**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Операционные системы»**

Тема: «Исследование структур загрузочных модулей»

|  |  |  |
| --- | --- | --- |
| Студент гр. 6381 |  | Ширяев Я.А. |
| Преподаватель |  | Губкин А.Ф. |

Санкт-Петербург

2018

**Постановка задачи:**

Требуется написать текст исходного .COM модуля, который определяет тип PC и версию системы. Ассемблерная программа должна читать содержимое предпоследнего байта ROM BIOS, по таблице, сравнивая коды, определять тип РС и выводить строку с названием модели. Если код не совпадает ни с одним значением, то двоичный код переводиться в символьную строку, содержащую запись шестнадцатеричного числа и выводиться на экран в виде соответствующего сообщения. Затем определяется версия системы. Ассемблерная программа должна по значениям регистров AL и AH формировать текстовую строку в формате xx.yy, где xx - номер основной версии, а yy - номер модификации в десятичной системе счисления, формировать строки с серийным номером OEM (Original Equipment Manufacturer) и серийным номером пользователя. Полученные строки выводятся на экран.

Далее необходимо отладить полученный исходный модуль и получить «хороший» .COM модуль, а также необходимо построить «плохой» .EXE, полученный из исходного текста для .COM модуля.

Затем нужно написать текст «хорошего» .EXE модуля, который выполняет те же функции, что и модуль .COM, далее его построить, отладить и сравнить исходные тексты для .COM и .EXE модулей.

**Описание данных:**

\_TYPE – строка “OS TYPE: “

\_PC – строка “PC“ и переход на новую строку

\_PCXT – строка “PC/XT“ и переход на новую строку

\_AT – строка “AT“ и переход на новую строку

\_PS2\_30 – строка “PS2 model 30“и переход на новую строку

;\_PS2\_50\_60 – строка “PS2 модель 50/60 и переход на новую строку

\_PS2\_80 – строка “PS2 model 80 и переход на новую строку

\_PCJR – строка “PCjr“ и переход на новую строку

\_PC\_CONV – строка “PC Convertible“ и переход на новую строку

\_VER – строка “OS version: . “ и переход на новую строку

\_OEM – строка “OEM: “ и переход на новую строку

\_USSERN – строка “User serial number: “ и переход на новую строку

\_HEX – строка, содержащая символ перехода на новую строку

\_ENDL – строка, содержащая символы конца строки и перехода на новую строку.

**Описание процедур:**

PRINT – осуществляет вывод строки.

\_TYPE\_OS – выводит тип системы.

