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		
_AVAILABLEMEMORY	определяет размер доступной памяти		
_EXTENDEDMEMORY	определяет размер расширенной памяти		
_DATA	определяет цепочку блоков управления		
	памятью		
OUTPUT	выводит на экран цепочку блоков управления		
	памятью		
BYTE_TO_HEX	переводит байтовое число из регистра AL в		
	шестнадцатеричную систему счисления,		
	записывая получившееся в al и ah		
TETR_TO_HEX	вспомогательная функция для работы функции		
	ВҮТЕ_ТО_НЕХ, которая переводит из		
	двоичной в шестнадцатеричную систему		
	счисления		
WRD_TO_HEX	переводит число из регистра АХ в строку в		
	шестнадцатеричной системе счисления с,		
	записывая получившееся в di		
BYTE_TO_DEC	переводит байт из регистра AL в десятичную		
	систему счисления и записывает получившееся		
	число по адресу si, начиная с младшей цифры		
_TO_DEC	переводит два байта в десятеричную систему		
	счисления		
PRINT	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. Обрабатывает 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
             Туре
  Adress
                        0wner
                                            Size
                                                         Name
  016F
                  4D
                            0008
                                                16
                            0000
                                                         DPMILOAD
  0171
                  4D
                                                64
  0176
                  4D
                            0040
                                               256
                  4D
                            0192
  0187
                                               144
  0191
                  5A
                            0192
                                            648912
                                                         LR3_1
```

Рисунок 1 – Результат выполнения программы 3_1.com

```
K: \times 1r3_2
 Amount of available memory: 648912 b
  Extended memory size: 15360 kB
                                              Size
 Adress
               Туре
                           0wner
                                                            Name
  016F
                4D
                             0008
                                                   16
  0171
                4D
                             0000
                                                   64
                                                            DPMILOAD
  0176
                4D
                             0040
                                                  256
  0187
                4D
                             0192
                                                  144
  0191
                4D
                             0192
                                                            LR3<sub>2</sub>
                                                13264
  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			
	Adress	Туре	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 ^լ &ú∟	

Рисунок 3 – Результат выполнения программы 3_3.com

K:\>lr3_4							
Amount of available memory: 648912 b							
Extended memory size: 15360 kB							
Error!							
Adress	Туре	Owner	Size	Name			
016F	4D	0008	16				
0171	4D	0000	64	DPMILOAD			
0176	4D	0040	256				
0187	4D	0192	14 4				
0191	4D	0192	13792	LR3_4			
04F0	5A	0000	63510 4				
	*						

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 .СОМ файла, который выводит на экран количество доступной памяти, размер расширенной памяти и цепочку блоков управления памяти.

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 МСВ 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 1446, 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

add al,0°

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_HE	EX
	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	d1,30	h
	mov	[si],dl	
	dec	si	
	xor	dx,dx	
	cmp	ax,10	
	jae	loop_bd	
	cmp	al,001	h
	je	end_l	l
	or	al,30l	h
	mov	[si],al	
end_l: pop	dx		
	pop	cx	
	ret		
BYTE_TO_DEC		ENDP	
;		PD CC	
_TO_DEC	nuch	PROC near	
	push push	cx dx	
	push		
	mov	ax cx,10	,
_loop_bd:	IIIOV	CX,10	,
	div	cx	
	or	d1,30	h
	mov	[si],dl	
	dec	si	
	xor	dx,dx	(
	cmp	ax,10)
	jae	_loop	_bd
	cmp	ax,00	h
	jbe	_end	_l

