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. 7381	 Габов Е. С.
Преподаватель	 Ефремов М. А.

Санкт-Петербург 2019

Цель работы

Исследование структур данных и работы функций управления памятью ядра операционной системы.

Необходимые сведения для составления программы

Учет занятой и свободной памяти ведётся при помощи списка блоков управления памятью MCB (Memory Control Block). MCB занимает 16 байт (параграф) и располагается всегда с адреса кратного 16 (адрес сегмента ОП) и находится в адресном пространстве непосредственно перед тем участком памяти, которым он управляет.

MCR	широт	следующ	vio cmr	www.nv.
$M \cup D$	имеет	слебующ	ую спц	γκιπγρη.

Смещение	Длина поля (байт)	Содержимое поля	
00h	1	тип MCB: 5Ah, если последний в списке, 4Dh, если не последний	
01h	2	4Dh, если не последний Сегментный адрес PSP владельца участка памяти либо 0000h - свободный участок, 0006h - участок принадлежит драйверу ОЅ XMS UMB 0007h - участок является исключенной верхней памятью драйверов 0008h - участок принадлежит MS DOS FFFAh - участок занят управляющим блоком 386MAX UMB FFFDh - участок заблокирован 386MAX FFFEh - участок принадлежит 386MAX	
03h	2	Размер участка в параграфах	
05h	3	Зарезервирован	
08h	8	"SC" - если участок принадлежит MS DOS, то в нем системный код "SD" - если участок принадлежит MS DOS, то в нем системные данные	

По сегментному адресу и размеру участка памяти, контролируемого этим МСВ можно определить местоположение следующего МСВ в списке.

Адрес первого MCB хранится во внутренней структуре MS DOS, называемой "List of Lists" (список списков). Доступ к указателю на эту структуру можно получить используя функцию 52h "Get List of Lists" int 21h. В результате выполнения этой функции ES:BX будет указывать на список списков. Слово по адресу ES:[BX-2] и есть адрес самого первого MCB.

Размер расширенной памяти находится в ячейках 30h, 31h CMOS. CMOS это энергонезависимая память, в которой хранится информация о конфигурации ПЭВМ. Объем памяти составляет 64 байта. Размер расширенной памяти в Кбайтах можно определить обращаясь к ячейкам CMOS следующим образом:

```
mov AL, 30h; запись адреса ячейки CMOS out 70h, AL in AL, 71h; чтение младшего байта mov BL, AL; размера расширенной памяти mov AL, 31h; запись адреса ячейки CMOS out 70h, AL in AL, 71h; чтение старшего байта; размера расширенной памяти
```

Порядок выполнения работы:

- Необходимо написать и отладить программный модуль типа .COM, который выбирает и распечатывает следующую информацию:
 - 1. Количество доступной памяти.
 - 2. Размер расширенной памяти.
 - 3. Выводит цепочку блоков управления памятью.
 - Адреса при выводе представляются шестнадцатеричными числами. Объём памяти функциями управления памятью выводится в параграфах. Необходимо преобразовать его в байты и выводить в виде десятичных чисел. Последние восемь байт МСВ выводятся как символы, не следует преобразовывать их в шестнадцатеричные числа.
- Далее необходимо изменить программу таким образом, чтобы она освобождала память, которую она не занимает. Для этого используйте функцию 4Ah прерывания 21h (пример в разделе «Использование функции 4AH»). Хвост командной строки в символьном виде.
- Затем необходимо изменить программу еще раз таким образом, чтобы после освобождения памяти, программа запрашивала 64Кб памяти функцией 48Н прерывания 21Н.
- Далее нужно изменить первоначальный вариант программы, запросив 64Кб памяти функцией 48Н прерывания 21Н до освобождения памяти. Оформить отчёт и ответить на контрольные вопросы.

