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

Кафедра МОЭВМ

### ЛАБОРАТОРНАЯ РАБОТА № 3

по дисциплине «Операционные системы»

ТЕМА: Исследование организации управления основной памятью.

Студент гр. 9381	Николаев А.А.
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.

Был написан код .СОМ который выводит в консоль:

- Количество доступной памяти
- Размер расширенной памяти
- Выводит цепочку битов управления памятью

```
C:\>lab3 1.COM
Available memory:
                                   648912Ъ
Extended memory:
                                 15360 КЪ
List of memory control blocks:
MCB type: 4Dh
MCB type: 4Dh
MCB type: 4Dh
                 PSP adress: 0008h
                                           Size:
                                                         16 b
                 PSP adress: 0000h
                                           Size:
                                                         64 b
                 PSP adress: 0040h
                                           Size:
                                                        256 Ъ
MCB type: 4Dh
                 PSP adress: 0192h
                                           Size:
                                                        144 Ъ
MCB type: 5Ah
                 PSP adress: 0192h
                                           Size:
                                                    648912 Ъ
                                                                      LAB3_1
```

Был переписан код таким образом, чтобы он освобождал память, которую не занимает.

```
C: \times 1ab3_2.00M
Amount of available memory:
                                 648912 Ъ
Size of extended memory:
                               15360 КЪ
List of memory control blocks:
                PSP adress: 0008h
MCB type: 4Dh
                                         Size:
                PSP adress: 0000h
MCB type: 4Dh
                                         Size:
MCB tupe: 4Dh
                PSP adress: 0040h
                                         Size:
MCB type: 4Dh
                PSP adress: 0192h
                                         Size:
MCB type: 4Dh
                PSP adress: 0192h
                                         Size:
MCB type: 5Ah
                PSP adress: 0000h
                                         Size:
```

Был переписан код, таким образом чтобы он освобождал память и затем запрашивал еще 64кб памяти функцией 48h прерывания 21h.

```
Amount of available memory:
Size of extended memory:
                              15360 КЪ
List of memory control blocks:
                PSP adress: 0008h
                                        Size:
                                                    16 b
MCB type: 4Dh
MCB type: 4Dh
                PSP adress: 0000h
                                        Size:
                                                    64 b
MCB type: 4Dh
                PSP adress: 0040h
                                                   256 Ъ
                                        Size:
MCB type: 4Dh
                PSP adress: 0192h
                                        Size:
                                                   144 Ъ
MCB type: 4Dh
                PSP adress: 0192h
                                        Size:
                                                   864 Ъ
MCB type: 4Dh
                PSP adress: 0192h
                                        Size:
                                                 65536 Ъ
MCB type: 5Ah
                PSP adress: 0000h
                                        Size:
                                                582480 Ь
```

Был изменен первоначальный вариант программы, запросив 64кб памяти функцией 48h прерывания 21h.

```
C:\>lab3_4.COM
Amount of available memory:
                                 648912 Ъ
ERROR! Memory can not be allocated!
Size of extended memory:
List of memory control blocks:
                PSP adress: 0008h
MCB type: 4Dh
                                          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
MCB type: 4Dh
MCB type: 5Ah
                PSP adress: 0192h
                                          Size:
                                                      864
                PSP adress: 0000h
                                          Size:
                                                  648032
```

# Функции и структуры данных.

availableMemory	Amount of available memory: b\$	
extendedMemory	Size of extended memory: Kb\$	
mcbNums	List of memory control blocks:\$	
mcbType	MCB type: 00h\$	
pspAdress	PSP adress: 0000h\$	
sizeS	Size: b\$	
endLine	13,10 '\$'	
tab	9, '\$'	

# 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 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,	

	<u> </u>
	помещенного в АХ, в
	последовательность символов в
	десятичной системе счисления,
	помещая результат в SI. (по
	аналогии с BYTE_TO_DEC)
PRINT_STRING	осуществляет вывод строки на
	экран.
PRINT_SYMBOL	осуществляет вывод символа
	на экран (используя функцию DOS
	02h,
	прерывания 21).