or

al,30h

```
mov
                               [si],al
_end_l:
                               ax
               pop
                               dx
               pop
               pop
                               cx
               ret
_TO_DEC
                       ENDP
;-----
PRINT PROC NEAR
               push
                       ah, 09h
               mov
         int
                21h
                       ax
         pop
         ret
PRINT ENDP
_AVAILABLEMEMORY PROC NEAR ; Поиск доступной памяти
               push
               push
                       bx
               push
                       dx
               push
                        \dot{si}
                        ax, ax
               xor
                       ah, 04Ah
               mov
                        bx, 0FFFFh
               mov
                       21h
               int
                        ax, 10h
               mov
                       bx
               mul
                       si, offset AVAILABLEMEMORY
               mov
                       si, 23h
               add
                        _TO_DEC
               call
                       si
               pop
                       dx
               pop
               pop
                       bx
               pop
                       ax
               ret
_AVAILABLEMEMORY ENDP
_EXTENDEDMEMORY PROC
                            near; Поиск расширенной памяти
               push
                       ax
               push
                       bx
               push
                       si
```

push

dx

```
al, 30h
                mov
                                 70h, al
                out
                                 al, 71h
                in
                                 bl, al
                mov
                                 al, 31h
                mov
                                 70h, al
                out
                                 al, 71h
                in
                         ah, al
                mov
                         al, bl
                mov
                         dx, dx
                sub
                         si, offset EXTENDEDMEMORY
                mov
                add
                         si, 28
                         _TO_DEC
                call
                pop
                                 dx
                pop
                                 si
                pop
                                 bx
                pop
                                 ax
                ret
_EXTENDEDMEMORY ENDP
;-----
_DATA PROC near; Поиск МСВ
                         di, offset DATA ; Адрес МСВ
                mov
                mov
                         ax, es
                add
                         di, 05h
                call
                         WRD_TO_HEX ; Тип МСВ
                         di, 0Fh
                add
                xor
                         ah, ah
                         al, es:[00h]
                mov
                         BYTE_TO_HEX
                call
                         [di], al
                mov
                inc
                         di
                         [di], ah
                mov
                         di, offset DATA
                mov
                         ax, es:[01h]
                mov
                         di, 1Dh
                add
                         WRD_TO_HEX
                call
                         di, offset DATA
                mov
                         ax, es:[03h]
                mov
                         bx, 10h
                mov
                         bx
                mul
```

add

push

di, 2Eh

si

```
mov
                         si, di
                         _TO_DEC
                call
                         si
                pop
                         di, offset DATA
                mov
                add
                         di, 35h
                         bx, 0h
                mov
                print_:
                                  mov dl, es:[bx + 8]
                                  mov [di], dl
                                  inc di
                                  inc bx
                                  cmp bx, 8h
                         print_
                jne
                         ax, es:[3h]
                mov
                         bl, es:[0h]
                mov
                ret
_DATA ENDP
OUTPUT PROC NEAR ; Поиск цепочки блоков управления памятью
                         ah, 52h
                mov
                         21h
                int
                         bx, 2h
                sub
                mov
                         es, es:[bx]
                output_:
                         call
                                  _DATA
                                  dx, offset DATA
                         mov
                                  PRINT
                         call
                         mov
                                  cx, es
                         add
                                  ax, cx
                         inc
                                  ax
                         mov
                                  es, ax
                         cmp
                                  bl, 4Dh
                         je
                                          output_
                ret
OUTPUT ENDP
BEGIN:
                         _AVAILABLEMEMORY
                call
                                  dx, offset AVAILABLEMEMORY
                mov
                call
                         PRINT
                         _EXTENDEDMEMORY
                call
                                  dx, offset EXTENDEDMEMORY
                mov
                         PRINT
                call
```

