
    dec
             di
    mov
             al, bh
    call
             byte to hex
             [di], ah
    mov
    dec
             di
    mov
             [di], al
    pop
             bx
    ret
   wrd_to_hex endp
;Перевод в 10 cc, si - адрес поля младшей цифры
byte to dec proc near
    push
            CX
             dx
    push
             ah, ah
    xor
             dx, dx
    xor
             cx, 10
    mov
loop bd:
    div
             CX
             dl, 30h
    or
             [si], dl
    mov
             si
    dec
             dx, dx
    xor
             ax, 10
    cmp
             loop bd
    jae
             al, \overline{0}0h
    cmp
             end l
    jе
             al, 30h
    or
             [si], al
    mov
end 1:
    pop
             dx
    pop
             CX
    ret
   byte to dec endp
wrd to dec proc near
    push
           CX
    push
             dx
             cx, 10
    mov
wloop bd:
    div
             CX
             dl, 30h
    or
             [si], dl
    mov
    dec
             si
 xor dx, dx
            ax, 10
    cmp
    jae
             wloop bd
             al, 0\overline{0}h
    cmp
    jе
             wend 1
             al, \overline{3}0h
    or
```

```
[si], al
   mov
wend 1:
          dx
   pop
   pop
          CX
   ret
  wrd_to_dec endp
;вывод строки
print proc near
   push ax
   push
          dx
   mov ah, 09h
          21h
   int
          dx
   pop
          ax
   pop
   ret
  print endp
;вывод символа
print symb proc near
push ax
push dx
mov
          ah, 02h
 int
          21h
          dx
pop
          ax
pop
ret
  print_symb endp
begin:
;количество доступной памяти
mov ah, 4Ah
 mov bx, Offffh
 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
        bx, 10h
   mov
          dx, dx
   xor
          bx
   div
   inc
           ax
       bx, ax al, 0 ah, 4Ah
   mov
   mov
   mov
          21h
   int
```

; запрос памяти

```
ax, ax
 xor
          ah, 48h
bx, 1000h
 mov
 mov
           21h
 int
           mem ok
 jnc
           dx, offset error
 mov
 call print
 mov
          dx, offset endl
 call print
mem_ok:
;размер расширенной памяти
       al, 30h
mov
           70h, al
 out
           al, 71h
 in
          bl, al ;младший байт
          al, 31h
           70h, al
 out
           al, 71h ;старший байт
           ah, al
           al, bl
           si, offset ex mem+34
 xor dx, dx
 call wrd to dec
           dx, offset ex_mem
 call print
           dx, offset endl
 mov
 call print
;цепочка блоков управления памятью
   mov dx, offset mcb
 call print mov dx, offset endl
 call print
           ah, 52h
   mov
           21h
   int
           ax, es:[bx-2]
   mov
           es, ax
   mov
   ;тип МСВ
tag1:
 mov al, es:[0000h]
   call byte_to_hex
           di, offset typeMCB+10
   mov
   mov
           [di], ax
           dx, offset typeMCB
          print
   call
           dx, offset tab
   mov
           print
   call
   ; сегментный адрес PSP владельца участка памяти
        ax, es:[0001h]
   mov
            di, offset adrPSP+15
   mov
           wrd to hex
   call
          dx, offset adrPSP
   mov
          print
   call
          dx, offset tab
   mov
   call
          print
```

```
;размер участка в параграфах
        ax, es:[0003h]
cx, 10h
   mov
   mov
   mul
            CX
            si, offset size_s+13
 mov
            wrd_to_dec
   call
            dx, offset size_s
   mov
   call
           print
   mov
            dx, offset tab
           print
   call
   ;последние 8 байт
   push
           ds
   push
            es
            ds
   pop
           dx, 08h
   mov
            di, dx
   mov
            cx, 8
tag2:
 cmp
             cx,0
 jе
            tag3
            dl, byte PTR [di]
   call
           print symb
   dec
            CX
   inc
             di
            tag2
   jmp
tag3:
 pop ds
            dx, offset endl
 mov
            print
   call
    ;проверка, последний блок или нет
         byte ptr es:[0000h], 5ah
            quit
    jе
    ;адрес следующего блока
         ax, es
   mov
            ax, es:[0003h]
   add
    inc
            ax
            es, ax
   mov
    jmp
            tag1
quit:
            ax, ax ah, 4ch
   mov
   int
             21h
SegEnd:
PCinfo ENDS
      END
             START
Файл lab3 4.asm
PCinfo segment
      assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
      100h
 org
start:
 jmp
            begin
```