\_W\_OS – выводит версию MS DOS

\_W\_OEM – выводит серийный номер OEM

\_W\_SERN – выводит серийный номер пользователя

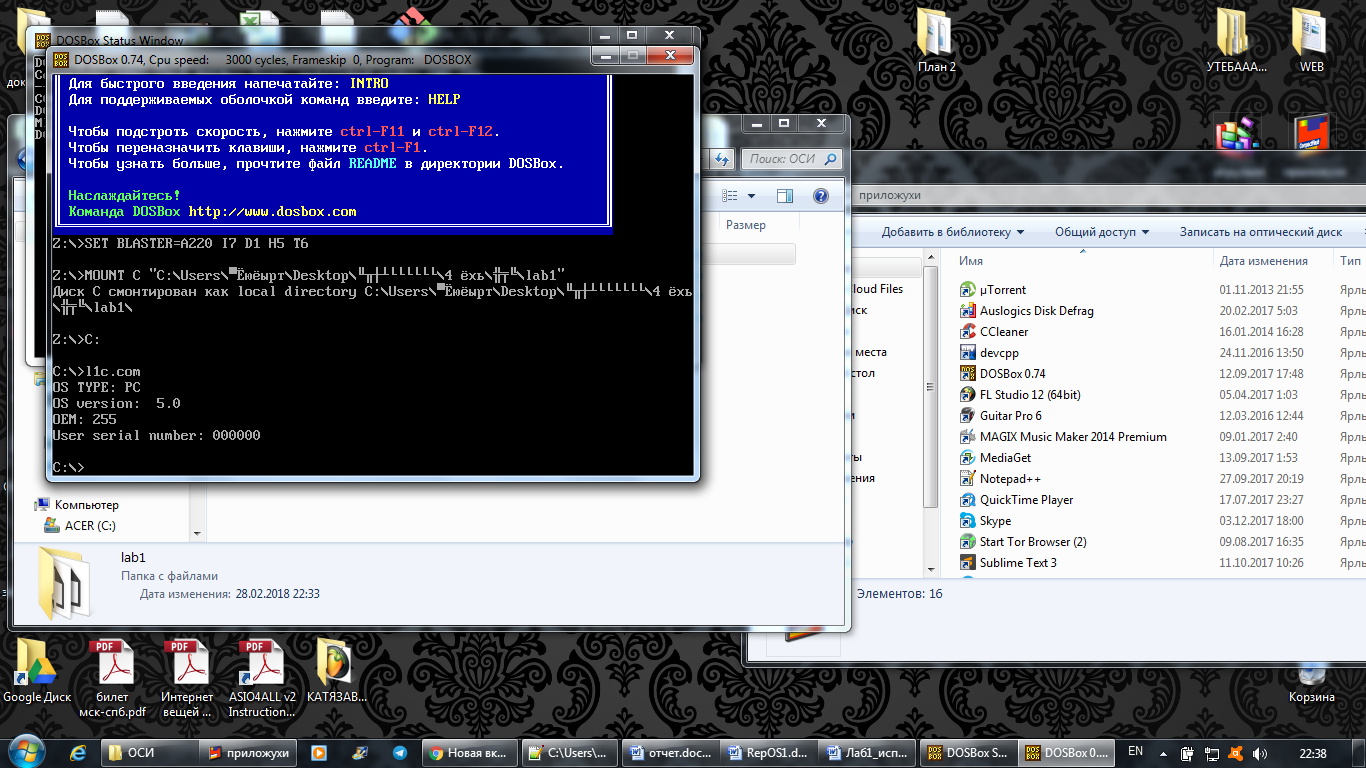
BYTE\_TO\_HEX – переводит число в коды символов 16-ой системы счисления.

