**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Операционные системы»**

**Тема: Исследование организации управления основной памятью**

|  |  |  |
| --- | --- | --- |
| Студент гр. 7381 |  | Судакова П.С. |
| Преподаватель |  | Ефремов М. А. |

Санкт-Петербург

2019

**Цель лабораторной работы**

Для исследования организации управления памятью необходимо ориентироваться на тип основной памяти, реализованный в компьютере и способ организации, принятый в ОС. В лабораторной работе рассматривается нестраничная память и способ управления динамическими разделами. Для реализации управления памятью в этом случае строится список занятых и свободных участков памяти. Функции ядра, обеспечивающие управление основной памятью, просматривают и преобразуют этот список.

В лабораторной работе исследуются структуры данных и работа функций управления памятью ядра операционной системы.

**Постановка задачи**

**Шаг 1.** Необходимо написать и отладить программный модуль типа .COM, выбирает и распечатывает следующую информацию:

1. Количество доступной памяти.

2. Размер расширенной памяти.

3. Выводит цепочку блоков управления памятью.

Адреса при выводе представляются шестнадцатеричными числами. Объем памяти функциями управления памятью выводится в параграфах. Необходимо преобразовать его в байты и выводить в виде десятичных чисел. Последние восемь байт МСВ выводятся как символы, не следует преобразовывать их в шестнадцатеричные числа.

**Шаг 2.** Изменить программу таким образом, чтобы она освобождала память, которую она не занимает. Для этого используйте функцию 4Ah прерывания 21h (пример в разделе «Использование функции 4АН»).

**Шаг 3.** Изменить программу еще раз таким образом, чтобы после освобождения памяти, программа запрашивала 64Кб памяти функцией 48Н прерывания 21Н.

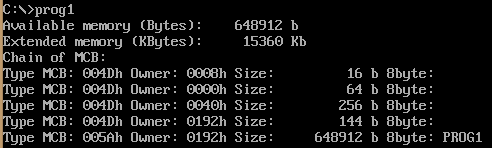
**Шаг 4.** Изменить первоначальный вариант программы, запросив 64Кб памяти функцией 48Н прерывания 21Н до освобождения памяти.

**Сведения о функциях, используемых в программе**

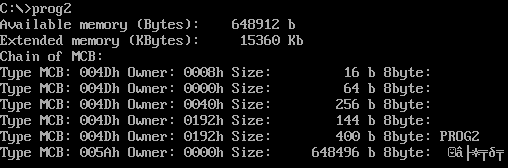
**Сведения о функциях и структурах данных, используемых в программе**

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Назначение |
| AVAILABLE\_STRING | db | Хранение строки «Available memory (Bytes):» |
| EXTENDED\_STRING | db | Хранение строки «Extended memory (KBytes):» |
| CHAIN\_STRING | db | Хранение строки «Chain of MCB:» |
| TYPE\_MCB | db | Хранение строки «Type MCB: 0000h» |
| OWNER | db | Хранение строки «Owner: 0000h» |
| SIZE\_SPACE | db | Хранение строки «Size:» |
| BYTE8 | db | Хранение строки «8byte:» |
| EMPTY\_SPACE | db | Хранение пустой строки « » |
| SPACE | db | Хранение символа пробела |
| BYTES | db | Хранение значка байта |
| KBYTES | db | Хранение значка килобайта |
| nl | db | Хранение символов перевода строки и возврата каретки |
| ERROR | db | Хранение строки «Memory ERROR.» |
| WRITE | PROC | Вызов процедур, необходимых для работы программы |
| PRINT | PROC | Печать на экран |
| END\_LINE | PROC | Перевод на новую строку |
| FREE\_M | PROC | Вывод пустой строки с символами пробела |
| AVAILABLE\_M | PROC | Определение доступной памяти |
| EXTENDED\_M | PROC | Определение расширенной памяти |
| PRINT\_MCB | PROC | Определение и вывод на экран типа MCB |
| TETR\_TO\_HEX | PROC | Перевод из четверичной системы счисления в шестнадцатиричную |
| BYTE\_TO\_HEX | PROC | Перевод байта в два символа 16-ти ричного числа |
| WRD\_TO\_HEX | PROC | Перевод слова в 4 символа 16-ти ричного числа |
| BYTE\_TO\_DEC | PROC | Перевод байта в 10-ти ричную систему счисления |

