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

ОТЧЕТ

по лабораторной работе №6

по дисциплине «Операционные системы»

Тема: Построение модуля динамической структуры

Студент гр. 7381 _____ Трушников А.П.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2019

Цель работы.

Исследовать возможности построения загрузочного модуля динамической структуры. Исследовать интерфейс между вызывающим и вызываемым модулями по управлению и по данным. Для запуска вызываемого модуля используется функция 4B00h прерывания int 21h. Все загрузочные модули находятся в одном каталоге. Необходимо обеспечить возможность запуска модуля динамической структуры из любого каталога. стандартному прерыванию.

Описание функций.

Название	Описание
Write_message	Вывод сообщения на экран
Main	Основная функция

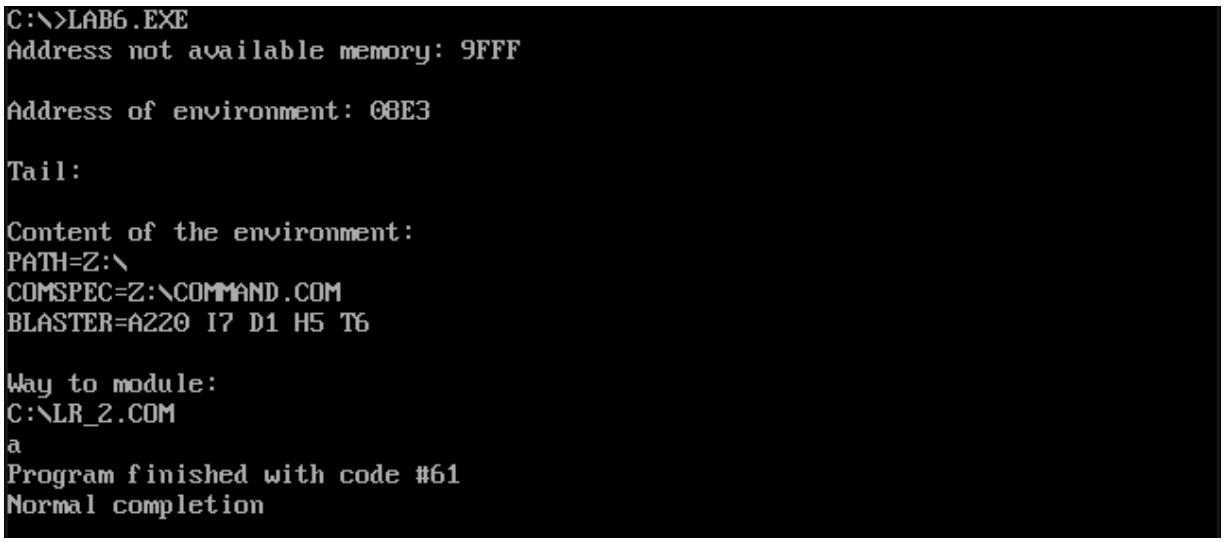
Описание структур данных.

Название	Описание
error_4Ah	Сообщение об ошибке в 4Ah
error4Ah_7	Сообщение об ошибке с кодом 7
error4Ah_8	Сообщение об ошибке с кодом 8
error4Ah_9	Сообщение об ошибке с кодом 9
error_4Bh	Сообщение об ошибке в 4Bh
error4Bh_1	Сообщение об ошибке с кодом 1
error4Bh_2	Сообщение об ошибке с кодом 2
error4Bh_5	Сообщение об ошибке с кодом 5
error4Bh_8	Сообщение об ошибке с кодом 8
error4Bh_10	Сообщение об ошибке с кодом 10
error4Bh_11	Сообщение об ошибке с кодом 11
finish_message	Сообщение о завершении программы (с указанием кода завершения)
finish_code_0	Сообщение о причине завершения с кодом 0
finish_code_1	Сообщение о причине завершения с кодом 1
finish_code_2	Сообщение о причине завершения с кодом 2
finish_code_3	Сообщение о причине завершения с кодом 3
_enter	enter

param_block	Хранение блока параметров
file_path	Хранение пути к файлу
Keep_SS	Хранение содержимого SS
Keep_SP	Хранение содержимого SP
position	Хранение позиции после слеша в имени файла