# Ответы на контрольные вопросы:

# 1. Что означает "доступный объем памяти"?

Доступный объем памяти – максимальный объем памяти, который может быть доступен для запуска и выполнения программ.

# 2. Где МСВ блок вашей программы в списке?

У MCB блоков после PSP address – сегментный адрес PSP владельца участка памяти – значение 0192.

# 3. Какой размер памяти занимает программа в каждом случае?

lab 3 1 - Программа заняла всю доступную память.

lab 3 2 - 784 + 144 - объем, занимаемый самой программой.

lab 3\_3 - 784 + 144 - объем, занимаемый самой программой и еще 64 кб, которые мы попросили выделить.

lab 3\_4 - 784 + 144 - объем, занимаемый самой программой. Выделить 64кб невозможно.

**Вывод:** Была исследована организация управления памятью. Рассмотрена не страничная память и способ управления динамическими разделами.

#### ПРИЛОЖЕНИЕ А

# исходный код

### lab3 1.asm

```
LAB
      segment
            assume cs:LAB, ds:LAB, es:NOTHING, ss:NOTHING
            100h
      org
START:
                        BEGIN
            jmp
      availableMemory
                        db 'Amount of available memory:
                                                                      b$'
                        db 'Size of extended memory:
      extendedMemory
                                                                   Kb$'
                  db 'List of memory control blocks:$'
      mcbNums
      mcbType db 'MCB type: 00h$'
                  db 'PSP adress: 0000h$'
      pspAdress
                  db 'Size:
      sizeS
    endLine db 13, 10, '$'
                        9,'$'
                  db
    tab
TETR_TO_HEX proc near
    and
            al, 0Fh
            al, 09
    cmp
            next
    jbe
    add
            al, 07
next:
    add
            al, 30h
    ret
   TETR_TO_HEX endp
BYTE_TO_HEX proc near
    push
            \mathsf{CX}
            ah, al
    mov
            TETR_TO_HEX
    call
    xchg
            al, ah
            cl, 4
    mov
    shr
            al, cl
    call
            TETR_TO_HEX
    pop
    ret
   BYTE_TO_HEX endp
WRD_TO_HEX proc near
    push
            bx
    mov
            bh, ah
            BYTE_TO_HEX
    call
    mov
            [di], ah
    dec
            di
            [di], al
    mov
    dec
            di
    mov
            al, bh
            BYTE_TO_HEX
    call
            [di], ah
    mov
    dec
            di
```

```
mov
              [di], al
    pop
             bx
    ret
   WRD_TO_HEX endp
BYTE_TO_DEC proc near
    push
             \mathsf{cx}
    push
             dx
             ah, ah
    xor
    xor
             dx, dx
             cx, 10
    mov
loop_bd:
    div
             cx
                    dl, 30h
    or
              [si], dl
    mov
    dec
              si
    xor
             dx, dx
    cmp
             ax, 10
             loop_bd
    jae
    cmp
             al, 00h
    jе
                    end 1
                    al, 30h
    or
              [si], al
    mov
end_1:
    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
             \mathsf{cx}
    or
             dl, 30h
    mov
              [si], dl
    dec
              si
      xor
             dx, dx
             ax, 10
    cmp
    jae
             wloop_bd
    cmp
             al, 00h
                    wend_1
    je
                    al, 30h
    or
              [si], al
    mov
wend_1:
    pop
             dx
    pop
             \mathsf{cx}
    ret
   WRD_TO_DEC endp
PRINT_STRING proc near
    push
             ax
    push
             dx
    mov
              ah, 09h
             21h
    int
```

```
pop
           dx
   pop
           ax