Ход работы:

1. Запуск .com файла без выделения и освобождения памяти (все доступные 648912 байт отдаются программе):

```
W:\>lab3_1.com
Size of available memory: 648912 byte
Size of extended memory: 245760 byte
 MCB #01
 Addr: 016F
 Owner: Area belongs to MS DOS
 Size: 16 byte
 Name:
 MCB #02
Addr: 0171
Owner: Empty area
Size: 64 byte
Name:
 CO# US
Addr: 0176
Owner: 0040
Size: 256 byte
Name:
MCB #04
Addr: 0187
Owner: 0192
Size: 144 byte
Name:
MCB #05
Addr: 0191
Owner: 0192
Size: 648912 byte
Name: LAB3_1
Press any key...
```

2. Запуск .com файла с освобождением незанятой памяти:

```
Size of available memory: 648912 byte
Freeing memory...
Suscess!
Size of extended memory: 245760 byte
MCB #01
Addr: 016F
Owner: Area belongs to MS DOS
Size: 16 byte
Name:
MCB #02
Addr: 0171
Owner: Empty area
Size: 64 byte
Name:
MCB #03
Addr: 0176
Owner: 0040
Size: 256 byte
Name:
```

```
MCB #04
Addr: 0187
Owner: 0192
Size: 144 byte
Name:

MCB #05
Addr: 0191
Owner: 0192
Size: 1904 byte
Name: LAB3_2

MCB #06
Addr: 0209
Owner: Empty area
Size: 646992 byte
Name:
Press any key...
```

3. Запуск .com файла с освобождением незанятой памяти и выделением дополнительных 64 Кбайт памяти:

```
Size of available memory: 648912 byte
Freeing memory...
Suscess!
Getting memory...
Suscess!
Size of extended memory: 245760 byte
MCB #01
Addr: 016F
Owner: Area belongs to MS DOS
Size: 16 byte
Name:
MCB #02
Addr: 0171
Owner: Empty area
Size: 64 byte
Name:
MCB #03
Addr: 0176
Owner: 0040
Size: 256 byte
Name:
MCB #04
Addr: 0187
Owner: 0192
Size: 144 byte
Name:
```

```
1CB #05
Addr: 0191
Owner: 0192
Size: 1904 byte
Name: LAB3_3
MCB #06
Addr: 0209
Owner: 0192
Size: 65536 byte
Name: LAB3_3
MCB #07
Addr: 120A
Owner: Empty area
Size: 581440 byte
Name:
Press any key..
```

4. Запуск .com файла с выделением дополнительных 64 Кбайт памяти, затем с освобождением незанятой памяти:

```
Size of available memory: 648912 byte
Getting memory...
Can't get memory
W:\>
```

Ответы на контрольные вопросы:

1. Что означает «доступный объем памяти»?

Доступный объем памяти — количество памяти, не занятое другими программами, которое загрузчик выделяет программе при загрузке в основную память, после чего программа сама распоряжается этой памятью с помощью тех же функций, что использовал загрузчик. Поэтому программа может освободить не используемую память, или запросить расширение текущего объема памяти. Если программа запрашивает слишком большой объем памяти (больше, чем размер ее доступной памяти) система откажет ей в этом.

- 2. Где МСВ блок Вашей программы в списке? Расположен в последнем блоке программы.
- 3. Какой размер памяти занимает программа в каждом случае? Для программы выделяется всегда 648912 байт = 634 Кбайт.

Вывод

В результате выполнения данной лабораторной работы были исследованы структуры данных и работа функций управления памятью ядра операционной системы.