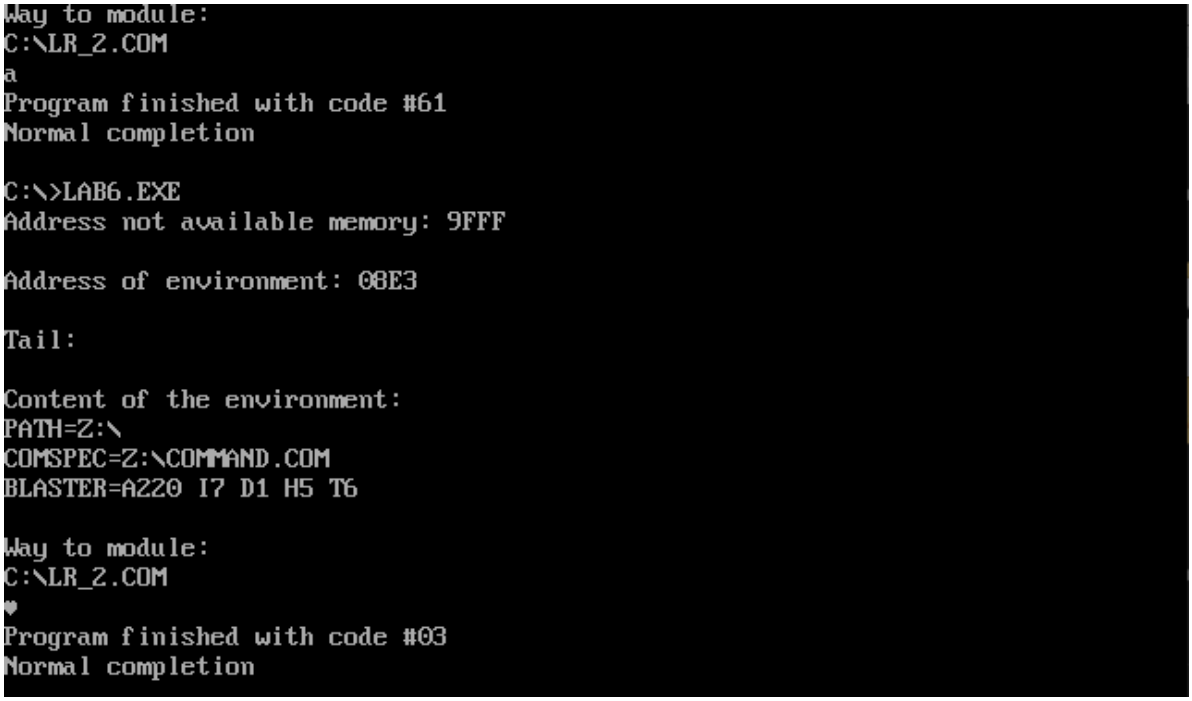
Выполнение работы.

Шаг 1.



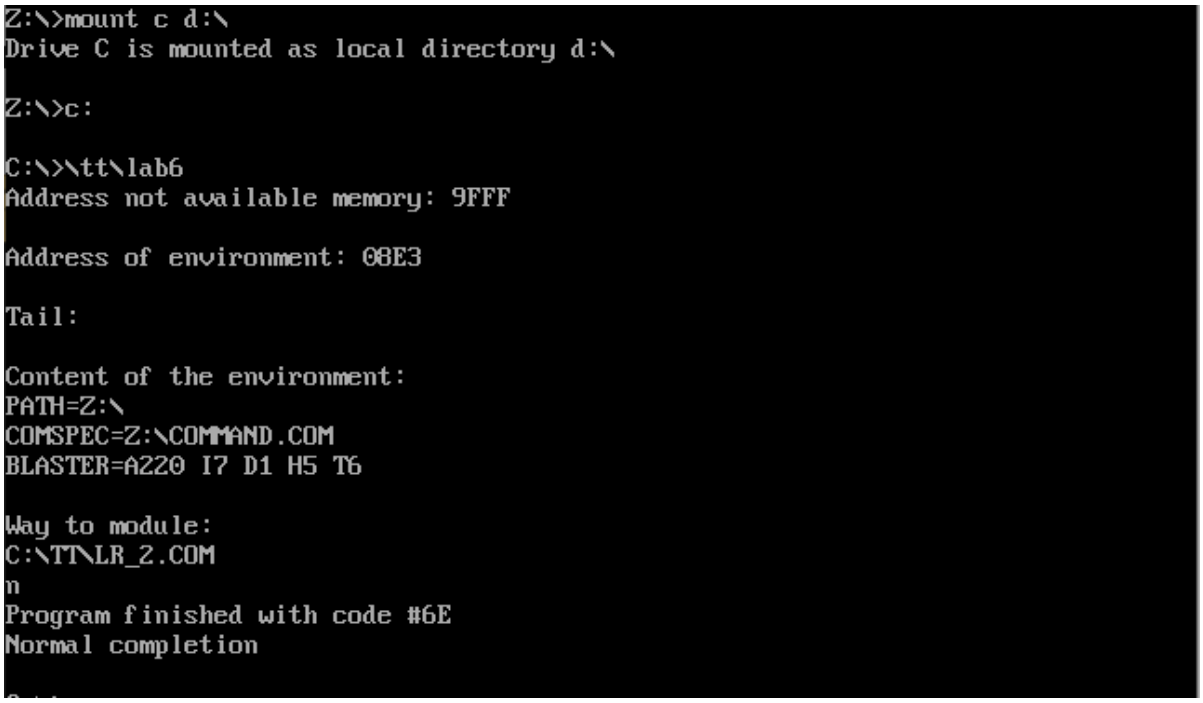
Программа и модуль в текущем каталоге. Завершение по нажатию клавиши.