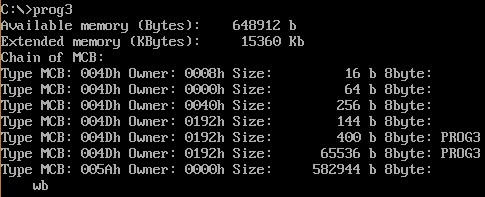
Работа программы **prog1.com** (Шаг 1)



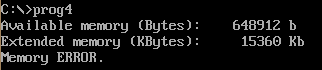
Работа программы **prog2.com** (Шаг 2)



Работа программы **prog3.com** (Шаг 3)



Работа программы **prog4.com** (Шаг 4)



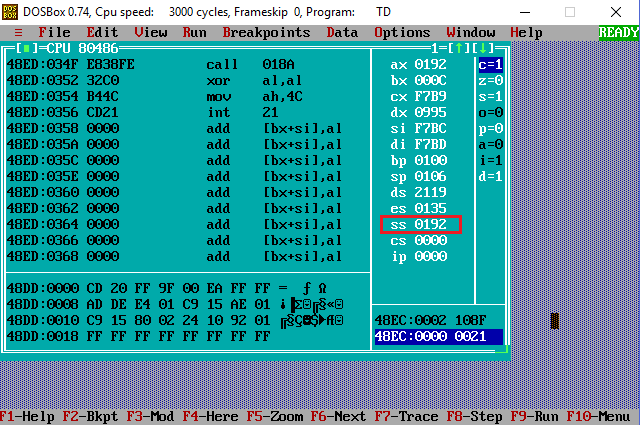
**Контрольные вопросы**

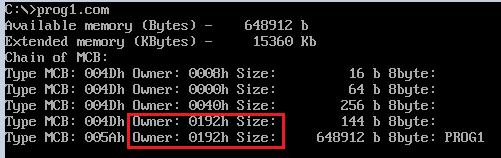
1. Что означает «доступный объем памяти?»

**Ответ:** Доступный объём памяти – это объём стандартной памяти (conventional memory), эта память представляет собой "нижние" 640 Кбайт ОЗУ. Для использования базовой памяти не нужны никакие дополнительные драйверы, поскольку операционная система MS DOS изначально создана для работы в адресах 0 - 640 Кбайт

1. Где MCB блок Вашей программы?

**Ответ:** Это можно определить по байтам 01h – 02h, где хранится идентификатор владельца процесса (PID - process identifier). PID для процесса получается от адреса сегмента, взятого из сегмента программного префикса (PSP - program segment prefix) данного процесса





Каждая загруженная программа имеет два распределенных для нее блока памяти, потому что при загрузке программы в память MS-DOS всегда присоединяет в начало программы блок операционной среды, запоминая в нем владельца блока памяти, который соотносится с каждой программой. Блок среды программы содержит собственную копию операционной среды MS-DOS.

1. Какой размер памяти занимает программа в каждом случае?

В первом случае он равен 648912+144=649056 байт



Во втором случае он равен 144 + 400 = 544 кб



В третьем случае 144+400+65536=66080 байт



В четвертом случае программа не может выделить запрашиваемые 64 кб так как она уже владеет всей доступной памятью, как в первом случае.

**Выводы.**

В результате выполнения данной лабораторной работы была исследована организация управления оперативной памятью. Была написана программа, которая выводит на экран количество доступной памяти, размер расширенной памяти и цепочку блоков управления памятью.