

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

Студентка гр. 7383

Иолшина В.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2019

Цель работы.

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

Таблица 1 - Описание процедур и структур данных.

Название процедуры	Назначение
<code>_AVAILABLEMEMORY</code>	определяет размер доступной памяти
<code>_EXTENDEDMEMORY</code>	определяет размер расширенной памяти
<code>_DATA</code>	определяет цепочку блоков управления памятью
<code>OUTPUT</code>	выводит на экран цепочку блоков управления памятью
<code>BYTE_TO_HEX</code>	переводит байтовое число из регистра AL в шестнадцатеричную систему счисления, записывая получившееся в al и ah
<code>TETR_TO_HEX</code>	вспомогательная функция для работы функции <code>BYTE_TO_HEX</code> , которая переводит из двоичной в шестнадцатеричную систему счисления
<code>WRD_TO_HEX</code>	переводит число из регистра AX в строку в шестнадцатеричной системе счисления с, записывая получившееся в di
<code>BYTE_TO_DEC</code>	переводит байт из регистра AL в десятичную систему счисления и записывает получившееся число по адресу si, начиная с младшей цифры
<code>_TO_DEC</code>	переводит два байта в десятичную систему счисления
<code>PRINT</code>	вызывает функцию печати строки

Действия, выполняемые программой.

1. Вывод количества доступной памяти;
2. Вывод размера расширенной памяти;
3. Вывод цепочки блоков управления памятью.

Написаны три модификации для программы:

1. Программа освобождает память, которую не использует.
2. Программа так же освобождает память. После освобождения памяти запрашивает 64Кб памяти.
3. Программа освобождает память. До освобождения памяти запрашивает 64Кб памяти. Обработывает завершение функций ядра

Результат работы программы представлен на рис. 1-4.

```
K:\>lr3_1
Amount of available memory: 648912 b
Extended memory size: 15360 kB
Address      Type      Owner      Size      Name
016F         4D        0008        16
0171         4D        0000        64      DPMILOAD
0176         4D        0040       256
0187         4D        0192       144
0191         5A        0192    648912      LR3_1
```

Рисунок 1 – Результат выполнения программы 3_1.com

```
K:\>lr3_2
Amount of available memory: 648912 b
Extended memory size: 15360 kB
Address      Type      Owner      Size      Name
016F         4D        0008        16
0171         4D        0000        64      DPMILOAD
0176         4D        0040       256
0187         4D        0192       144
0191         4D        0192    13264      LR3_2
04CF         5A        0000    635632
```

Рисунок 2 – Результат выполнения программы 3_2.com

```
K:\>lr3_3
Amount of available memory: 648912 b
Extended memory size: 15360 kB
```

Address	Type	Owner	Size	Name
016F	4D	0008	16	
0171	4D	0000	64	DPMILOAD
0176	4D	0040	256	
0187	4D	0192	144	
0191	4D	0192	13424	LR3_3
04D9	4D	0192	65536	LR3_3
14DA	5A	0000	569920	↑ î ¸ú

Рисунок 3 – Результат выполнения программы 3_3.com

```
K:\>lr3_4
Amount of available memory: 648912 b
Extended memory size: 15360 kB
Error!
```

Address	Type	Owner	Size	Name
016F	4D	0008	16	
0171	4D	0000	64	DPMILOAD
0176	4D	0040	256	
0187	4D	0192	144	
0191	4D	0192	13792	LR3_4
04F0	5A	0000	635104	

Рисунок 4 – Результат выполнения программы 3_4.com

Выводы.

В процессе выполнения данной лабораторной работы были исследованы структуры данных и работа функций управления памятью ядра операционной системы. Был написан текст исходного .COM файла, который выводит на экран количество доступной памяти, размер расширенной памяти и цепочку блоков управления памятью.

Ответы на контрольные вопросы.

1) Что означает «доступный объем памяти»?

Доступный объём памяти – максимальный объём памяти, который может быть использован для загрузки программ.

2) Где MCB блок вашей программы в списке?

В программах lr3_1.com, lr3_2.com и lr3_4.com MCB блок имеет адрес 0191h, а в программе lr3_3.com присутствует два MCB блока, первый из них имеет адрес 0191h, второй – 04D9h. В каждой программе присутствует блок MCB

размером 144б, который имеет адрес 0187h. Данный блок служит для управления памятью для области среды программы.

3) Какой размер памяти занимает программа в каждом случае?

В первом случае программа занимает 648912 байт.