Шаг 2.



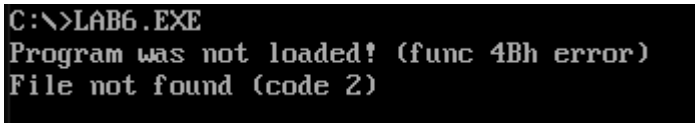
Программа и модуль в текущем каталоге. Завершение по нажатию клавиш Ctrl+C.

Шаг 3.



Программа и модуль находятся не в текущем каталоге. Завершение по нажатию клавиши.

Шаг 4.



Программа и модуль в разных каталогах. Ошибка в функции 4Bh: файл-модуль не найден.

Ответы на контрольные вопросы.

1. Как реализовано прерывание CTRL+C?

Функция 01h проверяет наличие в буфере символов нажатия комбинации Ctrl+C, при обнаружении которых производится вызов прерывания int 23h.

2. В какой точке заканчивается вызываемая программа, если код завершения 0?

Программа заканчивается при вызове функции 4Ch прерывания int 21h.

3. В какой точке заканчивается вызываемая программа по прерывания Ctrl+C?

Программа заканчивается на этапе ожидания ввода символа с клавиатуры (на функции 01h прерывания int 21h).

Выводы.

В ходе данной лабораторной работы я исследовал возможность построения загрузочного модуля динамической структуры, а также изучил возможности функций 4Ah и 4Bh прерывания int 21h.

