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 _____ Прибылов Н.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Для исследования организации управления памятью необходимо ориентироваться на тип основной памяти, реализованнй в компьютере и способ организации, принятый в ОС. В лабораторной работе рассматривается нестраничная память и способ управления динамическими разделами. Для реализации управления памятью в этом случае строится список занятых и свободных участков памяти. Функция ядра, обеспечивающие управление основной памятью, просматривают и преобразуют этот список. Исследование структуры данных и работа функций управления памятью ядра операционной системы.

Ход работы.

1) Был написан и отлажен .COM модуль, который выбирает и распечатывает следующую информацию:

а) Количество доступной памяти.

б) Размер расширенной памяти.

в) Цепочку блоков управления памятью.

Результат работы программы:

```
X:\>lab3_1.com
- Available memory:       633 Kb,  720 b
- Extended memory:      15360 Kb
- MCB table:
MCB type: 4D    PSP addr: 0008  Mem size:     0 Kb,   16 b      End:
MCB type: 4D    PSP addr: 0000  Mem size:     0 Kb,   64 b      End:
MCB type: 4D    PSP addr: 0040  Mem size:     0 Kb,  256 b      End:
MCB type: 4D    PSP addr: 0192  Mem size:     0 Kb,  144 b      End:
MCB type: 5A    PSP addr: 0192  Mem size:   633 Kb,  720 b      End: LAB3_1
```

2) Программа была изменена: неиспользованная память освобождается функцией 4Ah прерывания 21h.

Результат работы программы:

```
X:\>lab3_2.com
- Available memory:       633 Kb,  720 b
- Extended memory:      15360 Kb
- MCB table:
MCB type: 4D    PSP addr: 0008  Mem size:     0 Kb,   16 b      End:
MCB type: 4D    PSP addr: 0000  Mem size:     0 Kb,   64 b      End:
MCB type: 4D    PSP addr: 0040  Mem size:     0 Kb,  256 b      End:
MCB type: 4D    PSP addr: 0192  Mem size:     0 Kb,  144 b      End:
MCB type: 4D    PSP addr: 0192  Mem size:     0 Kb,  816 b      End: LAB3_2
MCB type: 5A    PSP addr: 0000  Mem size:   632 Kb,  912 b      End: ;Ø°u◙ï┴
```

Как видно, размер памяти, занимаемый программой, существенно уменьшился.

3) Программа была изменена: после освобождения памяти она запрашивает 64Кб памяти функцией 48h прерывания 21h.

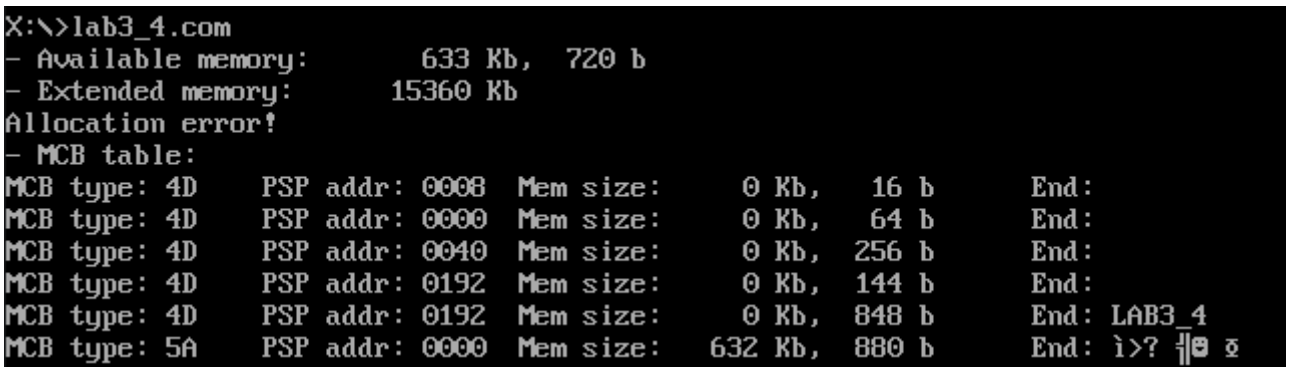
Результат работы программы:

```
X:\>lab3_3.com
- Available memory:       633 Kb,  720 b
- Extended memory:      15360 Kb
- MCB table:
MCB type: 4D    PSP addr: 0008  Mem size:     0 Kb,   16 b      End:
MCB type: 4D    PSP addr: 0000  Mem size:     0 Kb,   64 b      End:
MCB type: 4D    PSP addr: 0040  Mem size:     0 Kb,  256 b      End:
MCB type: 4D    PSP addr: 0192  Mem size:     0 Kb,  144 b      End:
MCB type: 4D    PSP addr: 0192  Mem size:     0 Kb,  848 b      End: LAB3_3
MCB type: 4D    PSP addr: 0192  Mem size:    64 Kb,    0 b      End: LAB3_3
MCB type: 5A    PSP addr: 0000  Mem size:   568 Kb,  864 b      End: crosoft
```