   ret
  PRINT_STRING endp
PRINT_SYMBOL proc near
     push ax
     push dx
                 ah, 02h
     mov
                 21h
     int
     pop
                 dx
     pop
                 ax
     ret
  PRINT_SYMBOL endp
BEGIN:
;количество доступной памяти
     mov
           ah, 4Ah
           bx, 0ffffh
     mov
     int
           21h
           dx, dx
     xor
           ax, bx
     mov
     mov
           cx, 10h
     mul
           СХ
     mov
           si, offset availableMemory + 37
     call WRD_TO_DEC
           dx, offset availableMemory
     mov
     call PRINT_STRING
           dx, offset endLine
     mov
     call PRINT_STRING
;размер расширенной памяти
           al, 30h
     mov
     out
           70h, al
           al, 71h
     in
           bl, al ;младший байт
     mov
           al, 31h
     mov
     out
           70h, al
           al, 71h ;старший байт
           ah, al
     mov
           al, bl
     mov
           si, offset extendedMemory + 34
     mov
     xor
           dx, dx
     call WRD_TO_DEC
           dx, offset extendedMemory
     mov
     call PRINT_STRING
           dx, offset endLine
     call PRINT_STRING
;цепочка блоков управления памятью
                 dx, offset mcbNums
   mov
```

```
call
            PRINT_STRING
      mov
                   dx, offset endLine
      call PRINT_STRING
    mov
                   ah, 52h
    int
            21h
            ax, es:[bx-2]
    mov
            es, ax
    mov
    ;тип МСВ
tag1:
            al, es:[0000h]
      mov
    call
            BYTE_TO_HEX
                  di, offset mcbType + 10
    mov
    mov
            [di], ax
    mov
                   dx, offset mcbType
    call
            PRINT_STRING
                   dx, offset tab
    mov
    call
            PRINT_STRING
    ; сегментный адрес PSP владельца участка памяти
    mov
            ax, es:[0001h]
            di, offset pspAdress + 15
    mov
    call
            WRD_TO_HEX
    mov
                   dx, offset pspAdress
    call
            PRINT STRING
    mov
                   dx, offset tab
    call
            PRINT_STRING
    ;размер участка в параграфах
    mov
            ax, es:[0003h]
    mov
            cx, 10h
    mul
            \mathsf{cx}
                   si, offset sizeS + 13
      mov
    call
            WRD TO DEC
                   dx, offset sizeS
    mov
    call
            PRINT_STRING
    mov
                   dx, offset tab
    call
            PRINT_STRING
    ;последние 8 байт
    push
            ds
    push
            es
            ds
    pop
            dx, 08h
    mov
            di, dx
    mov
            cx, 8
    {\sf mov}
tag2:
                   cx,0
      cmp
      jе
                   tag3
                   dl, byte PTR [di]
    mov
            PRINT_SYMBOL
    call
```

```
dec
            cx
    inc
                  di
    jmp
                  tag2
tag3:
      pop
            ds
                  dx, offset endLine
      mov
            PRINT_STRING
    call
    ;проверка на последний блок
    cmp
            byte ptr es:[0000h], 5ah
                  endProgramm
    je
    ;адрес следующего блока
    mov
            ax, es
    add
            ax, es:[0003h]
    inc
            ax
            es, ax
    mov
    jmp
            tag1
endProgramm:
    xor
            ax, ax
            ah, 4ch
    mov
            21h
    int
LAB ENDS
            END
                   START
                                     lab3 2.asm
LAB
      segment
            assume cs:LAB, ds:LAB, es:NOTHING, ss:NOTHING
      org
START:
                        BEGIN
            jmp
      availableMemory
