

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

ОТЧЕТ
по лабораторной работе №3
по дисциплине «Операционные системы»
Тема: Исследование организации управления основной памятью

Студент гр. 9383

Мосин К.К.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

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

Задание.

Шаг 1. Написать и отладить модуль типа .COM, который выбирает и распечатывает следующую информацию:

- 1) Количество доступной памяти.
- 2) Размер расширенной памяти.
- 3) Цепочку блоков управления памятью.

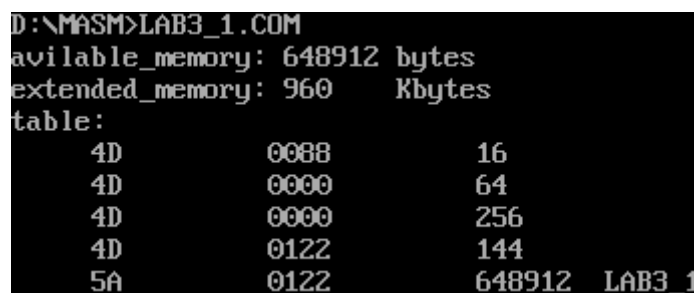
Шаг 2. Изменить .COM модуль для освобождения памяти, которую программа не занимает.

Шаг 3. Изменить .COM модуль для запроса 64Кб памяти после освобождения памяти.

Шаг 4. Изменить .COM модуль для запроса 64Кб памяти до освобождения памяти.

Выполнение работы.

Был разработан модуль типа .COM. Пример работы программы представлен на рисунке 1.



```
D:\MASM>LAB3_1.COM
available_memory: 648912 bytes
extended_memory: 960 Kbytes
table:
4D      0088      16
4D      0000      64
4D      0000     256
4D      0122     144
5A      0122    648912 LAB3_1
```

Рис. 1 - Результат выполнения первого .COM модуля

Для освобождения памяти использовалась функция 4Ah прерывания 21h. Пример работы программы представлен на рисунке 2.

```

D:\MASM>LAB3_2.COM
available_memory: 648912 bytes
extended_memory: 960 Kbytes
table:
    4D      0088      16
    4D      0000      64
    4D      0000     256
    4D      0122     144
    4D      0122     736    LAB3_2
    5A      0000    648160 [X_3]

```

Рис. 2 - Результат выполнения второго .COM модуля

Для запроса памяти использовалась функция 48H прерывания 21h. Пример работы программы представлен на рисунке 3.

```

D:\MASM>LAB3_3.COM
available_memory: 648912 bytes
extended_memory: 960 Kbytes
table:
    4D      0088      16
    4D      0000      64
    4D      0000     256
    4D      0122     144
    4D      0122     736    LAB3_3
    4D      0122    65536    LAB3_3
    5A      0000    582608 249494B

```

Рис. 3 - Результат выполнения третьего .COM модуля

При запросе памяти до ее очистки программа не может выделить память для блока, на который уже выделена память. Пример работы программы представлен на рисунке 4.

```

D:\MASM>LAB3_4.COM
available_memory: 648912 bytes
extended_memory: 960 Kbytes
memory_allocate error
table:
    4D      0088      16
    4D      0000      64
    4D      0000     256
    4D      0122     144
    5A      0122    648912    LAB3_4

```

Рис. 4 - Результат выполнения четвертого .COM модуля

Контрольные вопросы.

1) Что означает “доступный объем памяти”?

Память, отводимая программе для использования.

2) Где MCB блок Вашей программы в списке?

Каждый блок подписан соответствующим названием: LAB3_#, где # номер модуля.

Для первого модуля - 5 место.

Для второго - 5 место.

Для третьего - 5 и 6 место.

Для четвертого - 5 место.

3) Какой размер памяти занимает программа в каждом случае.

Первый модуль занимает всю память.

Второй освобождает незанятую память, поэтому достаточно для данного случая 736 байт.

В третьем случае занимается 736 байт и 64Кб выделенной памяти.

Выводы.