Приложение А

```
1) Lab3_1.asm
PCinfosegment
             assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
             100h
      org
start:
                   begin
      jmp
      ;data
                   db 'Amount of available memory:
                                                            b$'
      av mem
                   db 'Size of extended memory:
                                                        Kb$'
      ex mem
      mcb db 'List of memory control blocks:$'
      typeMCB db 'MCB type: 00h$'
                   db 'PSP adress: 0000h$'
      adrPSP
      size_s
                   db 'Size:
                                 b$'
  endl
            db 13, 10, '$'
  tab
            db
                   9,'$'
tetr_to_hex proc near
            al, 0Fh
  and
            al. 09
  cmp
  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
            \mathsf{CX}
  mov
            ah, al
            tetr to hex
  call
  xchg
            al, ah
            cl, 4
  mov
  shr al, cl
            tetr to hex; B al старшая цифра, в ah младшая
  call
            \mathsf{CX}
  pop
  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
            [di], ah
  mov
  dec
            di
            [di], al
  mov
            di
  dec
            al, bh
  mov
            byte to hex
  call
            [di], ah
  mov
  dec
            di
```

```
[di], al
  mov
  pop
             bx
  ret
 wrd_to_hex endp
;Перевод в 10 сс, si - адрес поля младшей цифры
byte_to_dec proc near
  push
             \mathsf{CX}
  push
             dx
  xor ah, ah
  xor dx, dx
             cx, 10
  mov
loop bd:
  div cx
             dl, 30h
  or
             [si], dl
  mov
             si
  dec
  xor dx, dx
             ax, 10
  cmp
  jae loop_bd
             al, 00h
  cmp
             end I
  je
             al, 30h
  or
             [si], al
  mov
end_l:
  pop
             dx
  pop
             \mathsf{CX}
  ret
 byte_to_dec endp
wrd_to_dec proc near
  push
             \mathsf{CX}
  push
             dx
             cx, 10
  mov
wloop bd:
  div cx
  or dl, 30h
             [si], dl
  mov
  dec
             si
             dx, dx
      xor
             ax, 10
  cmp
  jae wloop_bd
             al, 00h
  cmp
  je
             wend I
             al, 30h
  or
  mov
             [si], al
wend I:
             dx
  pop
             \mathsf{CX}
  pop
  ret
 wrd_to_dec endp
;вывод строки
print proc near
  push
             ax
             dx
  push
```

```
ah, 09h
  mov
  int 21h
           dx
  pop
  pop
           ax
  ret
 print endp
;вывод символа
print_symb proc near
     push ax
     push dx
                 ah, 02h
     mov
     int
                 21h
                 dx
      pop
      pop
                 ax
     ret
 print symb endp
begin:
;количество доступной памяти
     mov ah, 4Ah
     mov bx, 0ffffh
     int
           21h
                 dx, dx
     xor
     mov ax, bx
     mov cx, 10h
     mul
           \mathsf{CX}
     mov si, offset av mem+37
           wrd_to_dec
     call
     mov dx, offset av_mem
     call
           print
                 dx, offset endl
      mov
     call
           print
;размер расширенной памяти
                 al, 30h
     mov
                 70h, al
     out
                 al, 71h
     in
                 bl, al ;младший байт
     mov
                 al, 31h
     mov
     out
                 70h, al
                 al, 71h ;старший байт
     in
     mov
                 ah, al
                 al, bl
     mov
                 si, offset ex_mem+34
     mov
           dx, dx
     xor
           wrd_to_dec
     call
                 dx, offset ex_mem
     mov
```

```
print
      call
      mov