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

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

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 c

ret

BYTE_TO_HEX ENDP

;-----

WRD_TO_HEX PROC near

push bx

mov bh,ah

call BYTE_TO_HEX

mov [di],ah

di

dec

mov [di],al

dec di

```
al,bh
                mov
                        BYTE_TO_HEX
                call
                        [di],ah
                mov
                        di
                dec
                mov
                        [di],al
                        bx
                pop
                ret
                        ENDP
WRD_TO_HEX
;-----
BYTE_TO_DEC
                        PROC
                                near
                push
                        cx
                        dx
                push
                xor
                        ah,ah
                        dx,dx
                xor
                        cx,10
                mov
loop_bd:div
                cx
                                 d1,30h
                or
                        [si],dl
                mov
                        si
                dec
                xor
                          dx,dx
                        ax,10
                cmp
                jae
                        loop_bd
                                 al,00h
                cmp
                                 end\_l
                je
                                 al,30h
                or
                        [si],al
                mov
end_l:
                dx
       pop
                pop
                                 cx
                ret
BYTE_TO_DEC
                        ENDP
_TO_DEC
                        PROC
                                near
                push
                        cx
                push
                        dx
                push
                        ax
                                 cx,10
                mov
_loop_bd:
                div
                                 cx
                                 d1,30h
                or
                mov
                        [si],dl
                dec
                        si
                                 dx,dx
                xor
                                 ax,10
                cmp
                jae
                                 _loop_bd
                                 ax,00h
                cmp
                                 _end_l
```

jbe

```
al,30h
              or
                              [si],al
              mov
_end_l:
                              ax
              pop
                              dx
              pop
                              cx
              pop
              ret
                      ENDP
_TO_DEC
;-----
PRINT PROC NEAR
              push
                      ax
                      ah, 09h
              mov
               21h
         int
         pop
                      ax
         ret
PRINT ENDP
;-----
_AVAILABLEMEMORY PROC NEAR; Поиск доступной памяти
              push
              push
                      bx
                      dx
              push
                      si
              push
              xor
                      ax, ax
                      ah, 04Ah
              mov
                      bx, 0FFFFh
              mov
                      21h
              int
                      ax, 10h
              mov
              mul
                      bx
                      si, offset AVAILABLEMEMORY
              mov
                      si, 23h
              add
              call
                      _TO_DEC
                      si
              pop
              pop
                      dx
                      bx
              pop
              pop
              ret
_AVAILABLEMEMORY ENDP
;-----
_EXTENDEDMEMORY PROC near; Поиск расширенной памяти
              push
                      ax
              push
                      bx
              push
                      si
```

push

dx

```
al, 30h
                mov
                                  70h, al
                out
                                  al, 71h
                in
                                  bl, al
                mov
                                  al, 31h
                mov
                                  70h, al
                out
                                  al, 71h
                in
                         ah, al
                mov
                          al, bl
                mov
                         dx, dx
                sub
                         si, offset EXTENDEDMEMORY
                mov
                          si, 28
                add
                          _TO_DEC
                call
                pop
                                  dx
                pop
                                  si
                                  bx
                pop
                                  ax
                pop
                ret
_EXTENDEDMEMORY ENDP
_DATA PROC near; Поиск МСВ
                mov
                          di, offset DATA; Адрес МСВ
                         ax, es
                mov
                         di, 05h
                add
                          WRD_TO_HEX
                call
                         di, offset DATA; Тип МСВ
                mov
                          di, 0Fh
                add
                         ah, ah
                xor
                mov
                         al, es:[00h]
                         BYTE_TO_HEX
                call
                         [di], al
                mov
                         di
                inc
                         [di], ah
                mov
                         di, offset DATA
                mov
                          ax, es:[01h]
                mov
                add
                         di, 1Dh
                          WRD_TO_HEX
                call
```

di, offset DATA

ax, es:[03h]

bx, 10h

mov

mov

mov

15

```
mul
                         bx
                add
                         di, 2Eh
                push
                         si
                mov
                         si, di
                         _TO_DEC
                call
                         si
                pop
                         di, offset DATA
                mov
                add
                         di, 35h
                         bx, 0h
                mov
                print_:
                                 mov dl, es:[bx + 8]
                                 mov [di], dl
                                 inc di
                                 inc bx
                                 cmp bx, 8h
                jne
                         print_
                mov
                         ax, es:[3h]
                         bl, es:[0h]
                mov
                ret
_DATA ENDP
;-----
OUTPUT PROC NEAR ; Поиск цепочки блоков управления памятью
                         ah, 52h
                mov
                         21h
                int
                         bx, 2h
                sub
                         es, es:[bx]
                mov
                output_:
                        call
                                 _DATA
                                 dx, offset DATA
                        mov
                                 PRINT
                        call
                        mov
                                 cx, es
                        add
                                 ax, cx
                        inc
                                 ax
                        mov
                                 es, ax
                                 bl, 4Dh
                        cmp
                                         output_
                        je
                ret
OUTPUT ENDP
BEGIN:
                         _AVAILABLEMEMORY
                call
                                 dx, offset AVAILABLEMEMORY
                mov
                         PRINT
                call
                         _EXTENDEDMEMORY
                call
```