В ходе выполнения лабораторной работы была исследована структура данных и функций по управлению памятью.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab3_1.asm

```
testpc segment
assume cs:testpc,ds:testpc,es:nothing,ss:nothing
org 100h
start:
jmp begin

available_memory db 'available_memory:      bytes',0dh,0ah','$'
extended_memory db 'extended_memory:      Kbytes',0dh,0ah','$'
table db 'table:',0dh,0ah','$'
table_data db '                                ',0dh,0ah','$'

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 cx
mov ah,al
call tetr_to_hex
xchg al,ah
mov cl,4
shr al,cl
call tetr_to_hex
pop cx
ret
```

```
byte_to_hex endp
```

```
wrd_to_hex proc near
```

```
push bx
```

```
mov bh,ah
```

```
call byte_to_hex
```

```
mov [di],ah
```

```
dec di
```

```
mov [di],ah
```

```
dec di
```

```
mov al,bh
```

```
call byte_to_hex
```

```
mov [di],ah
```

```
dec di
```

```
mov [di],al
```

```
pop bx
```

```
ret
```

```
wrd_to_hex endp
```

```
byte_to_dec proc near
```

```
push cx
```

```
push dx
```

```
xor ah,ah
```

```
xor dx,dx
```

```
mov cx,10
```

```
loop_bd:
```

```
div cx
```

```
or dl,30h
```

```
mov [si],dl
```

```
dec si
```

```
xor dx,dx
```

```
cmp ax,10
```

```
jae loop_bd
```

```
cmp al,00h
```

```
je end_1
```

```
or al,30h
mov [si],al
end_l:
pop dx
pop cx
ret
byte_to_dec endp
```

```
wrd_to_byte proc near
mov bx,10h
mul bx
mov bx,0ah
xor cx,cx
```

```
divide:
div bx
push dx
inc cx
xor dx,dx
cmp ax,0h
jnz divide
```

```
print_symbol:
pop dx
or dl,30h
mov [si],dl
inc si
loop print_symbol
```

```
ret
wrd_to_byte endp
```

```
get_mcb_type proc near
mov di, offset table_data
add di, 5
```

```

xor ah, ah
mov al, es:[00h]
call byte_to_hex
mov [di], al
inc di
mov [di], ah

ret
get_mcb_type endp

get_psp_address proc near

mov di, offset table_data+19
mov ax, es:[01h]
call wrd_to_hex

ret
get_psp_address endp

get_mcb_size proc near
push bx
mov di, offset table_data+29
mov ax, es:[03h]
mov si, di
call wrd_to_byte

pop bx
ret
get_mcb_size endp

sc_sd proc near
mov di, offset table_data+37
    mov bx, 0h

jmp_next:

```



```

    mov dl,es:[bx + 8]
mov [di],dl
inc di
inc bx
cmp bx,8h
jne jmp_next

ret
sc_sd endp

print proc near
push ax
mov ah,09h
int 21h
pop ax
ret
print endp

begin:
mov ah,04ah
mov bx,0ffffh
int 21h

mov ax,bx

mov si,offset avilable_memory+17
call wrd_to_byte

mov dx,offset avilable_memory
call print

mov al,30h
out 70h,al
in al,71h
mov bl,al

```

```

mov al,31h
out 70h,al
in al,71h

mov si,offset extended_memory+17
call wrd_to_byte

mov dx,offset extended_memory
call print

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

mov dx,offset table
call print
print_mcb_data:
call get_mcb_type
call get_psp_address
call get_mcb_size
call sc_sd

mov ax,es:[03h]
    mov bl,es:[00h]
    mov dx,offset table_data
    call print

mov cx,es
    add ax,cx
    inc ax
    mov es,ax

cmp bl,4Dh
je print_mcb_data

```

```

xor al,al
mov ah,4ch
int 21h
testpc ends
end start

```

Название файла: lab3_2.asm

```

testpc segment
assume cs:testpc,ds:testpc,es:nothing,ss:nothing
org 100h
start:
jmp begin_ptr

```

```

available_memory db 'available_memory:      bytes',0dh,0ah,$'
extended_memory db 'extended_memory:      Kbytes',0dh,0ah,$'
table db 'table:',0dh,0ah,$'
table_data db '                                ',0dh,0ah,$'

```

```

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 cx
mov ah,al
call tetr_to_hex
xchg al,ah
mov cl,4
shr al,cl

```

```
call tetr_to_hex
pop cx
ret
byte_to_hex endp
```

```
wrd_to_hex proc near
push bx
mov bh,ah
call byte_to_hex
mov [di],ah
dec di
mov [di],ah
dec di
mov al,bh
call byte_to_hex
mov [di],ah
dec di
mov [di],al
pop bx
ret
wrd_to_hex endp
```

```
byte_to_dec proc near
push cx
push dx
xor ah,ah
xor dx,dx
mov cx,10
loop_bd:
div cx
or dl,30h
mov [si],dl
dec si
xor dx,dx
cmp ax,10
```

```

    jae loop_bd
    cmp al,00h
    je end_l
    or al,30h
    mov [si],al
end_l:
    pop dx
    pop cx
    ret
byte_to_dec endp

wrd_to_byte proc near
    mov bx,10h
    mul bx
    mov bx,0ah
    xor cx,cx

divide:
    div bx
    push dx
    inc cx
    xor dx,dx
    cmp ax,0h
    jnz divide

print_symbol:
    pop dx
    or dl,30h
    mov [si],dl
    inc si
    loop print_symbol

    ret
wrd_to_byte endp