                        db 'Amount of available memory:
                                                                     b$'
      extendedMemory
                        db 'Size of extended memory:
                                                                  Kb$'
                 db 'List of memory control blocks:$'
      mcbType db 'MCB type: 00h$'
      pspAdress
                  db 'PSP adress: 0000h$'
                  db 'Size:
                                      b$'
      sizeS
    endLine db 13, 10, '$'
                        9,'$'
    tab
                  db
TETR_TO_HEX proc near
    and
            al, 0Fh
    cmp
            al, 09
    jbe
            next
    add
            al, 07
next:
    add
            al, 30h
    ret
   TETR_TO_HEX endp
BYTE_TO_HEX proc near
    push
            \mathsf{cx}
    mov
            ah, al
```

```
TETR_TO_HEX
    call
    xchg
              al, ah
    mov
              cl, 4
    shr
              al, cl
              TETR_TO_HEX
    call
    pop
    ret
   BYTE_TO_HEX endp
WRD_TO_HEX proc near
    push
              bx
    mov
              bh, ah
              {\tt BYTE\_TO\_HEX}
    call
    mov
              [di], ah
    dec
              di
              [di], al
    mov
    dec
              di
    mov
              al, bh
              BYTE_TO_HEX
    call
    mov
              [di], ah
    dec
              di
              [di], al
    \text{mov}
    pop
              bx
    ret
   WRD_TO_HEX endp
WRD_TO_DEC proc near
    push
              \mathsf{C}\mathsf{X}
    push
              dx
    mov
              cx, 10
wloop_bd:
    div
              \mathsf{cx}
              dl, 30h
    or
    \text{mov}
              [si], dl
    dec
              si
              dx, dx
       xor
    cmp
              ax, 10
    jae
              wloop_bd
              al, 00h
    \mathsf{cmp}
                     wend_1
    je
    or
                     al, 30h
    mov
              [si], al
wend_1:
    pop
              dx
    pop
              \mathsf{C}\mathsf{X}
    ret
   WRD_TO_DEC endp
PRINT_STRING proc near
    push
              ax
              dx
    push
              ah, 09h
    mov
              21h
    int
              dx
    pop
    pop
              ax
    ret
```

```
PRINT_STRING endp
PRINT_SYMBOL proc near
      push ax
      push dx
                  ah, 02h
      mov
      int
                  21h
                  dx
      pop
                  ax
      pop
      ret
   PRINT_SYMBOL endp
BEGIN:
;количество доступной памяти
            ah, 4Ah
      mov
            bx, 0ffffh
      mov
      int
            21h
                  dx, dx
      xor
      mov
            ax, bx
      mov
            cx, 10h
      mul
            \mathsf{C}\mathsf{X}
      mov
            si, offset availableMemory + 37
      call WRD_TO_DEC
            dx, offset availableMemory
      mov
            PRINT_STRING
      call
                  dx, offset endLine
      mov
      call PRINT_STRING
;освобождение памяти
            ax, offset SegEnd
    mov
    mov
            bx, 10h
            dx, dx
    xor
    div
            bx
    inc
            ax
            bx, ax
    mov
            al, 0
    mov
            ah, 4Ah
    mov
    int
            21h
;размер расширенной памяти
      mov
                  al, 30h
                  70h, al
      out
                  al, 71h
      in
                  bl, al ;младший байт
      mov
                  al, 31h
      mov
                  70h, al
      out
                  al, 71h ;старший байт
      in
      mov
                  ah, al
                  al, bl
      mov
```