ПРИЛОЖЕНИЕ А

LAB6.ASM

```
ASTACK          SEGMENT  STACK
    DW 256 DUP(?)
ASTACK          ENDS

DATA            SEGMENT
        ERROR_4AH    DB 'MEMORY CAN NOT BE FREED! (FUNC 4AH ERROR)', 0DH, 0AH, '$'
          ;ПРИЧИНА НЕУДАЧНОГО ВЫПОЛНЕНИЯ ФУНКЦИИ 4AH
      ERROR4AH_7  DB 'CONTROL UNIT MEMORY DESTROYED (CODE 7)', 0DH, 0AH, '$'
        ;РАЗРУШЕН УПРАВЛЯЮЩИЙ БЛОК ПАМЯТИ (КОД 7)
      ERROR4AH_8        DB 'NOT ENOUGHT MEMORY TO PERFORM THE FUNCTION (CODE 8)',
0DH, 0AH, '$'   ;НЕДОСТАТОЧНО ПАМЯТИ ДЛЯ ВЫПОЛНЕНИЯ ФУНКЦИИ (КОД 8)
      ERROR4AH_9        DB 'INVALID ADRESS OF THE MEMORY BLOCK (CODE 9)', 0DH, 0AH,
'$'                     ;НЕВЕРНЫЙ АДРЕС БЛОКА ПАМЯТИ (КОД 9)

        ERROR_4BH    DB 'PROGRAM WAS NOT LOADED! (FUNC 4BH ERROR)', 0DH, 0AH, '$'
    ;ПРИЧИНА НЕУДАЧНОГО ЗАВЕРШЕНИЯ ФУНКЦИИ 4BH
        ERROR4BH_1  DB 'INCORRECT NUMBER OF THE FUNCTION (CODE 1)', 0DH, 0AH, '$'
;НЕВЕРНЫЙ НОМЕР ФУНКЦИИ (КОД 1)
        ERROR4BH_2  DB 'FILE NOT FOUND (CODE 2)', 0DH, 0AH, '$'
            ;ФАЙЛ НЕ НАЙДЕН (КОД 2)
        ERROR4BH_5  DB 'HARD DRIVE ERROR (CODE 5)', 0DH, 0AH, '$'
        ;ОШИБКА ДИСКА (КОД 5)
        ERROR4BH_8  DB 'NOT ENOUGHT MEMORY (CODE 8)', 0DH, 0AH, '$'
            ;НЕДОСТАТОЧНО ПАМЯТИ (КОД 8)
        ERROR4BH_10 DB 'INVALID STRING (CODE 10)', 0DH, 0AH, '$'
        ;НЕПРАВИЛЬНАЯ СТРОКА СРЕДЫ (КОД 10)
        ERROR4BH_11 DB 'INCORRECT FORMAT (CODE 11)', 0DH, 0AH, '$'
        ;НЕВЕРНЫЙ ФОРМАТ (КОД 11)

        FINISH_MESSAGE DB 0DH, 0AH, 'PROGRAM FINISHED WITH CODE #$'
        ;СООБЩЕНИЕ О ЗАВЕРШЕНИИ ПРОГРАММЫ
        FINISH_CODE_0 DB   'NORMAL COMPLETION', 0DH, 0AH, '$'
            ;НОРМАЛЬНОЕ ЗАВЕРШЕНИЕ (КОД 0)
        FINISH_CODE_1 DB   'COMPLETION BY CTRL-BREAK', 0DH, 0AH, '$'
        ;ЗАВЕРШЕНИЕ ПО CTRL+BREAK
        FINISH_CODE_2 DB   'COMPLETION BY DEVICE ERROR', 0DH, 0AH, '$'
        ;ЗАВЕРШЕНИЕ ПО ОШИБКЕ УСТРОЙСТВА
        FINISH_CODE_3 DB        'COMPLETION BY 31H FUNCTION', 0DH, 0AH, '$'
    ;ЗАВЕРШЕНИЕ ПО ФУНКЦИИ 31H, ОСТАВЛЯЮЩЕЙ ПРОГРАММУ РЕЗИДЕНТНОЙ В
ПАМЯТИ
        _ENTER DB 0DH, 0AH, '$'

DATA            ENDS

CODE   SEGMENT
```

```
        PARAM_BLOCK DB 14 DUP(0)   ;МЕСТО ПОД БЛОК ПАРАМЕТРОВ
        FILE_PATH   DB 30 DUP(0)   ;МЕСТО ПОД ПУТЬ ДО ФАЙЛА (ОСТАВЛЯЕМ МЕСТО С
ЗАПАСОМ)
        KEEP_SS     DW ?           ;МЕСТО ПОД ХРАНЕНИЕ СОДЕРЖИМОГО РЕГИСТРА
SS
        KEEP_SP     DW ?           ;МЕСТО ПОД ХРАНЕНИЕ СОДЕРЖИМОГО РЕГИСТРА
SP
        POSITION    DW 0
      ASSUME CS:CODE, DS:DATA, SS:ASTACK

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

; ФУНКЦИЯ ВЫВОДА СООБЩЕНИЯ НА ЭКРАН
WRITE_MESSAGE   PROC
        PUSH AX
        MOV AH, 09H
        INT 21H
        POP AX
        RET
WRITE_MESSAGE   ENDP


; ГЛАВНАЯ ФУНКЦИЯ
MAIN  PROC
        MOV AX, DATA
        MOV DS, AX

;1)  ПОДГОТОВКА МЕСТА В ПАМЯТИ
        MOV AX, ALL_MEMORY      ;ВСЯ ПАМЯТЬ, ВЫДЕЛЕННАЯ ПРОГРАММЕ
```

```
        MOV BX, ES                  ;ИСПОЛЬЗУЕМАЯ ПАМЯТЬ