```

```
get_mcb_type proc near
mov di, offset table_data
add di, 5
xor ah, ah
mov al, es:[00h]
call byte_to_hex
mov [di], al
inc di
mov [di], ah
```

```
ret
get_mcb_type endp
```

```
get_psp_address proc near

mov di,offset table_data+19
mov ax,es:[01h]
call wrd_to_hex
```

```
ret
get_psp_address endp
```

```
get_mcb_size proc near
push bx
mov di,offset table_data+29
mov ax,es:[03h]
mov si,di
call wrd_to_byte
```

```
pop bx
ret
get_mcb_size endp
```

```
sc_sd proc near
mov di,offset table_data+37
```

```

    mov bx,0h

jmp_next:
    mov dl,es:[bx + 8]
    mov [di],dl
    inc di
    inc bx
    cmp bx,8h
    jne jmp_next

    ret
sc_sd endp

print proc near
    push ax
    mov ah,09h
    int 21h
    pop ax
    ret
print endp

begin_ptr:
    mov ah,04ah
    mov bx,0ffffh
    int 21h

    mov ax,bx

    mov si,offset avilable_memory+17
    call wrd_to_byte

    mov dx,offset avilable_memory
    call print

    mov al,30h

```

out 70h,al

in al,71h

mov bl,al

mov al,31h

out 70h,al

in al,71h

mov si,offset extended_memory+17

call wrd_to_byte

mov dx,offset extended_memory

call print

lea ax,end_ptr

mov bx,10h

xor dx,dx

div bx

inc ax

mov bx,ax

mov al,0

mov ah,4ah

int 21h

mov ah,52h

int 21h

mov es,es:[bx-2h]

mov dx,offset table

call print

print_mcb_data:

call get_mcb_type

call get_psp_address

call get_mcb_size

call sc_sd


```

mov ax,es:[03h]
    mov bl,es:[00h]
    mov dx,offset table_data
    call print

```

```

mov cx,es
    add ax,cx
    inc ax
    mov es,ax

```

```

cmp bl,4Dh
je print_mcb_data

```

```

xor al,al
mov ah,4ch
int 21h
end_ptr:
testpc ends
end start

```

Название файла: lab3_2.asm

```

testpc segment
assume cs:testpc,ds:testpc,es:nothing,ss:nothing
org 100h
start:
jmp begin_ptr

```

```

available_memory db 'available_memory:    bytes',0dh,0ah','$'
extended_memory db 'extended_memory:    Kbytes',0dh,0ah','$'
table db 'table:',0dh,0ah','$'
table_data db '                                ',0dh,0ah','$'

```

```

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 cx
mov ah,al
call tetr_to_hex
xchg al,ah
mov cl,4
shr al,cl
call tetr_to_hex
pop cx
ret
byte_to_hex endp
```

```
wrd_to_hex proc near
push bx
mov bh,ah
call byte_to_hex
mov [di],ah
dec di
mov [di],ah
dec di
mov al,bh
call byte_to_hex
mov [di],ah
dec di
mov [di],al
pop bx
ret
wrd_to_hex endp
```

```

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

```

```

wrd_to_byte proc near
mov bx,10h
mul bx
mov bx,0ah
xor cx,cx

```

```

divide:
div bx
push dx
inc cx

```

```
xor dx,dx
cmp ax,0h
jnz divide
```

```
print_symbol:
pop dx
or dl,30h
mov [si],dl
inc si
loop print_symbol
```

```
ret
wrd_to_byte endp
```

```
get_mcb_type proc near
mov di, offset table_data
add di, 5
xor ah, ah
mov al, es:[00h]
call byte_to_hex
mov [di], al
inc di
mov [di], ah
```

```
ret
get_mcb_type endp
```

```
get_psp_address proc near

mov di,offset table_data+19
mov ax,es:[01h]
call wrd_to_hex

ret
get_psp_address endp
```

```
get_mcb_size proc near
push bx
mov di,offset table_data+29
mov ax,es:[03h]
mov si,di
call wrd_to_byte
```

```
pop bx
ret
get_mcb_size endp
```

```
sc_sd proc near
mov di,offset table_data+37
    mov bx,0h
```

```
jmp_next:
    mov dl,es:[bx + 8]
mov [di],dl
inc di
inc bx
cmp bx,8h
jne jmp_next
```

```
ret
sc_sd endp
```

```
print proc near
push ax
mov ah,09h
int 21h
pop ax
ret
print endp
```

```

begin_ptr:
mov ah,04ah
mov bx,0ffffh
int 21h

mov ax,bx

mov si,offset available_memory+17
call wrd_to_byte

mov dx,offset available_memory
call print

mov al,30h
out 70h,al
in al,71h
mov bl,al
mov al,31h
out 70h,al
in al,71h

mov si,offset extended_memory+17
call wrd_to_byte

mov dx,offset extended_memory
call print

lea ax,end_ptr
mov bx,10h
xor dx,dx
div bx
inc ax
mov bx,ax
mov al,0
mov ah,4ah