                   dx, offset endl
      call
            print
;цепочка блоков управления памятью
                   dx, offset mcb
  mov
  call
             print
                   dx, offset endl
      mov
      call
            print
                   ah, 52h
  mov
  int 21h
            ax, es:[bx-2]
  mov
            es, ax
  mov
  ;тип МСВ
tag1:
            al, es:[0000h]
      mov
            byte_to_hex
  call
                   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]
  mov
  mov
            di, offset adrPSP+15
            wrd to hex
  call
                   dx, offset adrPSP
  mov
  call
            print
                   dx, offset tab
  mov
  call
            print
  ;размер участка в параграфах
            ax, es:[0003h]
  mov
            cx, 10h
  mov
  mul
            \mathsf{CX}
                   si, offset size s+13
      mov
            wrd to dec
  call
                   dx, offset size s
  mov
  call
             print
  mov
                   dx, offset tab
  call
            print
  ;последние 8 байт
            ds
  push
  push
             es
            ds
  pop
            dx, 08h
  mov
            di, dx
  mov
```

```
cx, 8
  mov
tag2:
                 cx,0
     cmp
                 tag3
     je
                 dl, byte PTR [di]
  mov
  call print_symb
  dec
           \mathsf{CX}
           di
  inc
  jmp
           tag2
tag3:
     pop
           ds
                 dx, offset endl
      mov
  call
           print
  ;проверка, последний блок или нет
  cmp
           byte ptr es:[0000h], 5ah
           quit
  ie
  ;адрес следующего блока
  mov
           ax, es
  add
           ax, es:[0003h]
  inc ax
           es, ax
  mov
  jmp
           tag1
quit:
  xor ax, ax
           ah, 4ch
  mov
  int 21h
PCinfoENDS
                 END START
   2) Lab3 2.asm
      PCinfo
                 segment
                 assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
                 100h
           org
      start:
           jmp
                       begin
           ;data
                       db 'Amount of available memory:
                                                                 b$'
            av mem
           ex mem db 'Size of extended memory:
                                                             Kb$'
           mcb db 'List of memory control blocks:$'
           typeMCB db 'MCB type: 00h$'
           adrPSP
                       db 'PSP adress: 0000h$'
                       db 'Size:
                                      b$'
           size_s
                 db 13, 10, '$'
        endl
        tab
                       db
                             9.'$'
      tetr to hex proc near
        and
                 al, 0Fh
                 al, 09
        cmp
```

```
ibe
           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
           \mathsf{CX}
  mov
           ah, al
  call
           tetr to hex
           al, ah
  xchg
           cl, 4
  mov
  shr
           al, cl
  call
           tetr to hex; B al старшая цифра, в ah младшая
  pop
           \mathsf{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
           [di], al
  mov
  dec
           di
  mov
           al, bh
  call
           byte to hex
           [di], ah
  mov
           di
  dec
  mov
           [di], al
  qoq
           bx
  ret
  wrd to hex endp
;Перевод в 10 сс, si - адрес поля младшей цифры
byte to dec proc near
  push
           \mathsf{CX}
  push
           dx
           ah, ah
  xor
           dx, dx
  xor
  mov
           cx, 10
loop bd:
  div
           \mathsf{CX}
           dl, 30h
  or
  mov
           [si], dl
           si
  dec
           dx, dx
  xor
```