WRD\_TO\_HEX – переводит число в строку в 16-ой системе счисления и записывает получившееся, начиная с младшей цифры.

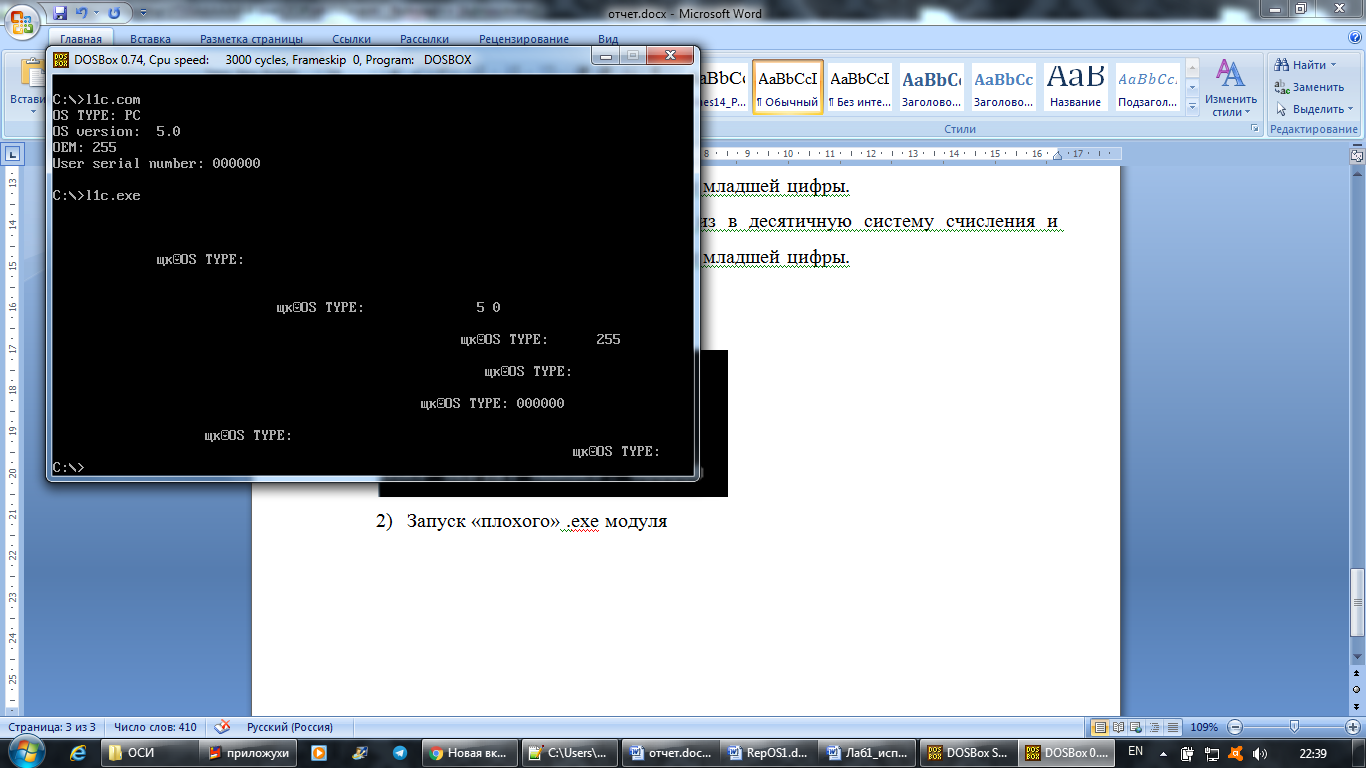
BYTE\_TO\_DEC – переводит байт из в десятичную систему счисления и записывает получившееся, начиная с младшей цифры.