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

 ${\sf END_PROG}\ {\sf db}\ 0$

TESTPC ENDS

END START

приложение в

3_3.ASM

TESTPC SEGME	NT ASSUM ORG	Œ	CS:TESTPC, 100H	DS:TESTPC,	ES:NOTHING,	SS:NOTHING
START: jmp		BEGIN				
;data AVAILABLEME EXTENDEDMEN HEAD DATA			ount of available rended memory sizedb' Adress db'			', 0dh, 0ah, '\$'
;procedurs						
;TETR_TO_HEX	and cmp jbe add al,30h ret	PROC al,0fh al,09 NEXT al,07	near			
TETR_TO_HEX;		ENDP				
BYTE_TO_HEX	push mov call xchg mov shr call pop	al,ah cl,4 al,cl	ear TO_HEX TO_HEX			
BYTE_TO_HEX	ret	ENDP				
;WRD_TO_HEX	push mov call mov dec mov dec mov call mov dec mov pop	[di],ah di [di],al di al,bh	near TO_HEX TO_HEX			
; BYTE_TO_DEC		PROC	near			
B116_10_DEC	push push xor xor mov	cx dx ah,ah dx,dx cx,10	near			
loop_bd:div	cx or mov dec xor cmp jae	[si],dl si dx,dx ax,10 loop_bd		10		

```
cmp
                                 al,00h
                                 end_l
                je
                                 al,30h
                or
                         [si],al
                mov
end_l:
        pop
                dx
                                 cx
                pop
                ret
                        ENDP
BYTE_TO_DEC
                        PROC
_TO_DEC
                                 near
                push
                         cx
                         dx
                push
                push
                         ax
                mov
                                 cx,10
_loop_bd:
                div
                                 cx
                                 d1,30h
                or
                         [si],dl
                mov
                dec
                xor
                                 dx,dx
                                 ax,10
                cmp
                                 _loop_bd
                jae
                                 ax,00h
                cmp
                jbe
                                 _end_l
                                 al,30h
                or
                                 [si],al
                mov
_end_l:
                pop
                                 ax
                                 dx
                pop
                                 cx
                pop
                ret
_TO_DEC
                        ENDP
PRINT PROC NEAR
                         ax
                push
                mov
                         ah, 09h
          int
                21h
          pop
                         ax
          ret
PRINT ENDP
_AVAILABLEMEMORY PROC NEAR; Поиск доступной памяти
                push
                push
                         bx
                push
                         dx
                push
                         si
                xor
                         ax, ax
                mov
                         ah, 04Ah
                         bx, 0FFFFh
                mov
                         21h
                int
                         ax, 10h
                mov
                mul
                         si, offset AVAILABLEMEMORY
                mov
                         si, 23h
                add
                         _TO_DEC
                call
                         si
                pop
                         dx
                pop
                         bx
                pop
                pop
                         ax
                ret
_AVAILABLEMEMORY ENDP
_EXTENDEDMEMORY PROC
                             near; Поиск расширенной памяти
                push
                         ax
                         bx
                push
                push
                         si
```

```
push
                          dx
                                   al, 30h
                 mov
                                   70h, al
                 out
                                   al, 71h
                 in
                                   bl, al
                 mov
                                   al, 31h
                 mov
                 out
                                   70h, al
                                   al, 71h
                 in
                          ah, al
                 mov
                 mov