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

```
dx
     pop
     pop
                ax
     ret
 print symb endp
begin:
;количество доступной памяти
     mov ah, 4Ah
     mov bx, 0ffffh
     int
          21h
                dx, dx
     xor
     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
          ax, offset SegEnd
  mov
          bx, 10h
  mov
          dx, dx
  xor
  div
          bx
  inc
          ax
          bx, ax
  mov
          al. 0
  mov
          ah, 4Ah
  mov
  int 21h
;размер расширенной памяти
                al, 30h
     mov
                70h, al
     out
                al, 71h
     in
                bl, al ;младший байт
     mov
                al, 31h
     mov
                70h, al
     out
                al, 71h ;старший байт
     in
```

ah, al

al, bl

dx, dx

si, offset ex_mem+34

mov

mov

mov

xor

```
call
           wrd to dec
                 dx, offset ex mem
     mov
     call
           print
                 dx, offset endl
     mov
     call
           print
;цепочка блоков управления памятью
                 dx, offset mcb
  mov
  call
           print
                 dx, offset endl
     mov
     call
           print
                 ah, 52h
  mov
  int 21h
  mov
           ax, es:[bx-2]
  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
                 dx, offset tab
  mov
  call
           print
  ;сегментный адрес PSP владельца участка памяти
           ax, es:[0001h]
  mov
           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
           ax, es:[0003h]
  mov
           cx, 10h
  mov
  mul
           CX
                 si, offset size s+13
     mov
  call
           wrd to dec
  mov
                 dx, offset size s
           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
                   tag3
        je
                   dl, byte PTR [di]
     mov
     call
             print symb
     dec
             \mathsf{CX}
              di
     inc
     jmp
                   tag2
  tag3:
             ds
        pop
                   dx, offset endl
        mov
     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
             ah, 4ch
     mov
     int 21h
  SegEnd:
  PCinfo
              ENDS
              END START
3) Lab3 3
  PCinfo
              segment
              assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
              100h
        org
  start:
                   begin
        jmp
        ;data
```

```
b$'
                 db 'Amount of available memory:
     av mem
                                                      Kb$'
     ex mem
                 db 'Size of extended memory:
     mcb db 'List of memory control blocks:$'
     typeMCB db 'MCB type: 00h$'
     adrPSP
                 db 'PSP adress: 0000h$'
                 db 'Size:
                                b$'
     size s
           db 13, 10, '$'
  endl
                 db
                       9,'$'
  tab
  error
           db 'ERROR! Memory can not be allocated!$'
tetr to hex proc near
  and
           al, 0Fh
           al, 09
  cmp
           next
  jbe
           al, 07
  add
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
           \mathsf{CX}
  mov
           ah, al
  call
           tetr to hex
  xchg
           al, ah
  mov
           cl, 4
  shr
           al, cl
  call
           tetr to hex; B al старшая цифра, в ah младшая
  pop
           \mathsf{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
           [di], al
  mov
  dec
           di
           al, bh
  mov
  call
           byte to hex
  mov
           [di], ah
  dec
           di
           [di], al
  mov
  pop
           bx
  ret
 wrd to hex endp
```

```
;Перевод в 10 сс, si - адрес поля младшей цифры
byte_to_dec proc near
  push
            \mathsf{CX}
  push
            dx
            ah, ah
  xor
            dx, dx
  xor
            cx, 10
  mov
loop bd:
  div
            \mathsf{CX}
            dl, 30h
  or
            [si], dl
  mov
  dec
            si
            dx, dx
  xor
            ax, 10
  cmp
  jae
            loop bd
            al, 00h
  cmp
            end I
  je
            al, 30h
  or
  mov
            [si], al
end I:
            dx
  pop
  pop
            \mathsf{CX}
  ret
  byte_to_dec endp
wrd to dec proc near
  push
            \mathsf{CX}
  push
            dx
            cx, 10
  mov
wloop bd:
  div
            \mathsf{CX}
            dl, 30h
  or
            [si], dl
  mov
  dec
            si
      xor
            dx, dx
            ax, 10
  cmp
  jae
            wloop bd
            al, 00h
  cmp
            wend I
  ie
            al, 30h
  or
            [si], al
  mov
wend I:
  pop
            dx
  pop
            \mathsf{CX}
  ret
  wrd to dec endp
;вывод строки
print proc near
  push
            ax
```