        SUB AX, BX                  ;ВЫЧИСЛЯЕМ ОСТАТОК
        MOV CX, 0004H
        SHL AX, CL                  ;ПЕРЕВОДИМ В ПАРАГРАФЫ
        MOV BX, AX                  ;УКАЗЫВАЕМ ВХОДНОЙ ПАРАМЕТР ДЛЯ ФУНКЦИИ 4AH

        MOV AX, 4A00H
        INT 21H                     ;ВЫПОЛНЯЕМ ФУНКЦИЮ (В AX ЗАНЕСЁТСЯ КОД ОШИБКИ ЕСЛИ ОНА БУДЕТ)

    ;1.1) ОБРАБОТКА ОШИБОК ФУНКЦИИ 4AH
        JNC NO_ERROR_4AH  ;ЕСЛИ ФЛАГ CF=0 - ОШИБОК НЕ БЫЛО, ДВИЖЕМСЯ ДАЛЕЕ

        LEA DX, ERROR_4AH ;ВЫВОДИМ СООБЩЕНИЕ О ТОМ, ЧТО ПРОИЗОШЛА ОШИБКА В ФУНКЦИИ 4AH
        CALL WRITE_MESSAGE

        CMP AX, 7                   ;ОШИБКА С КОДОМ 7
        JE ERROR4AH_7_LABEL ;СООТВЕТСТВУЮЩАЯ ОБРАБОТКА

        CMP AX, 8                   ;ОШИБКА С КОДОМ 8
        JE ERROR4AH_8_LABEL ;СООТВЕТСТВУЮЩАЯ ОБРАБОТКА

        CMP AX, 9                   ;ОШИБКА С КОДОМ 9
        JE ERROR4AH_8_LABEL ;СООТВЕТСТВУЮЩАЯ ОБРАБОТКА

    ERROR4AH_7_LABEL:
        LEA DX, ERROR4AH_7
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ERROR4AH_8_LABEL:
        LEA DX, ERROR4AH_8
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ERROR4AH_9_LABEL:
        LEA DX, ERROR4AH_9
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ;2)  СОЗДАНИЕ БЛОКА ПАРАМЕТРОВ
    NO_ERROR_4AH:
        MOV BYTE PTR [PARAM_BLOCK], 0 ;ЕСЛИ СЕГМЕНТНЫЙ АДРЕС СРЕДЫ 0 - ВЫЗЫВАЕМАЯ ПРОГРАММАЯ НАСЛЕДУЕТ СРЕДУ ВЫЗЫВАЮЩЕЙ

    ;3)  ПОДГОТОВКА СТРОКИ, СОДЕРЖАЩЕЙ ПУТЬ И ИМЯ ВЫЗЫВАЕМОЙ ПРОГРАММЫ
```

```
        MOV ES, ES:[2CH]
        MOV SI, 0
FIND_ZERO:
        MOV AX, ES:[SI]
        INC SI
        CMP AX, 0000H
        JNE FIND_ZERO

        ADD SI, 3
        MOV DI, 0

WRITE:
                MOV CL, ES:[SI]
                CMP CL, 0
                JE      CONT
                CMP CL, '\'
                JNE NOT_POS
                MOV POSITION, DI
NOT_POS:
                MOV BYTE PTR [FILE_PATH+DI], CL
                INC SI
                INC DI
                JMP WRITE

CONT:
                MOV BX, POSITION
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], 'L'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], 'R'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], '_'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], '2'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], '.'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], 'C'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], 'O'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], 'M'
                INC BX
                MOV BYTE PTR [FILE_PATH+BX], '$'

;4)  СОХРАНЕНИЕ РЕГИСТРОВ SS И SP
        PUSH DS                         ;ЗАПОМИНАЕМ В СТЕК DS
        PUSH ES                         ;И ES
```

```
        MOV KEEP_SP, SP             ;СОХРАНЯЕМ SP
        MOV KEEP_SS, SS             ;И SS

;5)  ПОДГОТОВКА И ВЫПОЛНЕНИЕ ФУНКЦИИ 4BH
        MOV SP, 0FEH
        MOV AX, CODE
        MOV DS, AX
        MOV ES, AX

