**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №6**

**по дисциплине «Операционные системы»**

**Тема: Построение модуля динамической структуры**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 6381 |  | Герасимова Д.В. |
| Преподаватель |  | Губкин А.Ф. |

Санкт-Петербург

2018

**Цель работы:**

Исследование возможности построения загрузочного модуля динамической структуры. В отличии от предыдущих лабораторных работ в этой работе рассматривается приложение, состоящее из нескольких модулей, а не из одного модуля простой структуры. В этом случае разумно предположить, что все модули приложения находятся в одном каталоге и полный путь в этот каталог можно взять из среды, как это делалось в работе 2. Понятно, что такое приложение должно запускаться в соответствии со стандартами ОС.

В работе исследуется 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.

**Постановка задачи:**

**Шаг 1.** Для выполнения лабораторной работы необходимо написать и отладить программный модуль типа .EXE, который выполняет функции:

1) Подготавливает параметры для запуска загрузочного модуля из того же каталога, в котором находится он сам. Вызываемому модулю передается новая среда, созданная вызывающим модулем и новая командная строка.

2) Вызываемый модуль запускается с использованием загрузчика.

3) После запуска проверяется выполнение загрузчика, а затем результат выполнения вызываемой программы. Необходимо проверять причину завершения и, в зависимости от значения, выводить соответствующее сообщение. Если причина завершения 0, то выводится код завершения.

В качестве вызываемой программы необходимо взять программу ЛР 2, которая распечатывает среду и командную строку. Эту программу следует немного модифицировать, вставив перед выходом из нее обращение к функции ввода символа с клавиатуры. Введенное значение записывается в регистр AL и затем происходит обращение к функции выхода 4Ch прерывания int 21h.

**Шаг 2.** Запустите отлаженную программу, когда текущим каталогом является каталог с разработанными модулями. Программа вызывает другую программу, которая останавливается, ожидая символ с клавиатуры.

Введите произвольный символ из числа A-Z. Посмотрите причину завершения и код. Занесите полученные данные в отчет.

**Шаг 3.** Запустите отлаженную программу, когда текущим каталогом является каталог с разработанными модулями. Программа вызывает другую программу, которая останавливается, ожидая символ с клавиатуры.

Введите комбинацию символов Ctrl-C. Посмотрите причину завершения и код. Занесите полученные данные в отчет.

**Шаг 4.** Запустите отлаженную программу, когда текущим каталогом является какой- либо другой каталог, отличный от того, в котором содержатся разработанные программные модули.

Повторите ввод комбинаций клавиш. Занесите полученные данные в отчет.

**Шаг 5.** Запустите отлаженную программу, когда модули находятся в разных каталогах. Занесите полученные данные в отчет.

**Необходимые сведения для составления программы:**

Для загрузки и выполнения одной программы из другой используется функция 4B00h прерывания int 21h (загрузчик ОС). Перед обращением к этой функции необходимо выполнить следующие действия:

1) Подготовить место в памяти. При начальном запуске программы ей отводится вся доступная в данный момент память OS, поэтому необходимо освободить место в памяти. Для этого можно использовать функцию 4Ah прерывания int 21h. Эта функция позволяет уменьшить отведенный программе блок памяти. Перед вызовом функции надо определить объем памяти, необходимый программе ЛР6 и задать в регистре ВХ число параграфов, которые будут выделяться программе. Если функция 4Ah не может быть выполнена, то устанавливается флаг переноса CF=1 и в АХ заносится код ошибки:

*7 - разрушен управляющий блок памяти;*

*8 - недостаточно памяти для выполнения функции;*

*9- неверный адрес блока памяти.*

Поэтому после выполнения каждого прерывания int 21h следует проверять флаг переноса CF=1.

2) Создать блок параметров. Блок параметров - это 14-байтовый блок памяти, в который помещается следующая информация:

dw сегментный адрес среды

dd сегмент и смещение командной строки

dd сегмент и смещение первого FCB

dd сегмент и смещение второго FCB

Если сегментный адрес среды 0, то вызываемая программа наследует среду вызывающей программы. В противном случае вызывающая программа должна сформировать область памяти в качестве среды, начинающуюся с адреса кратного 16 и поместить этот адрес в блок параметров.

Командная строка записывается в следующем формате:

первый байт - счетчик, содержащий число символов в командной строке, затем сама командная строка, содержащая не более 128 символов.

На блок параметров перед загрузкой вызываемой программы должны указывать ES:BX.

3) Подготовить строку, содержащую путь и имя вызываемой программы. В конце строки должен стоять код ASCII 0. На подготовленную строку должны указывать DS:DX.

4) Сохранить содержимое регистров SS и SP в переменных. При восстановлении SS и SP нужно учитывать, что DS необходимо также восстановить.