push

dx

```
ah, 09h
  mov
  int 21h
          dx
  pop
  pop
          ax
  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
     mov bx, Offffh
          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
;освобождение памяти
          ax, offset SegEnd
  mov
          bx, 10h
  mov
          dx, dx
  xor
  div
          bx
  inc
          ax
          bx, ax
  mov
          al, 0
  mov
  mov
          ah, 4Ah
  int 21h
```

```
;запрос памяти
     xor
                ax, ax
                ah, 48h
     mov
                bx, 1000h
     mov
     int
                 21h
                mem ok
     inc
                dx, offset error
     mov
     call
           print
                     offset endl
     mov
                dx,
     call
           print
mem ok:
;размер расширенной памяти
                al, 30h
     mov
                 70h. al
     out
                al, 71h
     in
     mov
                bl, al ;младший байт
                al, 31h
     mov
                 70h, al
     out
                al, 71h ;старший байт
     in
                ah, al
     mov
                al, bl
     mov
                si, offset ex_mem+34
     mov
           dx, dx
     xor
           wrd_to_dec
     call
                dx, offset ex mem
     mov
     call
           print
                dx, offset endl
     mov
     call
           print
;цепочка блоков управления памятью
                dx, offset mcb
  mov
  call
           print
                dx, offset endl
     mov
     call
           print
                ah, 52h
  mov
  int 21h
  mov
           ax, es:[bx-2]
  mov
           es, ax
  ;тип МСВ
tag1:
     mov al, es:[0000h]
  call
           byte to hex
                 di, offset typeMCB+10
  mov
  mov
           [di], ax
                dx, offset typeMCB
  mov
```

```
call
           print
                 dx, offset tab
  mov
  call
           print
  ;сегментный адрес PSP владельца участка памяти
           ax, es:[0001h]
  mov
           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
           ax, es:[0003h]
  mov
           cx, 10h
  mov
  mul
           \mathsf{CX}
     mov
                 si, offset size_s+13
  call
           wrd to dec
  mov
                 dx, offset size s
  call
           print
                 dx, offset tab
  mov
  call
           print
  ;последние 8 байт
           ds
  push
  push
           es
           ds
  pop
           dx, 08h
  mov
           di, dx
  mov
           cx, 8
  mov
tag2:
                 cx,0
     cmp
                 tag3
     je
                 dl, byte PTR [di]
  mov
  call
           print symb
  dec
           \mathsf{CX}
           di
  inc
                 tag2
  jmp
tag3:
     pop
           ds
                 dx, offset endl
     mov
  call
           print
  ;проверка, последний блок или нет
           byte ptr es:[0000h], 5ah
  cmp
           quit
  je
```

```
;адрес следующего блока
             ax, es
     mov
     add
              ax, es:[0003h]
     inc
              ax
              es, ax
     mov
             tag1
     jmp
  quit:
             ax, ax
     xor
              ah. 4ch
     mov
     int 21h
  SegEnd:
              ENDS
  PCinfo
              END START
4) Lab3 4.asm
  PCinfo
             segment
              assume cs:PCinfo, ds:PCinfo, es:nothing, ss:nothing
              100h
        org
  start:
        jmp
                   begin
        ;data
                                                           b$'
                   db 'Amount of available memory:
        av mem
                   db 'Size of extended memory:
                                                        Kb$'
        ex mem
        mcb db 'List of memory control blocks:$'
        typeMCB db 'MCB type: 00h$'
                   db 'PSP adress: 0000h$'
        adrPSP
        size s
                   db 'Size:
                                  b$'
             db 13, 10, '$'
     endl
                         9.'$'
     tab
                   db
              db 'ERROR! Memory can not be allocated!$'
     error
  tetr to hex proc near
     and
             al, 0Fh
             al, 09
     cmp
     ibe
             next
     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
             \mathsf{CX}
     mov
             ah, al
             tetr_to_hex
     call
     xchg
             al, ah
```

```
cl, 4
  mov
  shr
           al, cl
           tetr_to_hex ;В al старшая цифра, в ah младшая
  call
  qoq
           \mathsf{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