```

int 21h

mov ah,48h

mov bx,1000h

int 21h

mov ah,52h

int 21h

mov es,es:[bx-2h]

mov dx,offset table

call print

print_mcb_data:

call get_mcb_type

call get_psp_address

call get_mcb_size

call sc_sd

mov ax,es:[03h]

mov bl,es:[00h]

mov dx,offset table_data

call print

mov cx,es

add ax,cx

inc ax

mov es,ax

cmp bl,4Dh

je print_mcb_data

xor al,al

mov ah,4ch

int 21h

end_ptr:

testpc ends

end start

Название файла: lab3_3.asm

testpc segment

assume cs:testpc,ds:testpc,es:nothing,ss:nothing

org 100h

start:

jmp begin_ptr

avilable_memory db 'avilable_memory: bytes',0dh,0ah,'\$'

extended_memory db 'extended_memory: Kbytes',0dh,0ah,'\$'

table db 'table:',0dh,0ah,'\$'

table_data db ' ',0dh,0ah,'\$'

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 cx

mov ah,al

call tetr_to_hex

xchg al,ah

mov cl,4

shr al,cl

call tetr_to_hex

pop cx

ret


```
byte_to_hex endp
```

```
wrd_to_hex proc near
```

```
push bx
```

```
mov bh,ah
```

```
call byte_to_hex
```

```
mov [di],ah
```

```
dec di
```

```
mov [di],ah
```

```
dec di
```

```
mov al,bh
```

```
call byte_to_hex
```

```
mov [di],ah
```

```
dec di
```

```
mov [di],al
```

```
pop bx
```

```
ret
```

```
wrd_to_hex endp
```

```
byte_to_dec proc near
```

```
push cx
```

```
push dx
```

```
xor ah,ah
```

```
xor dx,dx
```

```
mov cx,10
```

```
loop_bd:
```

```
div cx
```

```
or dl,30h
```

```
mov [si],dl
```

```
dec si
```

```
xor dx,dx
```

```
cmp ax,10
```

```
jae loop_bd
```

```
cmp al,00h
```

```
je end_1
```

```
or al,30h
mov [si],al
end_l:
pop dx
pop cx
ret
byte_to_dec endp
```

```
wrd_to_byte proc near
mov bx,10h
mul bx
mov bx,0ah
xor cx,cx
```

```
divide:
div bx
push dx
inc cx
xor dx,dx
cmp ax,0h
jnz divide
```

```
print_symbol:
pop dx
or dl,30h
mov [si],dl
inc si
loop print_symbol
```

```
ret
wrd_to_byte endp
```

```
get_mcb_type proc near
mov di, offset table_data
add di, 5
```

```

xor ah, ah
mov al, es:[00h]
call byte_to_hex
mov [di], al
inc di
mov [di], ah

ret
get_mcb_type endp

get_psp_address proc near

mov di, offset table_data+19
mov ax, es:[01h]
call wrd_to_hex

ret
get_psp_address endp

get_mcb_size proc near
push bx
mov di, offset table_data+29
mov ax, es:[03h]
mov si, di
call wrd_to_byte

pop bx
ret
get_mcb_size endp

sc_sd proc near
mov di, offset table_data+37
    mov bx, 0h

jmp_next:

```

```

    mov dl,es:[bx + 8]
mov [di],dl
inc di
inc bx
cmp bx,8h
jne jmp_next

ret
sc_sd endp

print proc near
push ax
mov ah,09h
int 21h
pop ax
ret
print endp

begin_ptr:
mov ah,04ah
mov bx,0ffffh
int 21h

mov ax,bx

mov si,offset avilable_memory+17
call wrd_to_byte

mov dx,offset avilable_memory
call print

mov al,30h
out 70h,al
in al,71h
mov bl,al