Когда вся подготовка выполнена, вызывается загрузчик OS следующей последовательностью команд:

mov AX,4B0 0h int 21h

Если вызываемая программа не была загружена, то устанавливается флаг переноса CF=1 и в AX заносится код ошибки:

*1 - если номер функции неверен;*

*2 - если файл не найден;*

*5 - при ошибке диска;*

*8 - при недостаточном объеме памяти;*

*10 - при неправильной строке среды;*

*11 - если не верен формат.*

Если CF=0, то вызываемая программа выполнена и следует обработать ее завершение. Для этого необходимо воспользоваться функцией 4Dh прерывания int 21h. В качестве результата функция возвращает в регистре AH причину, а в регистре AL код завершения.

Причина завершения в регистре AH представляется следующими кодами:

*0 - нормальное завершение;*

*1 - завершение о Ctrl-Break;*

*2 - завершение o ошибке устройства;*

*3 - завершение по функции 31h, оставляющей программу резидентной.*

Код завершения формируется вызываемой программой в регистре AL перед выходом в OS с помощью функции 4Ch прерывания int 21h.

В качестве вызываемой программы целесообразно использовать программу, разработанную в Лабораторной работе №2, модифицировав ее следующим образом. Перед выходом из программы перед выполнением функции 4Ch прерывания int 21h следует за росить с клавиатуры символ и поместить введенный символ в регистр AL, в качестве кода завершения. Это можно сделать с помощью функции 01h прерывания int 21h.

mov AH,01h int 21h

Введенный символ остается в регистре AL и служит аргументом для функции 4Ch прерывания int 21h.

**В программе используются следующие процедуры:**

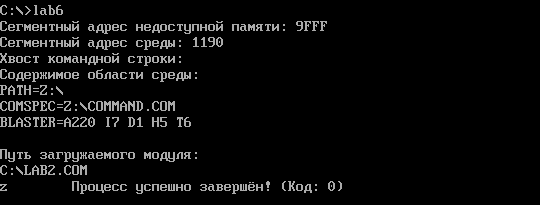
|  |  |
| --- | --- |
| **Название процедуры** | **Назначение** |
| PRINT | Вывод строки |
| BEGIN | Основная процедура программы |

**Определение структуры данных:**

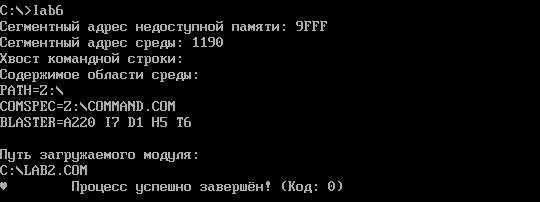
|  |  |  |
| --- | --- | --- |
| **Название поля данных** | **Тип** | **Назначение** |
| string\_1 | db | Процесс успешно завершён! (Код: 0) |
| string\_2 | db | Неверный номер функции |
| string\_3 | db | Файл не найден |
| string\_4 | db | Ошибка диска |
| string\_5 | db | Недостаточно памяти |
| string\_6 | db | Неправильная строка среды |
| string\_7 | db | Неверный формат |
| string\_8 | db | Разрушен управляющий блок памяти |
| string\_9 | db | Недостаточно памяти для выполнения функции |
| string\_10 | db | Неверный адрес блока памяти |
| string\_11 | db | Завершение по Ctrl + C (Код: 1) |
| string\_12 | db | Завершение по ошибке устройства (Код: 2) |
| string\_13 | db | Завершение по функции 31h, оставляющей программу резидентной (Код: 3) |

**Результат работы программ:**

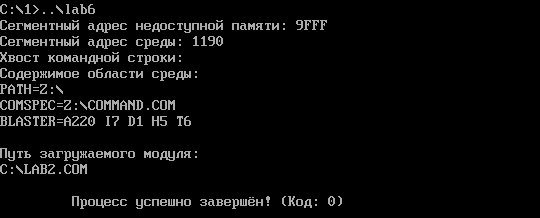
1. Запуск программы из каталога с разработанными модулями и ввод символа z



1. Запуск программы из каталога с разработанными модулями и завершение через Ctrl + Break (программа завершается нормально, т.к. обработка данного сочетание клавиш в DOSBox не реализована)



1. Запуск программы во время нахождения в другом каталоге



1. Запуск программы при условии, что программный и загрузочный модуль находятся в разных каталогах



**Ответы на контрольные вопросы:**

1. Как реализовано прерывание Ctrl-C?

**Ответ:** при нажатии клавиш Ctrl-C управление передаётся по адресу 0000:008Ch. Этот адрес копируется в PSP функциями 26h и 4Ch и восстанавливается из PSP при выходе из программы.

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

**Ответ:** если код завершения 0, то программа завершается при выполнении функции 4Ch прерывания int 21h;

1. В какой точке заканчивается вызываемая программа по прерыванию Ctrl-C?

**Ответ:** если во время выполнения программы было нажато Ctrl-C, то программа завершится непосредственно в том месте, в котором произошло нажатие сочетания клавиш (то есть в месте ожидания нажатия клавиши: 01h вектора прерывания 21h);

**Заключение:**

В ходе лабораторной работы был построен загрузочный модуль динамической структуры, а также модифицирован ранее построенный программный модуль. Изучены дополнительные функции работы с памятью и исследованы возможности использования интерфейса между вызывающим и вызываемым модулями по управлению и по данным.