           [di], ah
  mov
  dec
           di
           [di], al
  mov
  dec
           di
           al, bh
  mov
  call
           byte to hex
           [di], ah
  mov
  dec
           di
           [di], al
  mov
           bx
  pop
  ret
  wrd to hex endp
;Перевод в 10 сс, si - адрес поля младшей цифры
byte to dec proc near
  push
           \mathsf{CX}
  push
           dx
  xor
           ah, ah
           dx, dx
  xor
           cx, 10
  mov
loop bd:
  div
           \mathsf{CX}
           dl, 30h
  or
           [si], dl
  mov
  dec
           si
  xor
           dx, dx
           ax, 10
  cmp
           loop bd
  jae
  cmp
           al, 00h
           end I
  ie
           al, 30h
  or
           [si], al
  mov
end I:
           dx
  pop
  pop
           \mathsf{CX}
  ret
  byte_to_dec endp
```

```
wrd to dec proc near
  push
           \mathsf{CX}
           dx
  push
  mov
           cx, 10
wloop bd:
  div
           \mathsf{CX}
           dl, 30h
  or
           [si], dl
  mov
  dec
           si
           dx, dx
     xor
           ax, 10
  cmp
           wloop bd
  jae
           al, 00h
  cmp
           wend I
  je
           al, 30h
  or
           [si], al
  mov
wend I:
           dx
  pop
  pop
           \mathsf{CX}
  ret
 wrd_to_dec endp
;вывод строки
print proc near
  push
           ax
           dx
  push
  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
     int
                 21h
                 dx
     pop
     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
          wrd to dec
     call
     mov dx, offset av mem
     call
          print
     mov
                dx, offset endl
     call
          print
;запрос памяти
                ax, ax
     xor
                ah, 48h
     mov
                bx, 1000h
     mov
                21h
     int
                mem ok
     inc
                dx, offset error
     mov
     call
          print
                dx, offset endl
     mov
     call
          print
mem ok:
;освобождение памяти
          ax,offset SegEnd
  mov
          bx, 10h
  mov
          dx, dx
  xor
  div
          bx
  inc
          ax
          bx, ax
  mov
          al, 0
  mov
          ah, 4Ah
  mov
  int 21h
;размер расширенной памяти
                al, 30h
     mov
                70h, al
     out
                al, 71h
     in
                bl, al ;младший байт
     mov
                al, 31h
     mov
                70h, al
     out
                al, 71h ;старший байт
     in
                ah, al
     mov
                al, bl
     mov
                si, offset ex mem+34
     mov
```

```
dx, dx
     xor
           wrd_to_dec
     call
                 dx, offset ex mem
     mov
     call
           print
     mov
                 dx, offset endl
     call
           print
;цепочка блоков управления памятью
                 dx, offset mcb
  mov
  call
           print
                 dx, offset endl
     mov
     call
           print
  mov
                 ah, 52h
  int 21h
  mov
           ax, es:[bx-2]
  mov
           es, ax
  ;тип МСВ
tag1:
     mov al, es:[0000h]
           byte_to_hex
  call
                 di, offset typeMCB+10
  mov
           [di], ax
  mov
  mov
                 dx, offset typeMCB
  call
           print
  mov
                 dx, offset tab
  call
           print
  ;сегментный адрес PSP владельца участка памяти
           ax, es:[0001h]
  mov
           di, offset adrPSP+15
  mov
  call
           wrd_to_hex
                 dx, offset adrPSP
  mov
  call
           print
  mov
                 dx, offset tab
  call
           print
  ;размер участка в параграфах
           ax, es:[0003h]
  mov
  mov
           cx, 10h
  mul
           CX
                 si, offset size s+13
     mov
  call
           wrd to dec
  mov
                 dx, offset size s
  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:
     cmp
                cx,0
                tag3
     je
                dl, byte PTR [di]
  mov
           print_symb
  call
  dec
           \mathsf{CX}
           di
  inc
  jmp
                tag2
tag3:
     pop ds
                dx, offset endl
     mov
           print
  call
  ;проверка, последний блок или нет
           byte ptr es:[0000h], 5ah
  cmp
           quit
  je
  ;адрес следующего блока
  mov
           ax, es
           ax, es:[0003h]
  add
  inc
           ax
           es, ax
  mov
  jmp
           tag1
quit:
           ax, ax
  xor
           ah, 4ch
  mov
  int 21h
SegEnd:
PCinfo
           ENDS
           END START
```