**Ход работы:**

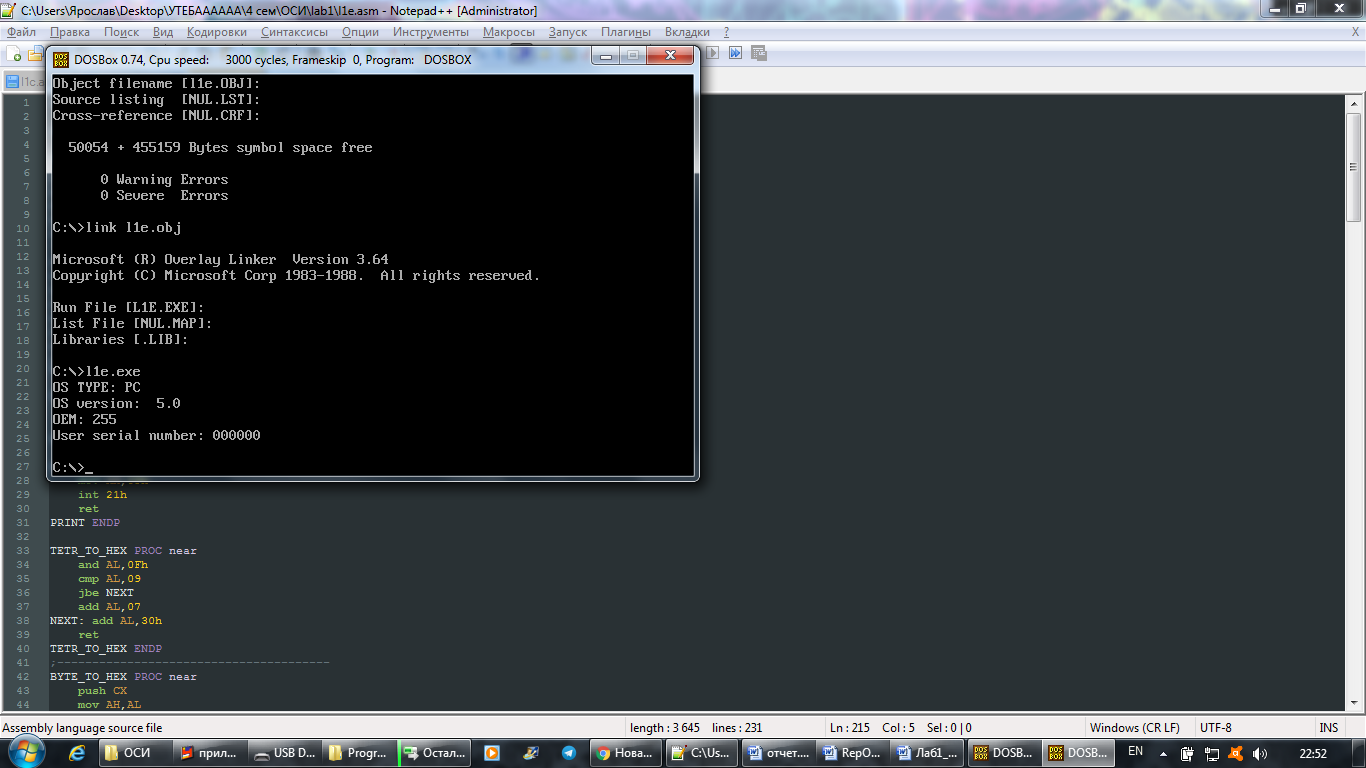
1. Запуск .соm модуля



1. Запуск «плохого» .ехе модуля



1. Запуск «хорошего» .ехе модуля



**Ответы на контрольные вопросы:**

1. **Сколько сегментов должна содержать СОМ-программа? ЕХЕ-программа?**

У СОМ-программы 1 сегмент.

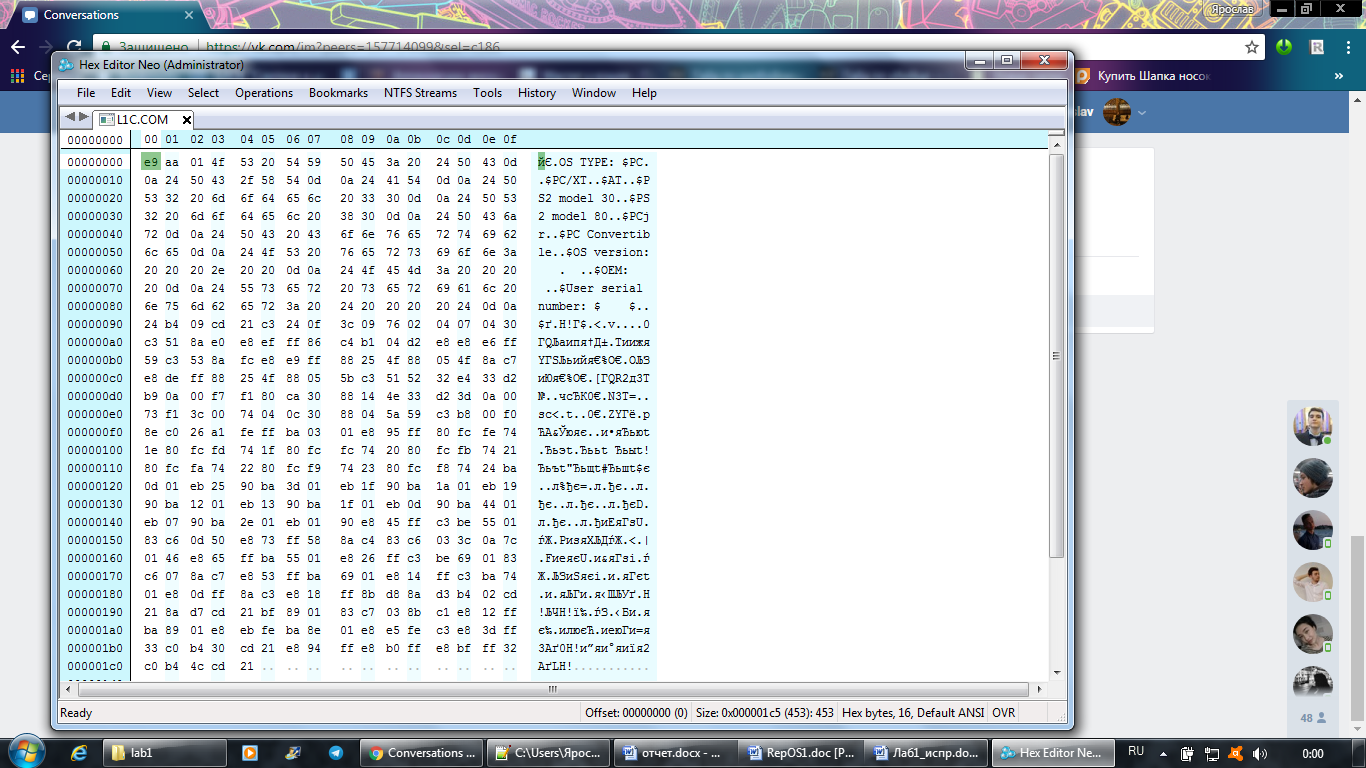
У ЕХЕ-программы отдельные сегменты для данных, кода и стека.