**Приложение**

**Код программы lab6.asm**

.MODEL SMALL

.DATA

;ДАННЫЕ

string\_1 db ' Процесс успешно завершён! (Код: 0)', 0DH, 0AH, '$'

string\_2 db ' Неверный номер функции', 0DH, 0AH, '$'

string\_3 db ' Файл не найден', 0DH, 0AH, '$'

string\_4 db ' Ошибка диска', 0DH, 0AH, '$'

string\_5 db ' Недостаточно памяти', 0DH, 0AH, '$'

string\_6 db ' Неправильная строка среды', 0DH, 0AH, '$'

string\_7 db ' Неверный формат', 0DH, 0AH, '$'

string\_8 db ' Разрушен управляющий блок памяти', 0DH, 0AH, '$'

string\_9 db ' Недостаточно памяти для выполнения функции', 0DH, 0AH, '$'

string\_10 db ' Неверный адрес блока памяти', 0DH, 0AH, '$'

string\_11 db ' Завершение по Ctrl + C (Код: 1)', 0DH, 0AH, '$'

string\_12 db ' Завершение по ошибке устройства (Код: 2)', 0DH, 0AH, '$'

string\_13 db ' Завершение по функции 31h, оставляющей программу резидентной (Код: 3)', 0DH, 0AH, '$'

EOL db "$"

filename db 50 dup(0)

param dw 7 dup(?)

keep\_SS dw ?

keep\_SP dw ?

.STACK 200h

.CODE

;ПРОЦЕДУРЫ

;Вывод на экран

PRINT PROC

push ax

mov ah, 09h

int 21h

pop ax

ret

PRINT ENDP

;-----------------------------------------------------

; КОД

BEGIN PROC FAR

mov ax, @data

mov ds, ax

push si

push di

push es

push dx

mov es, es:[2Ch]

xor si, si

lea di, filename

env\_char:

cmp byte ptr es:[si], 00h

je env\_crlf

inc SI

jmp env\_next

env\_crlf:

inc si

env\_next:

cmp word ptr es:[si], 0000h

jne env\_char

add si, 4

abs\_char:

cmp byte ptr es:[si], 00h

je vot

mov dl, es:[si]

mov [di], dl

inc si

inc di

jmp abs\_char

vot:

sub di, 5

mov dl, '2'

mov [di], dl

add di, 2

mov dl, 'c'

mov [di], dl

inc di

mov dl, 'o'

mov [di], dl

inc di

mov dl, 'm'

mov [di], dl

inc di

mov dl, 0h

mov [di], dl

inc di

mov dl, EOL

mov [di], dl

pop dx

pop es

pop di

pop si

lea bx, ALL\_MEM

mov ax, es

sub bx, ax

mov cl, 4

shr bx, cl

mov ah, 4Ah

int 21h

jnc no\_errors\_4A

err:

cmp ax, 7

je err\_7

cmp ax, 8

je err\_8

cmp ax, 9

je err\_9

err\_7:

lea dx, string\_8

call PRINT

jmp Exit

err\_8:

lea dx, string\_9

call PRINT

jmp Exit

err\_9:

lea dx, string\_10

call PRINT

jmp Exit

no\_errors\_4A:

push ds

pop es

lea dx, filename

lea bx, param

mov keep\_SS, ss

mov keep\_SP, sp

mov ax, 4b00h

int 21h

mov ss, keep\_SS

mov sp, keep\_SP

jnc no\_errors\_4B

cmp AX, 1

je err\_1

cmp AX, 2

je err\_2

cmp AX, 5

je err\_5

cmp AX, 8

je err8

cmp AX, 10

je err\_10

cmp AX, 11

je err\_11

err\_1:

lea DX, string\_2

call PRINT

jmp Exit

err\_2:

lea DX, string\_3

call PRINT

jmp Exit

err\_5:

lea DX, string\_4

call PRINT

jmp Exit

err8:

lea DX, string\_5

call PRINT

jmp Exit

err\_10:

lea DX, string\_6

call PRINT

jmp Exit

err\_11:

lea DX, string\_7

call PRINT

jmp Exit

no\_errors\_4B:

mov AX, 4D00h

int 21h

cmp AH, 0

je normal

cmp AH, 1

je ctrl\_c

cmp AH, 2

je err\_dev

cmp AH, 3

je err\_31

normal:

lea DX, string\_1

call PRINT

jmp Exit

ctrl\_c:

lea DX, string\_11

call PRINT

jmp Exit

err\_dev:

lea DX, string\_12

call PRINT

jmp Exit

err\_31:

lea DX, string\_13

call PRINT

jmp Exit

Exit:

; Выход в DOS

mov ah, 4Ch

int 21h

BEGIN ENDP

;-----------------------------------------------------

ALL\_MEM PROC

ALL\_MEM ENDP

;-----------------------------------------------------

END BEGIN