        LEA BX, PARAM_BLOCK     ;В BX - БЛОК ПАРАМЕТРОВ
        LEA DX, FILE_PATH   ;В DX - ПУТЬ К ФАЙЛУ
        MOV AX, 4B00H               ;ВЫЗЫВАЕМ ФУНКЦИЮ 4B
        INT 21H

        MOV SS, CS:KEEP_SS ;ВОССТАНАВЛИВАЕМ SS
        MOV SP, CS:KEEP_SP ;И SP

        POP ES                      ;ВОССТАНАВЛИВАЕМ ИЗ ОБНОВЛЁННЫХ
        POP DS                      ;SS И SP ES И DS

        JNC NO_ERROR_4BH  ;ЕСЛИ ФЛАГ CF=0 - ОШИБОК НЕТ, ИДЁМ ДАЛЕЕ

;5.1)  ОБРАБОТКА ОШИБОК ОТ ФУНКЦИИ 4BH

        LEA DX, ERROR_4BH ;ВЫВОДИМ СООБЩЕНИЕ О ТОМ,
        CALL WRITE_MESSAGE      ;ЧТО ПРОИЗОШЛА ОШИБКА В 4BH

        CMP AX, 1
        JE ERROR4BH_1_LABEL

        CMP AX, 2
        JE ERROR4BH_2_LABEL

        CMP AX, 5
        JE ERROR4BH_5_LABEL

        CMP AX, 8
        JE ERROR4BH_8_LABEL

        CMP AX, 10
        JE ERROR4BH_10_LABEL

        CMP AX, 11
        JE ERROR4BH_11_LABEL

ERROR4BH_1_LABEL:
        LEA DX, ERROR4BH_1
        CALL WRITE_MESSAGE
```

```
        JMP ERROR_FINISH

    ERROR4BH_2_LABEL:
        LEA DX, ERROR4BH_2
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ERROR4BH_5_LABEL:
        LEA DX, ERROR4BH_5
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ERROR4BH_8_LABEL:
        LEA DX, ERROR4BH_8
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ERROR4BH_10_LABEL:
        LEA DX, ERROR4BH_10
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ERROR4BH_11_LABEL:
        LEA DX, ERROR4BH_11
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH

    ;6)  ОБРАБОТКА ЗАВЕРШЕНИЯ ПРОГРАММЫ
    NO_ERROR_4BH:
        MOV AX, 4D00H       ;ВЫЗЫВАЕМ ФУНКЦИЮ 4DH ПРЕРЫВАНИЯ INT 21H
        INT 21H                 ;В КАЧЕСТВЕ РЕЗУЛЬТАТА В РЕГИСТРЕ АН БУДЕТ
ПРИЧИНА ЗАВЕРШЕНИЯ

        LEA DX, FINISH_MESSAGE
        CALL WRITE_MESSAGE
        CALL   BYTE_TO_HEX
        PUSH   AX
        MOV    AH,02H
        MOV    DL,AL
        INT    21H
        POP    AX
        PUSH   AX
        XCHG   AH,AL
        MOV    AH,02H
        MOV    DL,AL
        INT    21H
        POP          AX
        LEA DX, _ENTER
```

```
        CALL WRITE_MESSAGE

        CMP AH, 0
        JE FINISHCODE_0
        CMP AH, 1
        JE FINISHCODE_1
        CMP AH, 2
        JE FINISHCODE_2
        CMP AH, 3
        JE FINISHCODE_3
FINISHCODE_0:
        LEA DX, FINISH_CODE_0
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH
FINISHCODE_1:
        LEA DX, FINISH_CODE_1
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH
FINISHCODE_2:
        LEA DX, FINISH_CODE_2
        CALL WRITE_MESSAGE
        JMP ERROR_FINISH
FINISHCODE_3:
        LEA DX, FINISH_CODE_3
        CALL WRITE_MESSAGE
ERROR_FINISH:
        MOV AH, 4CH             ;ЗАВЕРШЕНИЕ ПО ФУНКЦИИ 4C
        INT 21H

MAIN   ENDP
CODE                    ENDS
ALL_MEMORY          SEGMENT       ;ПУСТОЙ СЕГМЕНТ В КОНЦЕ
ALL_MEMORY  ENDS ;ДЛЯ ОПРЕДЕЛЕНИЯ ПАМЯТИ, КОТОРАЯ НЕ ИСПОЛЬЗУЕМСЯ В CS
                END MAIN
```