si, offset extendedMemory + 34

mov xor

dx, dx

```
call WRD_TO_DEC
      mov
                  dx, offset extendedMemory
      call
            PRINT STRING
                  dx, offset endLine
      mov
      call
            PRINT_STRING
;цепочка блоков управления памятью
                  dx, offset mcbNums
    mov
            PRINT_STRING
    call
                  dx, offset endLine
      mov
      call PRINT_STRING
                  ah, 52h
    mov
            21h
    int
            ax, es:[bx-2]
    mov
    mov
            es, ax
    ;тип МСВ
tag1:
            al, es:[0000h]
      mov
    call
            BYTE_TO_HEX
                  di, offset mcbType + 10
    mov
    mov
            [di], ax
                  dx, offset mcbType
    mov
            PRINT_STRING
    call
                  dx, offset tab
    mov
    call
            PRINT_STRING
    ; сегментный адрес PSP владельца участка памяти
            ax, es:[0001h]
    mov
            di, offset pspAdress + 15
    mov
    call
            WRD_TO_HEX
                  dx, offset pspAdress
    mov
    call
            PRINT_STRING
    mov
                  dx, offset tab
            PRINT_STRING
    call
    ;размер участка в параграфах
            ax, es:[0003h]
    mov
            cx, 10h
    mov
            \mathsf{cx}
    mul
                  si, offset sizeS + 13
      mov
    call
            WRD_TO_DEC
                  dx, offset sizeS
    mov
            PRINT STRING
    call
                  dx, offset tab
    mov
    call
            PRINT_STRING
    ;последние 8 байт
    push
            ds
    push
            es
            ds
    pop
```

```
dx, 08h
    mov
            di, dx
    mov
            cx, 8
    mov
tag2:
      cmp
                  cx,0
                  tag3
      jе
                  dl, byte PTR [di]
    mov
            PRINT_SYMBOL
    call
            \mathsf{cx}
    dec
                  di
    inc
    jmp
                  tag2
tag3:
            ds
      pop
                  dx, offset endLine
      mov
            PRINT_STRING
    call
    ;проверка на последний блок
            byte ptr es:[0000h], 5ah
    cmp
    jе
                  endProgramm
    ;адрес следующего блока
    mov
            ax, es
    add
            ax, es:[0003h]
    inc
            ax
            es, ax
    mov
    jmp
            tag1
endProgramm:
            ax, ax
    xor
    mov
            ah, 4ch
            21h
    int
SegEnd:
LAB
    ENDS
            END
                   START
                                     lab3 3.asm
LAB
      segment
            assume cs:LAB, ds:LAB, es:NOTHING, ss:NOTHING
            100h
      org
START:
            jmp
                        BEGIN
                        db 'Amount of available memory:
                                                                      b$'
      availableMemory
                                                                   Kb$'
      extendedMemory
                        db 'Size of extended memory:
                  db 'List of memory control blocks:$'
      mcbNums
      mcbType db 'MCB type: 00h$'
                  db 'PSP adress: 0000h$'
      pspAdress
                  db 'Size:
                                      b$'
      sizeS
    endLine db 13, 10, '$'
                        9,'$'
                  db
    tab
    memoryFail
                  db 'Memory request failed$'
TETR_TO_HEX proc near
```

```
and
             al, 0Fh
    cmp
             al, 09
    jbe
             next
    add
             al, 07
next:
             al, 30h
    add
    ret
   TETR_TO_HEX endp
;Байт из al -> два символа 16-ричного числа из ах
BYTE_TO_HEX proc near
    push
             \mathsf{C}\mathsf{X}
    mov
             ah, al
             TETR_TO_HEX
    call
             al, ah
    xchg
             cl, 4
    mov
    shr
             al, cl
    call
             TETR_TO_HEX; старшая цифра в al,младшая в ah
    pop
    ret
   BYTE_TO_HEX endp
;Перевод 16-ти разрядного числа в 16 сс.
;в ах - число,в di - адрес последнего символа
WRD_TO_HEX proc near
    push
             bx
    mov
             bh, ah
             BYTE_TO_HEX
    call
    mov
             [di], ah
    dec
             di
             [di], al
    mov
             di
    dec
             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,в si - поле младшей цифры
BYTE_TO_DEC proc near
    push
             \mathsf{cx}
    push
             dx
             ah, ah
    xor
    xor
             dx, dx
             cx, 10
    mov
loop_bd:
    div
             \mathsf{C}\mathsf{X}
                   dl, 30h
    or
    mov
             [si], dl
    dec
             si
    xor
             dx, dx
    cmp
             ax, 10
    jae
             loop_bd
```

```
cmp
             al, 00h
    jе
                    end_1
    or
                    al, 30h
    mov
             [si], al
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
             \mathsf{C}\mathsf{X}
    or
             dl, 30h
    mov
             [si], dl
    dec
             si
             dx, dx
      xor
    cmp
             ax, 10
             wloop_bd
    jae
             al, 00h
    cmp
                    wend_1
    je
    or
                    al, 30h
             [si], al
    mov
wend_1:
             dx
    pop
    pop
             \mathsf{cx}
    ret
   WRD_TO_DEC endp
;вывод строки
PRINT_STRING proc near
    push
             ax
             dx
    push
             ah, 09h
    mov
             21h
    int
             dx
    pop
    pop
             ax
    ret
   PRINT_STRING endp
;вывод символа
PRINT_SYMBOL proc near
      push ax
      push dx
      mov
                    ah, 02h
      int
                    21h
                    dx
      pop
                    ax
      pop
      ret
   PRINT_SYMBOL endp
```