Как видно, программе было выделено 64Кб памяти.

4) Программы была изменена: теперь 64Кб памяти запрашивается до освобождения памяти.

Результат работы программы:



Как видно, выделение памяти не было произведено и операция завершилась ошибкой.

Функции.

Названия	Описание
TETR_TO_HEX	Перевод тетрады (4 младших бита AL) в 16-ю систему. Результат в AL.
BYTE_TO_HEX	Перевод байта AL в 16-ю систему. Результат: старшая цифра в AL, младшая в AX.
WRD_TO_HEX	Перевод слова AX в 16-ю систему. Адрес последнего символа результата в DI.
WRD_TO_DEC	Перевод слова AX в 10-ю систему. Адрес младшей цифры результата в SI.
PRINT	Вывод строки из DX на экран.

Контрольные вопросы.

1) Что означает «доступный объём памяти»?

Это область основной памяти, выделенная программе.

2) Где MCB блок вашей программы в списке?

Это те блоки, чей сегментный адрес PSP равен 0192h.

3) Какой размер памяти занимает программа в каждом случае?

1. Весь выделенный программе объём памяти — 648912 байт.

2. Непосредственно занимаемый программой объём — 816 байт.

3. Непосредственно занимаемый программой объём, а также выделенные ей 64 килобайт — всего 66384 байт.

4. Непосредственно занимаемый программой объём, поскольку 64 килобайт выделить не удалось, — 848 байт.

Выводы.