Во втором случае: $648912 - 635632 - 16 = 13264$ байт.

В третьем случае: $648912 - 569920 - 65536 - 32 = 13424$ байт.

В четвёртом случае: $648912 - 635104 - 16 = 13792$ байт.

ПРИЛОЖЕНИЕ А

3_1.ASM

TESTPC SEGMENT

ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
ORG 100H

START: jmp BEGIN

;Данные

AVAILABLEMEMORY db ' Amount of available memory: b',0dh,0ah','\$'

EXTENDEDMEMORY db ' Extended memory size: kB',0dh,0ah','\$'

HEAD db ' Adress Type Owner Size Name ', 0dh, 0ah, '\$'

DATA 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

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],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
;-----
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
;-----
_TO_DEC         PROC    near
        push    cx
        push    dx
        push    ax
        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     ax,00h
        jbe     _end_l
        or      al,30h

```

```

                                mov             [si],al
_end_1:

                                pop             ax
                                pop             dx
                                pop             cx
                                ret

_TO_DEC                        ENDP
;-----
PRINT PROC NEAR
                                push            ax
                                mov             ah, 09h
                                int             21h
                                pop             ax
                                ret
PRINT ENDP
;-----
_AVAILABLEMEMORY PROC NEAR ; Поиск доступной памяти
                                push            ax
                                push            bx
                                push            dx
                                push            si

                                xor             ax, ax
                                mov             ah, 04Ah
                                mov             bx, 0FFFFh
                                int             21h
                                mov             ax, 10h
                                mul             bx

                                mov             si, offset AVAILABLEMEMORY
                                add             si, 23h
                                call            _TO_DEC

                                pop             si
                                pop             dx
                                pop             bx
                                pop             ax
                                ret
_AVAILABLEMEMORY ENDP
;-----
_EXTENDEDMEMORY PROC    near ; Поиск расширенной памяти
                                push            ax
                                push            bx
                                push            si
                                push            dx

```



```

        mov     al, 30h
        out     70h, al
        in      al, 71h
        mov     bl, al
        mov     al, 31h
        out     70h, al
        in      al, 71h
        mov     ah, al
        mov     al, bl
        sub     dx, dx

        mov     si, offset EXTENDEDMEMORY
        add     si, 28
        call    _TO_DEC

        pop     dx
        pop     si
        pop     bx
        pop     ax
        ret

_EXTENDEDMEMORY ENDP
;-----
_DATA PROC near ; Поиск MCB

        mov     di, offset DATA ; Адрес MCB
        mov     ax, es
        add     di, 05h
        call    WRD_TO_HEX ; Тип MCB
        add     di, 0Fh
        xor     ah, ah
        mov     al, es:[00h]
        call    BYTE_TO_HEX
        mov     [di], al
        inc     di
        mov     [di], ah

        mov     di, offset DATA
        mov     ax, es:[01h]
        add     di, 1Dh
        call    WRD_TO_HEX

        mov     di, offset DATA
        mov     ax, es:[03h]
        mov     bx, 10h
        mul     bx
        add     di, 2Eh
        push    si

```

```

        mov     si, di
        call    _TO_DEC
        pop     si

        mov     di, offset DATA
        add     di, 35h
        mov     bx, 0h
print_:
                mov dl, es:[bx + 8]
                mov [di], dl
                inc di
                inc bx
                cmp bx, 8h

        jne     print_
        mov     ax, es:[3h]
        mov     bl, es:[0h]
        ret

_DATA ENDP
;-----
OUTPUT PROC NEAR ; Поиск цепочки блоков управления памятью

        mov     ah, 52h
        int     21h
        sub     bx, 2h
        mov     es, es:[bx]
output_:
        call    _DATA
        mov     dx, offset DATA
        call    PRINT
        mov     cx, es
        add     ax, cx
        inc     ax
        mov     es, ax
        cmp     bl, 4Dh
        je      output_

        ret

OUTPUT ENDP
;-----
BEGIN:

        call    _AVAILABLEMEMORY
        mov     dx, offset AVAILABLEMEMORY
        call    PRINT

        call    _EXTENDEDMEMORY
        mov     dx, offset EXTENDEDMEMORY
        call    PRINT

```