**BEGIN:** 

```
;количество доступной памяти
      mov
            ah, 4Ah
            bx, Offffh
      mov
      int
            21h
                  dx, dx
      xor
      mov
            ax, bx
      mov
            cx, 10h
      mul
            \mathsf{CX}
            si, offset availableMemory+37
      mov
      call WRD_TO_DEC
            dx, offset availableMemory
      mov
      call
            PRINT_STRING
                  dx, offset endLine
      mov
      call PRINT_STRING
;освобождение памяти
   mov
            ax, offset SegEnd
            bx, 10h
   mov
   xor
            dx, dx
   div
            bx
   inc
            ax
   mov
            bx, ax
            al, 0
   mov
            ah, 4Ah
   mov
            21h
   int
;запрос памяти
      xor
                  ax, ax
                  ah, 48h
      mov
                  bx, 1000h
      mov
      int
                  21h
      jnc
                  mem_ok
                  dx, offset memoryFail
      mov
      call PRINT_STRING
                        offset endLine
      mov
                  dx,
      call PRINT_STRING
mem_ok:
;размер расширенной памяти
                  al, 30h
      mov
      out
                  70h, al
                  al, 71h
      in
                  bl, al ;младший байт
      mov
                  al, 31h
      mov
                  70h, al
      out
                  al, 71h ;старший байт
      in
                  ah, al
      mov
      mov
                  al, bl
                  si, offset extendedMemory+34
      mov
            dx, dx
      xor
            WRD TO DEC
      call
```

```
mov
                  dx, offset extendedMemory
      call PRINT_STRING
      mov
                  dx, offset endLine
      call PRINT_STRING
;цепочка блоков управления памятью
    mov
                  dx, offset mcbNums
    call
            PRINT STRING
                  dx, offset endLine
      mov
      call PRINT_STRING
                  ah, 52h
    mov
    int
            21h
            ax, es:[bx-2]
    mov
    mov
            es, ax
    ;тип МСВ
tag1:
            al, es:[0000h]
      mov
    call
            BYTE_TO_HEX
                  di, offset mcbType+10
    mov
    mov
            [di], ax
                  dx, offset mcbType
    mov
    call
            PRINT_STRING
                  dx, offset tab
    mov
    call
            PRINT_STRING
    ; сегментный адрес PSP владельца участка памяти
    mov
            ax, es:[0001h]
            di, offset pspAdress+15
    mov
    call
            WRD_TO_HEX
    mov
                  dx, offset pspAdress
    call
            PRINT_STRING
                  dx, offset tab
    mov
    call
            PRINT_STRING
    ;размер участка в параграфах
    mov
            ax, es:[0003h]
    mov
            cx, 10h
    mul
            СХ
                  si, offset sizeS+13
      mov
    call
            WRD_TO_DEC
    mov
                  dx, offset sizeS
    call
            PRINT_STRING
                  dx, offset tab
    mov
    call
            PRINT_STRING
    ;последние 8 байт
    push
            ds
    push
            es
            ds
    pop
            dx, 08h
    mov
```

```
mov
            di, dx
            cx, 8
    mov
tag2:
                  cx,0
      cmp
      jе
                  tag3
                  dl, byte PTR [di]
    mov
    call
            PRINT_SYMBOL
    dec
            \mathsf{cx}
    inc
                  di
    jmp
                  tag2
tag3:
            ds
      pop
                  dx, offset endLine
      mov
            PRINT_STRING
    call
    ;проверка на последний блок
            byte ptr es:[0000h], 5ah
    cmp
    je
                  endProgramm
    ;адрес следующего блока
    mov