```

mov al,31h

out 70h,al

in al,71h

mov si,offset extended_memory+17

call wrd_to_byte

mov dx,offset extended_memory

call print

lea ax,end_ptr

mov bx,10h

xor dx,dx

div bx

inc ax

mov bx,ax

mov al,0

mov ah,4ah

int 21h

mov ah,48h

mov bx,1000h

int 21h

mov ah,52h

int 21h

mov es,es:[bx-2h]

mov dx,offset table

call print

print_mcb_data:

call get_mcb_type

call get_psp_address

call get_mcb_size

call sc_sd

```

mov ax,es:[03h]
    mov bl,es:[00h]
    mov dx,offset table_data
    call print

```

```

mov cx,es
    add ax,cx
    inc ax
    mov es,ax

```

```

cmp bl,4Dh
je print_mcb_data

```

```

xor al,al
mov ah,4ch
int 21h
end_ptr:
testpc ends
end start

```

Название файла: lab3_4.asm

```

testpc segment
assume cs:testpc,ds:testpc,es:nothing,ss:nothing
org 100h
start:
jmp begin

```

```

available_memory db 'available_memory:    bytes',0dh,0ah','$'
extended_memory db 'extended_memory:    Kbytes',0dh,0ah','$'
table db 'table:',0dh,0ah','$'
table_data db '                                ',0dh,0ah','$'
memory_allocate_error db 'memory_allocate error',0dh,0ah','$'

```

```

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 cx
mov ah,al
call tetr_to_hex
xchg al,ah
mov cl,4
shr al,cl
call tetr_to_hex
pop cx
ret
byte_to_hex endp
```

```
wrd_to_hex proc near
push bx
mov bh,ah
call byte_to_hex
mov [di],ah
dec di
mov [di],ah
dec di
mov al,bh
call byte_to_hex
mov [di],ah
dec di
mov [di],al
pop bx
```

```

ret
wrd_to_hex endp

byte_to_dec proc near
push cx
push dx
xor ah,ah
xor dx,dx
mov cx,10
loop_bd:
div cx
or dl,30h
mov [si],dl
dec si
xor dx,dx
cmp ax,10
jae loop_bd
cmp al,00h
je end_1
or al,30h
mov [si],al
end_1:
pop dx
pop cx
ret
byte_to_dec endp

wrd_to_byte proc near
mov bx,10h
mul bx
mov bx,0ah
xor cx,cx

divide:
div bx

```



```
push dx
inc cx
xor dx,dx
cmp ax,0h
jnz divide
```

```
print_symbol:
pop dx
or dl,30h
mov [si],dl
inc si
loop print_symbol
```

```
ret
wrd_to_byte endp
```

```
get_mcb_type proc near
mov di, offset table_data
add di, 5
xor ah, ah
mov al, es:[00h]
call byte_to_hex
mov [di], al
inc di
mov [di], ah
```

```
ret
get_mcb_type endp
```

```
get_psp_address proc near

mov di,offset table_data+19
mov ax,es:[01h]
call wrd_to_hex
```

```

ret
get_psp_address endp

get_mcb_size proc near
push bx
mov di,offset table_data+29
mov ax,es:[03h]
mov si,di
call wrd_to_byte

pop bx
ret
get_mcb_size endp

sc_sd proc near
mov di,offset table_data+37
    mov bx,0h

jmp_next:
    mov dl,es:[bx + 8]
mov [di],dl
inc di
inc bx
cmp bx,8h
jne jmp_next

ret
sc_sd endp

print proc near
push ax
mov ah,09h
int 21h
pop ax
ret

```

print endp

begin:

mov ah,04ah

mov bx,0ffffh

int 21h

mov ax,bx

mov si,offset avilable_memory+17

call wrd_to_byte

mov dx,offset avilable_memory

call print

mov al,30h

out 70h,al

in al,71h

mov bl,al

mov al,31h

out 70h,al

in al,71h

mov si,offset extended_memory+17

call wrd_to_byte

mov dx,offset extended_memory

call print

mov ah,48h

mov bx,1000h

int 21h

jnc allocated_memory

mov dx,offset memory_allocate_error

call print

allocated_memory:

mov ah,52h

int 21h

mov es,es:[bx-2h]

mov dx,offset table

call print

print_mcb_data:

call get_mcb_type

call get_psp_address

call get_mcb_size

call sc_sd

mov ax,es:[03h]

mov bl,es:[00h]

mov dx,offset table_data

call print

mov cx,es

add ax,cx

inc ax

mov es,ax

cmp bl,4Dh

je print_mcb_data

xor al,al

mov ah,4ch

int 21h

testpc ends

end start