```
lea    dx, HEAD
call   PRINT
call   OUTPUT

xor     al, al
mov     ah, 4ch
int     21h
```

TESTPC ENDS

END START

ПРИЛОЖЕНИЕ Б

3_2.ASM

TESTPC SEGMENT

ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
ORG 100H

START: jmp BEGIN

;data

AVAILABLEMEMORY db ' Amount of available memory: b',0dh,0ah,'\$'

EXTENDEDMEMORY db ' Extended memory size: kB',0dh,0ah,'\$'

HEAD db ' Adress Type Owner Size Name ', 0dh, 0ah, '\$'

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

;procedurs

;-----

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],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
;-----
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
;-----
_TO_DEC         PROC    near
        push    cx
        push    dx
        push    ax
        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     ax,00h
        jbe     _end_1

```

```

        or            al,30h
        mov           [si],al
_end_1:

        pop           ax
        pop           dx
        pop           cx
        ret

_TO_DEC      ENDP
;-----
PRINT PROC NEAR
        push          ax
        mov           ah, 09h
        int           21h
        pop           ax
        ret
PRINT ENDP
;-----
_AVAILABLEMEMORY PROC NEAR ; Поиск доступной памяти

        push          ax
        push          bx
        push          dx
        push          si

        xor           ax, ax
        mov           ah, 04Ah
        mov           bx, 0FFFFh
        int           21h
        mov           ax, 10h
        mul           bx

        mov           si, offset AVAILABLEMEMORY
        add           si, 23h
        call          _TO_DEC

        pop           si
        pop           dx
        pop           bx
        pop           ax
        ret
_AVAILABLEMEMORY ENDP
;-----
_EXTENDEDMEMORY PROC    near ; Поиск расширенной памяти

        push          ax
        push          bx
        push          si
        push          dx

```

```

mov     al, 30h
out     70h, al
in      al, 71h
mov     bl, al
mov     al, 31h
out     70h, al
in      al, 71h
mov     ah, al
mov     al, bl
sub     dx, dx

mov     si, offset EXTENDEDMEMORY
add     si, 28
call    _TO_DEC

pop     dx
pop     si
pop     bx
pop     ax
ret

```

_EXTENDEDMEMORY ENDP

;-----

_DATA PROC near ; Поиск MCB

```

mov     di, offset DATA ; Адрес MCB
mov     ax, es
add     di, 05h
call    WRD_TO_HEX

```

```

mov     di, offset DATA ; Тип MCB
add     di, 0Fh
xor     ah, ah
mov     al, es:[00h]
call    BYTE_TO_HEX
mov     [di], al
inc     di
mov     [di], ah

```

```

mov     di, offset DATA
mov     ax, es:[01h]
add     di, 1Dh
call    WRD_TO_HEX

```

```

mov     di, offset DATA
mov     ax, es:[03h]
mov     bx, 10h

```

```

mul    bx
add    di, 2Eh
push   si
mov     si, di
call    _TO_DEC
pop     si

mov     di, offset DATA
add     di, 35h
mov     bx, 0h
print_:
        mov dl, es:[bx + 8]
        mov [di], dl
        inc di
        inc bx
        cmp bx, 8h

jne     print_
mov     ax, es:[3h]
mov     bl, es:[0h]
ret

_DATA ENDP
;-----
OUTPUT PROC NEAR ; Поиск цепочки блоков управления памятью

mov     ah, 52h
int     21h
sub     bx, 2h
mov     es, es:[bx]
output_:
        call    _DATA
        mov     dx, offset DATA
        call    PRINT
        mov     cx, es
        add     ax, cx
        inc     ax
        mov     es, ax
        cmp     bl, 4Dh
        je      output_
ret

OUTPUT ENDP
;-----
BEGIN:

call    _AVAILABLEMEMORY
mov     dx, offset AVAILABLEMEMORY
call    PRINT

call    _EXTENDEDMEMORY

```



```

mov          dx, offset EXTENDEDMEMORY
call         PRINT

mov         ah, 4ah    ;Освобождение памяти
mov         bx, offset END_PROG
int         21h

mov         dx, offset HEAD
call        PRINT
call        OUTPUT

xor         al, al
mov         ah, 4ch
int         21h

END_PROG db 0