Были изучены структуры данных и работа функций управления памятью ядра операционной системы.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab3_4.asm

```
MAINSEG SEGMENT
        ASSUME CS:MAINSEG, DS:MAINSEG, ES:NOTHING, SS:NOTHING
        ORG 100H
START:  JMP BEGIN
; Данные
available_memory_string         db  '- Available memory: ',09h,'$'
memory_amount                   db  '      Kb,      b',09h,'$'
extended_memory_string          db  '- Extended memory: ',09h,'$'
extended_memory_amount          db  '      Kb',0dh,0ah,'$'
MCB_table                       db  '- MCB table:',0dh,0ah,'$'
MCB_type_string                 db  'MCB type: $'
MCB_type_number                 db  '  ',09h,'$'
PSP_segment_address_string      db  'PSP addr: $'
PSP_segment_address_number      db  '    ',09h,'$'
memory_block_size_string        db  'Mem size: $'
last_8bytes_string              db  'End: $'
endl                            db  0dh,0ah,'$'

; Процедуры
;-----------------------------------------------------
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
; Байт в AL переводится в два символа 16-числа AX
            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
            pop CX           ; В AH младшая цифра
            ret
BYTE_TO_HEX ENDP
;--------------------------------
WRD_TO_HEX PROC near
; Перевод в 16 с/с 16-ти разрядного числа
; в AX - число, DI - адрес последнего символа
            push BX
            mov BH,AH
            call BYTE_TO_HEX
            mov [DI],AH
            dec DI
```

```
                mov [DI],AL
                dec DI
                mov AL,BH
                call BYTE_TO_HEX
                mov [DI],AH
                dec DI
                mov [DI],AL
                pop BX
                ret
WRD_TO_HEX ENDP
;--------------------------------------------------
WRD_TO_DEC PROC near
; Перевод AX в 10с/с, SI - адрес поля младшей цифры
                push CX
                push DX
                ;xor AH,AH
                xor DX,DX
                mov CX,10
loop_bd:        div CX
                or DL,30h
                mov [SI],DL
                dec SI
                xor DX,DX
                cmp AX,10
                jae loop_bd
                cmp AL,00h
                je end_l
                or AL,30h
                mov [SI],AL
end_l:          pop DX
                pop CX
                ret
WRD_TO_DEC ENDP
;-------------------------------
PRINT   PROC    near
        push    ax
        mov     ah, 09h
        int     21h
        pop     ax
        ret
PRINT   ENDP
;-------------------------------



; КОД
BEGIN:
; . . . . . .
; распечатать размер доступной памяти
get_available_memory_info:
                lea     dx, available_memory_string
                call    PRINT

                mov     ah, 4Ah
                mov     bx, 0ffffh
                int     21h
                  ; в bx находится размер доступной памяти в параграфах
(16 байт)
```

```
            mov     ax, bx
            mov     cl, 6
            shr     ax, cl
            lea     si, memory_amount + 4
            call    WRD_TO_DEC

            and     bx, 0111111b
            mov     ax, bx
            mov     cl, 4
            shl     ax, cl
            lea     si, memory_amount + 13
            call    WRD_TO_DEC

            lea     dx, memory_amount
            call    PRINT
            lea     dx, endl
            call    PRINT

            mov     cx, 4
loop_clear1:
            mov     si, cx
            mov     [memory_amount + si], ' '
            mov     [memory_amount + si + 9], ' '
            loop    loop_clear1

; распечатать размер расширенной памяти
get_extended_memory_info:
            lea     dx, extended_memory_string
            call    PRINT

            mov     al, 30h
            out     70h, al
            in      al, 71h
            mov     bl, al
            ; в bl - младший байт
            mov     al, 31h
            out     70h, al
            in      al, 71h
            ; в al - старший байт

            mov     ah, al
            mov     al, bl
            ;xor     dx,dx
            lea     si, extended_memory_amount + 4
            call    WRD_TO_DEC
            lea     dx, extended_memory_amount
            call    PRINT

; респечатать цепочку MCB
get_MCB_chain_info:
            mov     ah, 52h
            int     21h
            mov     ax, es:[bx-2]
            mov     es, ax
            lea     dx, MCB_table
            call    PRINT
; распечатать информацию о текущем MCB
```

```
get_MCB_info:
            ; тип MCB - последний или нет
            lea     dx, MCB_type_string
            call    PRINT
            mov     al, es:[00h]
            call    BYTE_TO_HEX
            lea     di, MCB_type_number
            mov     [di], ax
            lea     dx, MCB_type_number
            call    PRINT

            ; сегментный адрес PSP
            lea     dx, PSP_segment_address_string
            call    PRINT
            mov     ax, es:[01h]
            lea     di, PSP_segment_address_number + 3
            call    WRD_TO_HEX
            lea     dx, PSP_segment_address_number
            call    PRINT

            ; размер участка в байтах
            lea     dx, memory_block_size_string
            call    PRINT
            mov     ax, es:[03h]

            mov     bx, ax
            mov     cl, 6
            shr     ax, cl
            lea     si, memory_amount + 4
            call    WRD_TO_DEC

            and     bx, 0111111b
            mov     ax, bx
            mov     cl, 4
            shl     ax, cl
            lea     si, memory_amount + 13
            call    WRD_TO_DEC

            lea     dx, memory_amount
            call    PRINT

            mov     cx, 4
loop_clear2:
            mov     si, cx
            mov     [memory_amount + si], ' '
            mov     [memory_amount + si + 9], ' '
            loop    loop_clear2

            ; последние 8 байт
            lea     dx, last_8bytes_string
            call    PRINT
            mov     bx, 8
loop_output_8_bytes:
            mov     dl, es:[bx]
            mov     ah, 02h
            int     21h
            inc     bx
            cmp     bx, 16
```

```
            jne     loop_output_8_bytes
            lea     dx, endl
            call    PRINT

            ; проверка на последний блок
            mov     ah, 5ah
            cmp     es:[00h], ah
            je      EXIT

            ; переход к следующему блоку, если не последний
            mov     ax, es
            add     ax, es:[03h]
            inc     ax
            mov     es, ax
            jmp     get_MCB_info


; . . . . . . . . . . . . . .
; Выход в DOS
EXIT:
            xor     AL,AL
            mov     AH,4Ch
            int     21H
MAINSEG     ENDS
            END START ; Конец модуля, START - точка входа
```