                          al, bl
                          dx, dx
                 sub
                          si, offset EXTENDEDMEMORY
                 mov
                          si, 28
                 add
                          _TO_DEC
                 call
                                   dx
                 pop
                                   si
                 pop
                                   bx
                 pop
                 pop
                 ret
_EXTENDEDMEMORY ENDP
_DATA PROC near; Поиск МСВ
                mov
                          di, offset DATA; Адрес МСВ
                 mov
                          ax, es
                 add
                          di, 05h
                          WRD_TO_HEX
                 call
                          di, offset DATA; Тип МСВ
                 mov
                          di, 0Fh
                 add
                xor
                          ah, ah
                          al, es:[00h]
                 mov
                          BYTE_TO_HEX
                call
                          [di], al
                 mov
                 inc
                          di
                          [di], ah
                 mov
                          di, offset DATA
                 mov
                          ax, es:[01h]
                 mov
                          di, 1Dh
                 add
                 call
                          WRD_TO_HEX
                          di, offset DATA
                 mov
                 mov
                          ax, es:[03h]
                          bx, 10h
                 mov
                 mul
                          bx
                 add
                          di, 2Eh
                 push
                          si
                          si, di
                 mov
                          _TO_DEC
                 call
                 pop
                          di, offset DATA
                 mov
                          di, 35h
                 add
                          bx, 0h
                 mov
                 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
                          ah, 52h
                 mov
                          21h
                 int
                 sub
                          bx, 2h
                          es, es:[bx]
                 mov
                 output_:
                                  _DATA
                         call
                         mov
                                  dx, offset DATA
                                  PRINT
                         call
                         mov
                                  cx, es
                         add
                                  ax, cx
                         inc
                                  ax
                                  es, ax
                         mov
                         cmp
                                  bl, 4Dh
                         je
                                           output_
                 ret
OUTPUT ENDP
BEGIN:
                         _AVAILABLEMEMORY
                 call
                                  dx, offset AVAILABLEMEMORY
                 mov
                         PRINT
                 call
                         _EXTENDEDMEMORY
                 call
                                  dx, offset EXTENDEDMEMORY
                 mov
                         PRINT
                 call
                         аh, 4ah ; Освобождение памяти
                 mov
                         bx, offset END_PROG
                 mov
                 int
                         21h
                                  аһ, 48h ; Запрос 64 КВ памяти
                 mov
                         bx, 1000h
                 mov
                         21h
                 int
                 mov
                         dx, offset HEAD
                         PRINT
                 call
                         OUTPUT
                 call
                         al, al
                 xor
                         ah, 4ch
                 mov
                         21h
                 int
                 \mathsf{END}\mathsf{\_PROG}\,\mathsf{db}\,0
TESTPC ENDS
```