```

TESTPC ENDS

END START

ПРИЛОЖЕНИЕ В

3_3.ASM

```

TESTPC SEGMENT
        ASSUME     CS:TESTPC,    DS:TESTPC,    ES:NOTHING,    SS:NOTHING
        ORG        100H

START: jmp          BEGIN

;data
AVAILABLEMEMORY db ' Amount of available memory:      b',0dh,0ah,'$'
EXTENDEDMEMORY  db ' Extended memory size:      kB',0dh,0ah,'$'
HEAD            db ' Address      Type      Owner      Size      Name      ',0dh,0ah,'$'
DATA            db '                                     ',0dh,0ah,'$'

;procedurs
;-----
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],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
;-----
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
end_l:    pop        [si],al
                                dx
                                pop
                                ret        cx
BYTE_TO_DEC                                ENDP
;-----
_TO_DEC                                PROC    near
                                push    cx
                                push    dx
                                push    ax
                                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     ax,00h
                                jbe     _end_l
                                or      al,30h
                                mov     [si],al
_end_l:
                                pop     ax
                                pop     dx
                                pop     cx
                                ret
_TO_DEC                                ENDP
;-----
PRINT PROC NEAR
                                push    ax
                                mov     ah, 09h
                                int     21h
                                pop     ax
                                ret
PRINT ENDP
;-----
_AVAILABLEMEMORY PROC NEAR ; Поиск доступной памяти
                                push    ax
                                push    bx
                                push    dx
                                push    si

                                xor     ax, ax
                                mov     ah, 04Ah
                                mov     bx, 0FFFFh
                                int     21h
                                mov     ax, 10h
                                mul     bx

                                mov     si, offset AVAILABLEMEMORY
                                add     si, 23h
                                call    _TO_DEC

                                pop     si
                                pop     dx
                                pop     bx
                                pop     ax
                                ret
_AVAILABLEMEMORY ENDP
;-----
_EXTENDEDMEMORY PROC    near ; Поиск расширенной памяти
                                push    ax
                                push    bx
                                push    si

```

```

        push    dx

        mov     al, 30h
        out     70h, al
        in      al, 71h
        mov     bl, al
        mov     al, 31h
        out     70h, al
        in      al, 71h
        mov     ah, al
        mov     al, bl
        sub     dx, dx

        mov     si, offset EXTENDEDMEMORY
        add     si, 28
        call    _TO_DEC

        pop     dx
        pop     si
        pop     bx
        pop     ax
        ret

_EXTENDEDMEMORY ENDP
;-----
_DATA PROC near ; Поиск MCB
        mov     di, offset DATA ; Адрес MCB
        mov     ax, es
        add     di, 05h
        call    WRD_TO_HEX

        mov     di, offset DATA ; Тип MCB
        add     di, 0Fh
        xor     ah, ah
        mov     al, es:[00h]
        call    BYTE_TO_HEX
        mov     [di], al
        inc     di
        mov     [di], ah

        mov     di, offset DATA
        mov     ax, es:[01h]
        add     di, 1Dh
        call    WRD_TO_HEX

        mov     di, offset DATA
        mov     ax, es:[03h]
        mov     bx, 10h
        mul     bx
        add     di, 2Eh
        push    si
        mov     si, di
        call    _TO_DEC
        pop     si

        mov     di, offset DATA
        add     di, 35h
        mov     bx, 0h
        print_:
                mov dl, es:[bx + 8]
                mov [di], dl
                inc di
                inc bx
                cmp bx, 8h

        jne     print_
        mov     ax, es:[3h]
        mov     bl, es:[0h]
        ret

_DATA ENDP
;-----

```

OUTPUT PROC NEAR ; Поиск цепочки блоков управления памятью

```
mov     ah, 52h
int     21h
sub     bx, 2h
mov     es, es:[bx]
output_:
        call    _DATA
        mov     dx, offset DATA
        call    PRINT
        mov     cx, es
        add     ax, cx
        inc     ax
        mov     es, ax
        cmp     bl, 4Dh
        je      output_
ret
```

OUTPUT ENDP

;-----

BEGIN:

```
call    _AVAILABLEMEMORY
mov     dx, offset AVAILABLEMEMORY
call    PRINT

call    _EXTENDEDMEMORY
mov     dx, offset EXTENDEDMEMORY
call    PRINT

mov     ah, 4ah ; Освобождение памяти
mov     bx, offset END_PROG
int     21h

mov     ah, 48h ; Запрос 64 KB памяти
mov     bx, 1000h
int     21h

mov     dx, offset HEAD
call    PRINT
call    OUTPUT

xor     al, al
mov     ah, 4ch
int     21h

END_PROG db 0
```

TESTPC ENDS

END START

ПРИЛОЖЕНИЕ Г

3_4.ASM

TESTPC SEGMENT

ASSUME CS:TESTPC, DS:TESTPC, ES:NOTHING, SS:NOTHING
ORG 100H

START: jmp BEGIN

;data

AVAILABLEMEMORY db ' Amount of available memory: b',0dh,0ah,'\$'