            ax, es
    add
            ax, es:[0003h]
    inc
            ax
    mov
            es, ax
    jmp
            tag1
endProgramm:
    xor
            ax, ax
    mov
            ah, 4ch
            21h
    int
SegEnd:
LAB
    ENDS
            END
                   START
                                     lab3 4.asm
LAB
      segment
            assume cs:LAB, ds:LAB, es:NOTHING, ss:NOTHING
      org
            100h
START:
            jmp
                        BEGIN
      availableMemory
                        db 'Amount of available memory:
                                                                     b$'
      extendedMemory
                      db 'Size of extended memory:
                                                                  Kb$'
                  db 'List of memory control blocks:$'
      mcbNums
      mcbType db 'MCB type: 00h$'
                  db 'PSP adress: 0000h$'
      pspAdress
      sizeS
                  db 'Size:
                                      b$'
    endLine db 13, 10, '$'
                        9,'$'
    tab
                  db
                  db 'ERROR! Memory can not be allocated!$'
    memoryFail
TETR_TO_HEX proc near
    and
            al, 0Fh
            al, 09
    cmp
    jbe
            next
```

```
add
            al, 07
next:
    add
            al, 30h
    ret
   TETR_TO_HEX endp
;Байт из al -> два символа 16-ричного числа из ах
BYTE_TO_HEX proc near
    push
            \mathsf{cx}
    mov
            ah, al
    call
            TETR_TO_HEX
    xchg
            al, ah
    mov
            cl, 4
    shr
            al, cl
            TETR_TO_HEX; старшая цифра в al, младшая в ah
    call
    pop
    ret
   BYTE_TO_HEX endp
;Перевод 16-ти разрядного числа в 16 сс.
;в ах - число, в 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
    mov
            al, bh
            BYTE_TO_HEX
    call
             [di], ah
    mov
    dec
            di
    mov
             [di], al
    pop
            bx
    ret
   WRD_TO_HEX endp
;Перевод в 10 сс,в si - поле младшей цифры
BYTE_TO_DEC proc near
    push
            \mathsf{cx}
    push
            dx
    xor
            ah, ah
    xor
            dx, dx
    mov
            cx, 10
loop_bd:
    div
            cx
                   dl, 30h
    or
             [si], dl
    mov
    dec
            si
    xor
            dx, dx
    cmp
            ax, 10
            loop_bd
    jae
    cmp
            al, 00h
                   end 1
    jе
                   al, 30h
    or
```

```
mov
             [si], al
end_1:
             dx
    pop
             \mathsf{cx}
    pop
    ret
   BYTE_TO_DEC endp
WRD_TO_DEC proc near
             \mathsf{cx}
    push
    push
             dx
    mov
             cx, 10
wloop_bd:
    div
             \mathsf{cx}
             dl, 30h
    or
    mov
             [si], dl
    dec
             si
             dx, dx
      xor
    cmp
             ax, 10
             wloop_bd
    jae
             al, 00h
    cmp
                    wend 1
    jе
    or
                    al, 30h
             [si], al
    mov
wend_1:
    pop
             dx
    pop
             \mathsf{C}\mathsf{X}
    ret
   WRD_TO_DEC endp
;вывод строки
PRINT_STRING proc near
    push
             ax
    push
             dx
             ah, 09h
    mov
    int
             21h
             dx
    pop
             ax
    pop
    ret
   PRINT_STRING endp
;вывод символа
PRINT_SYMBOL proc near
      push ax
      push dx
      mov
                    ah, 02h
      int
                    21h
      pop
                    dx
      pop
                    ax
      ret
   PRINT_SYMBOL endp
BEGIN:
;количество доступной памяти
             ah, 4Ah
      mov