```
;data
          db 'Amount of available memory:
db 'Size of extended memory:
 av mem
                                                        b$'
                                                     Kb$'
 ex mem
 mcb db 'List of memory control blocks:$'
 typeMCB db 'MCB type: 00h$'
         db 'PSP adress: 0000h$'
 adrPSP
             db 'Size:
                                b$'
 size s
    endl
           db 13, 10, '$'
           db 9,'$'
    tab
           db 'ERROR! Memory can not be allocated!$'
    error
tetr to hex proc near
          al, OFh
    and
            al, 09
    cmp
    jbe
           next
   add
           al, 07
next:
             al, 30h
   add
   tetr to hex endp
;Байт в al переводится в два символа 16-ричного числа в ах
byte to hex proc near
    push
            CX
             ah, al
   mov
    call
             tetr to hex
             al, ah
    xchq
             cl, 4
   mov
             al, cl
    shr
             tetr to hex ;В al старшая цифра, в ah младшая
   call
   pop
   ret
   byte to hex endp
;Перевод в 16 сс 16-ти разрядного числа
; ах - число, di - адрес последнего символа
wrd_to_hex proc near
    push
            bx
             bh, ah
    mov
            byte to hex
    call
             [di], ah
    mov
    dec
            di
             [di], al
    mov
    dec
            di
             al, bh
    mov
    call
            byte to hex
    mov
             [di], ah
    dec
             di
   mov
             [di], al
   pop
             bx
   ret
   wrd 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
           dl, 30h
    or
```

```
[si], dl
    mov
    dec
             si
            dx, dx
    xor
    cmp
             ax, 10
             loop bd
    jae
             al, \overline{00}h
    cmp
             end_l
    jе
             al, 30h
    or
             [si], al
    mov
end_l:
             dx
    pop
             CX
    pop
    ret
   byte to dec endp
wrd_to_dec proc near
    push
             dx
    push
             cx, 10
   mov
wloop bd:
    div
             CX
             dl, 30h
            [si], dl
    dec
            si
 xor dx, dx
            ax, 10
    cmp
    jae
             wloop bd
    cmp
             al, 00h
             wend 1
    jе
             al, \overline{30h}
    or
             [si], al
    mov
wend 1:
             dx
    pop
             CX
    pop
    ret
   wrd_to_dec endp
;вывод строки
print proc near
    push ax
            dx
    push
             ah, 09h
    mov
             21h
    int
             dx
    pop
    pop
             ax
    ret
   print endp
;вывод символа
print_symb proc near
push ax push dx
 mov
             ah, 02h
 int
             21h
 pop
             dx
             ax
 pop
 ret
  print symb endp
```

begin:

```
; количество доступной памяти
mov ah, 4Ah
mov bx, 0ffffh
     21h
 int
 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
           ah, 48h
          bx, 1000h
int
           21h
         mem_ok
dx, offset error
jnc
mov
call print
          dx, offset endl
call print
mem ok:
;освобождение памяти
   mov ax, offset SegEnd
           bx, 10h
   mov
         dx, dx
   xor
           bx
   div
           ax
   inc
          bx, ax
   mov
        bx, ax al, 0 ah, 4Ah
   mov
   mov
           21h
   int
;размер расширенной памяти
 mov al, 30h
           70h, al
 out
           al, 71h
bl, al ;младший байт
al, 31h
 in
            70h, al
           al, 71h ;старший байт
ah, al
 mov
            al, bl
 mov
           si, offset ex mem+34
 mov
 xor dx, dx
 call wrd to dec
           dx, offset ex mem
 mov
 call print
 mov dx, offset endl
 call print
```

;цепочка блоков управления памятью

```
mov dx, offset mcb
call print mov dx, offset endl
 call print
          ah, 52h
   mov
           21h
   int
            ax, es:[bx-2]
   mov
   mov
           es, ax
   ;тип МСВ
tag1:
 mov al, es:[0000h]
   call byte_to_hex
           di, offset typeMCB+10
   mov
           [di], ax
   mov
            dx, offset typeMCB
   mov
   call
           print
   mov
            dx, offset tab
   call
           print
   ; сегментный адрес PSP владельца участка памяти
         ax, es:[0001h]
            di, offset adrPSP+15
   mov
   call
           wrd_to_hex
            dx, offset adrPSP
   mov
   call
           print
            dx, offset tab
   mov
   call
           print
   ;размер участка в параграфах
   mov ax, es:[0003h]
           cx, 10h
   mov
   mul
            CX
            si, offset size s+13
 mov
           wrd to dec
   call
           dx, offset size s
   mov
           print
   call
            dx, offset tab
   mov
   call
           print
   ;последние 8 байт
   push ds
   push
           es
   pop
            ds
           dx, 08h
   mov
            di, dx
   mov
            cx, 8
   mov
tag2:
            cx,0
 cmp
            tag3
 jе
            dl, byte PTR [di]
   mov
   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
   jе
         quit
   ;адрес следующего блока
   mov ax, es
   add
         ax, es:[0003h]
   inc
         ax
      es, ax
   mov
   jmp
         tag1
quit:
  xor ax, ax
         ah, 4ch
  mov
          21h
  int
SegEnd:
PCinfo ENDS
     END START
```