EXTENDEDMEMORY db ' Extended memory size: kB',0dh,0ah,'\$'

HEAD db ' Adress Type Owner Size Name ', 0dh, 0ah, '\$'

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

ERRORM db ' Error!', 0dh, 0ah, '\$'

;procedurs

;-----

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],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
;-----
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
;-----
_TO_DEC         PROC    near
        push    cx
        push    dx
        push    ax
        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     ax,00h

```

```

        jbe      _end_1
        or       al,30h
        mov      [si],al
_end_1:

        pop      ax
        pop      dx
        pop      cx
        ret

_TO_DEC      ENDP
;-----
PRINT PROC NEAR

        push     ax
        mov      ah, 09h

        int      21h
        pop      ax
        ret

PRINT ENDP
;-----
_AVAILABLEMEMORY PROC NEAR ; Поиск доступной памяти

        push     ax
        push     bx
        push     dx
        push     si

        xor      ax, ax
        mov      ah, 04Ah
        mov      bx, 0FFFFh
        int      21h
        mov      ax, 10h
        mul      bx

        mov      si, offset AVAILABLEMEMORY
        add      si, 23h
        call     _TO_DEC

        pop      si
        pop      dx
        pop      bx
        pop      ax
        ret

_AVAILABLEMEMORY ENDP
;-----
_EXTENDEDMEMORY PROC near ; Поиск расширенной памяти

        push     ax
        push     bx
        push     si

```



```

push    dx

mov     al, 30h
out     70h, al
in      al, 71h
mov     bl, al
mov     al, 31h
out     70h, al
in      al, 71h
mov     ah, al
mov     al, bl
sub     dx, dx

mov     si, offset EXTENDEDMEMORY
add     si, 28
call    _TO_DEC

pop     dx
pop     si
pop     bx
pop     ax
ret

_EXTENDEDMEMORY ENDP
;-----
_DATA PROC near ; Поиск MCB

mov     di, offset DATA ; Адрес MCB
mov     ax, es
add     di, 05h
call    WRD_TO_HEX

mov     di, offset DATA ; Тип MCB
add     di, 0Fh
xor     ah, ah
mov     al, es:[00h]
call    BYTE_TO_HEX
mov     [di], al
inc     di
mov     [di], ah

mov     di, offset DATA
mov     ax, es:[01h]
add     di, 1Dh
call    WRD_TO_HEX

mov     di, offset DATA
mov     ax, es:[03h]

```

```

        mov     bx, 10h
        mul     bx
        add     di, 2Eh
        push    si
        mov     si, di
        call    _TO_DEC
        pop     si

        mov     di, offset DATA
        add     di, 35h
        mov     bx, 0h
print_:
                mov dl, es:[bx + 8]
                mov [di], dl
                inc di
                inc bx
                cmp bx, 8h
        jne     print_
        mov     ax, es:[3h]
        mov     bl, es:[0h]
        ret

_DATA ENDP
;-----
OUTPUT PROC NEAR ; Поиск цепочки блоков управления памятью
        mov     ah, 52h
        int     21h
        sub     bx, 2h
        mov     es, es:[bx]
output_:
        call    _DATA
        mov     dx, offset DATA
        call    PRINT
        mov     cx, es
        add     ax, cx
        inc     ax
        mov     es, ax
        cmp     bl, 4Dh
        je      output_
        ret

OUTPUT ENDP
;-----
BEGIN:
        call    _AVAILABLEMEMORY
        mov     dx, offset AVAILABLEMEMORY
        call    PRINT

```

```

call    _EXTENDEDMEMORY
mov     dx, offset EXTENDEDMEMORY
call    PRINT

```

```

mov     ah, 48h ; Запрос 64 KB памяти
mov     bx, 1000h
int     21h

```

```

jc      memoryErr ; Проверка на ошибку
jmp     next_

```

```
memoryErr:
```

```

mov     dx, offset ErrorM
call    PRINT

```

```
next_ : ; Освобождение памяти
```

```

mov     ah, 4ah
mov     bx, offset PROGRAMM_ENDS
int     21h

```

```

mov     dx, offset HEAD
call    PRINT
call    OUTPUT

```

```

xor     al, al
mov     ah, 4ch
int     21h

```

```
PROGRAMM_ENDS db 0
```

```
TESTPC ENDS
```

```
END START
```