END START

21

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
                        al,bh
               mov
                        BYTE_TO_HEX
               call
                        [di],ah
               mov
               dec
                        di
                        [di],al
               mov
                        bx
               pop
               ret
WRD\_TO\_HEX
                       ENDP
;-----
BYTE_TO_DEC
                       PROC
                               near
               push
                        cx
               push
                        dx
                        ah,ah
               xor
                        dx,dx
               xor
                        cx,10
               mov
loop_bd:div
                cx
                                d1,30h
               or
                        [si],dl
               mov
               dec
                        si
                          dx,dx
               xor
                        ax,10
               cmp
                        loop_bd
               jae
                                al,00h
               cmp
                                end_l
               je
                                al,30h
               or
                        [si],al
               mov
end_l: pop
                dx
               pop
                                cx
               ret
BYTE_TO_DEC
                       ENDP
;-----
_TO_DEC
                       PROC
                               near
               push
                        cx
               push
                        dx
               push
               mov
                                cx,10
_loop_bd:
               div
                                cx
                                d1,30h
               or
                        [si],dl
               mov
               dec
                        si
                                dx,dx
               xor
                                ax,10
               cmp
                                _loop_bd
               jae
                                ax,00h
               cmp
```

```
_end_l
               jbe
                                al,30h
               or
               mov
                                [si],al
_end_l:
               pop
                                ax
                                dx
               pop
                                cx
               pop
               ret
_TO_DEC
                       ENDP
PRINT PROC NEAR
               push
                        ax
               mov
                        ah, 09h
                21h
         int
         pop
                        ax
         ret
PRINT ENDP
_AVAILABLEMEMORY PROC NEAR; Поиск доступной памяти
               push
                        ax
               push
                        bx
                        dx
               push
               push
                        \dot{si}
               xor
                        ax, ax
                        ah, 04Ah
               mov
                        bx, 0FFFFh
               mov
                        21h
               int
               mov
                        ax, 10h
                        bx
               mul
                        si, offset AVAILABLEMEMORY
               mov
               add
                        si, 23h
                        _TO_DEC
               call
               pop
                        si
                        dx
               pop
               pop
                        bx
               pop
                        ax
               ret
_AVAILABLEMEMORY ENDP
;-----
_EXTENDEDMEMORY PROC near; Поиск расширенной памяти
               push
                        ax
               push
                        bx
```

push

si

```
push
                        dx
                                 al, 30h
                mov
                                 70h, al
                out
                                 al, 71h
                in
                                 bl, al
                mov
                                 al, 31h
                mov
                                 70h, al
                out
                                 al, 71h
                in
                         ah, al
                mov
                        al, bl
                mov
                sub
                        dx, dx
                        si, offset EXTENDEDMEMORY
                mov
                         si, 28
                add
                         _TO_DEC
                call
                pop
                                 dx
                                 si
                pop
                                 bx
                pop
                pop
                                 ax
                ret
_EXTENDEDMEMORY ENDP
;-----
_DATA PROC near; Поиск МСВ
                        di, offset DATA ; Адрес МСВ
                mov
                        ax, es
                mov
                        di, 05h
                add
                call
                         WRD_TO_HEX
                        di, offset DATA; Тип МСВ
                mov
                        di, 0Fh
                add
                xor
                        ah, ah
                        al, es:[00h]
                mov
                        BYTE_TO_HEX
                call
                        [di], al
                mov
                        di
                inc
                mov
                        [di], ah
                        di, offset DATA
                mov
                        ax, es:[01h]
                mov
                        di, 1Dh
                add
                call
                         WRD_TO_HEX
```

di, offset DATA

ax, es:[03h]

mov

mov

```
bx, 10h
                 mov
                mul
                          bx
                 add
                          di, 2Eh
                 push
                          si
                          si, di
                 mov
                          _TO_DEC
                 call
                          si
                 pop
                          di, offset DATA
                 mov
                          di, 35h
                 add
                          bx, 0h
                 mov
                 print_:
                                  mov dl, es:[bx + 8]
                                  mov [di], dl
                                  inc di
                                  inc bx
                                  cmp bx, 8h
                 jne
                          print_
                          ax, es:[3h]
                 mov
                 mov
                          bl, es:[0h]
                 ret
_DATA ENDP
OUTPUT PROC NEAR ; Поиск цепочки блоков управления памятью
                 mov
                          ah, 52h
                          21h
                 int
                          bx, 2h
                 sub
                          es, es:[bx]
                 mov
                 output_:
                         call
                                  _DATA
                                  dx, offset DATA
                         mov
                                  PRINT
                         call
                         mov
                                  cx, es
                         add
                                  ax, cx
                         inc
                                  ax
                         mov
                                  es, ax
                                  bl, 4Dh
                         cmp
                         je
                                           output\_
                 ret
OUTPUT ENDP
BEGIN:
                          _AVAILABLEMEMORY
                 call
                                  dx, offset AVAILABLEMEMORY
                 mov
                          PRINT
                 call
```

_EXTENDEDMEMORY call

dx, offset EXTENDEDMEMORY mov

PRINT call

аһ, 48h; Запрос 64 КВ памяти mov

bx, 1000h mov int 21h

jc тетотуЕтт ; Проверка на ошибку

jmp next_

memoryErr:

mov dx, offset ErrorM

PRINT call

next_: ; Освобождение памяти

ah, 4ah mov

bx, offset PROGRAMM_ENDS mov

int

mov dx, offset HEAD

PRINT call call OUTPUT

al, al xor ah, 4ch

21h int

mov

 ${\tt PROGRAMM_ENDS~db~0}$

TESTPC ENDS

END START