             bx, Offffh
      mov
```

```
int
            21h
      xor
                  dx, dx
            ax, bx
      mov
      mov
            cx, 10h
      mul
            \mathsf{C}\mathsf{X}
      mov
            si, offset availableMemory+37
      call WRD_TO_DEC
            dx, offset availableMemory
      mov
      call PRINT_STRING
                  dx, offset endLine
      mov
      call PRINT_STRING
;запрос памяти
      xor
                  ax, ax
                  ah, 48h
      mov
                  bx, 1000h
      mov
      int
                  21h
      jnc
                  mem_ok
                  dx, offset memoryFail
      mov
      call PRINT_STRING
      mov
                  dx,
                        offset endLine
      call PRINT_STRING
mem_ok:
;освобождение памяти
    mov
            ax, offset SegEnd
            bx, 10h
    mov
            dx, dx
    xor
    div
            bx
    inc
            ax
    mov
            bx, ax
            al, 0
    mov
            ah, 4Ah
    mov
            21h
    int
;размер расширенной памяти
      mov
                  al, 30h
      out
                  70h, al
                  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 extendedMemory+34
      mov
            dx, dx
      xor
      call WRD_TO_DEC
                  dx, offset extendedMemory
      mov
      call PRINT_STRING
```

```
mov
                   dx, offset endLine
      call PRINT_STRING
;цепочка блоков управления памятью
                   dx, offset mcbNums
    mov
    call
            PRINT_STRING
      mov
                   dx, offset endLine
      call PRINT_STRING
    mov
                   ah, 52h
            21h
    int
            ax, es:[bx-2]
    mov
    mov
            es, ax
    ;тип МСВ
tag1:
            al, es:[0000h]
      mov
    call
            BYTE_TO_HEX
                   di, offset mcbType+10
    mov
    mov
            [di], ax
                   dx, offset mcbType
    mov
            PRINT_STRING
    call
                   dx, offset tab
    mov
    call
            PRINT_STRING
    ; сегментный адрес PSP владельца участка памяти
            ax, es:[0001h]
    mov
    mov
            di, offset pspAdress+15
    call
            WRD_TO_HEX
                   dx, offset pspAdress
    mov
    call
            PRINT_STRING
    mov
                   dx, offset tab
    call
            PRINT_STRING
    ;размер участка в параграфах
    mov
            ax, es:[0003h]
            cx, 10h
    mov
    mul
            \mathsf{C}\mathsf{X}
                   si, offset sizeS+13
      mov
    call
            WRD_TO_DEC
                   dx, offset sizeS
    mov
    call
            PRINT_STRING
    mov
                   dx, offset tab
    call
            PRINT_STRING
    ;последние 8 байт
            ds
    push
    push
            es
    pop
            ds
    mov
            dx, 08h
            di, dx
    mov
            cx, 8
    mov
```

```
tag2:
                  cx,0
      cmp
      je
                  tag3
                  dl, byte PTR [di]
    mov
    call
            PRINT_SYMBOL
    dec
            \mathsf{cx}
    inc
                  di
    jmp
                  tag2
tag3:
      pop
            ds
                  dx, offset endLine
      mov
    call
            PRINT_STRING
    ;проверка на последний блок
            byte ptr es:[0000h], 5ah
    cmp
    je
                  endProgramm
    ;адрес следующего блока
    mov
            ax, es
            ax, es:[0003h]
    add
    inc
            ax
    mov
            es, ax
    jmp
            tag1
endProgramm:
    xor
            ax, ax
            ah, 4ch
    mov
    int
            21h
SegEnd:
LAB ENDS
            END
                   START
```