1. **Какие директивы должны обязательно быть в тексте COM программы?**

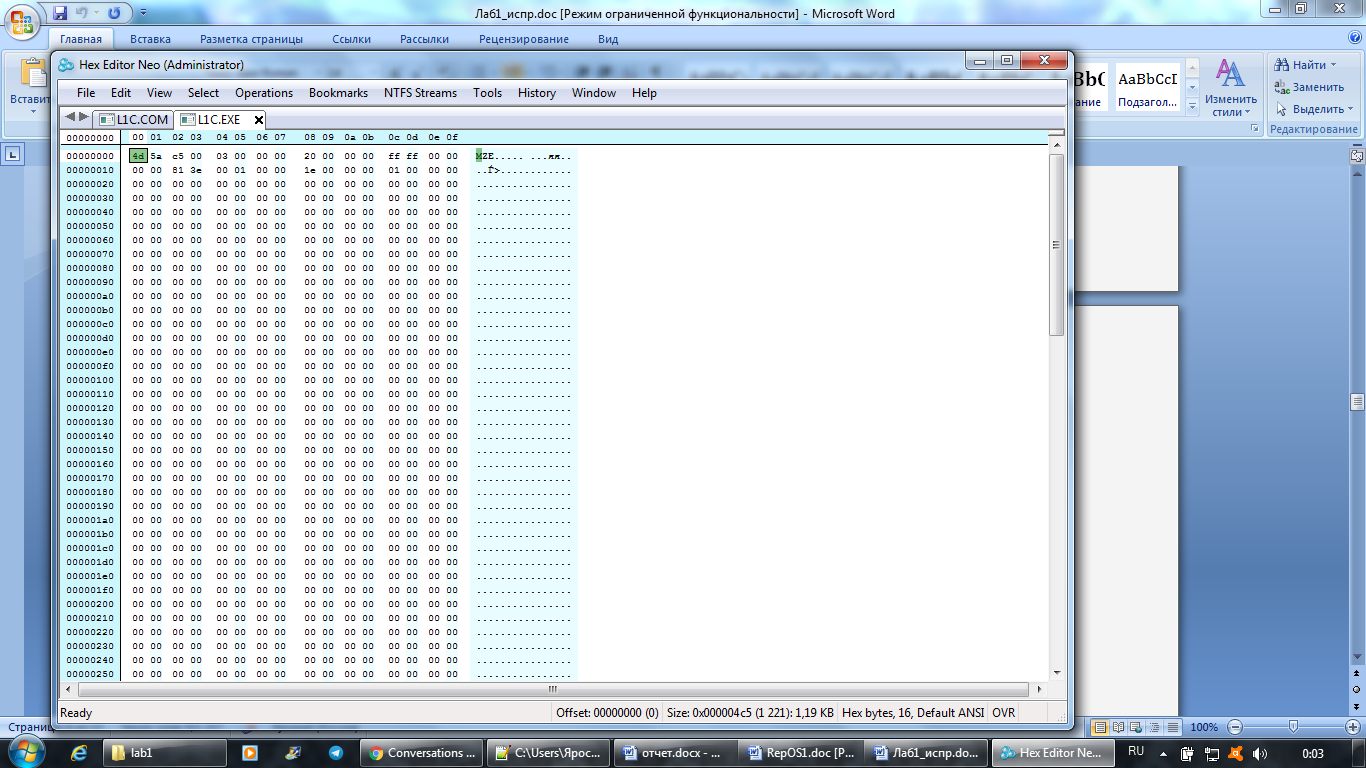
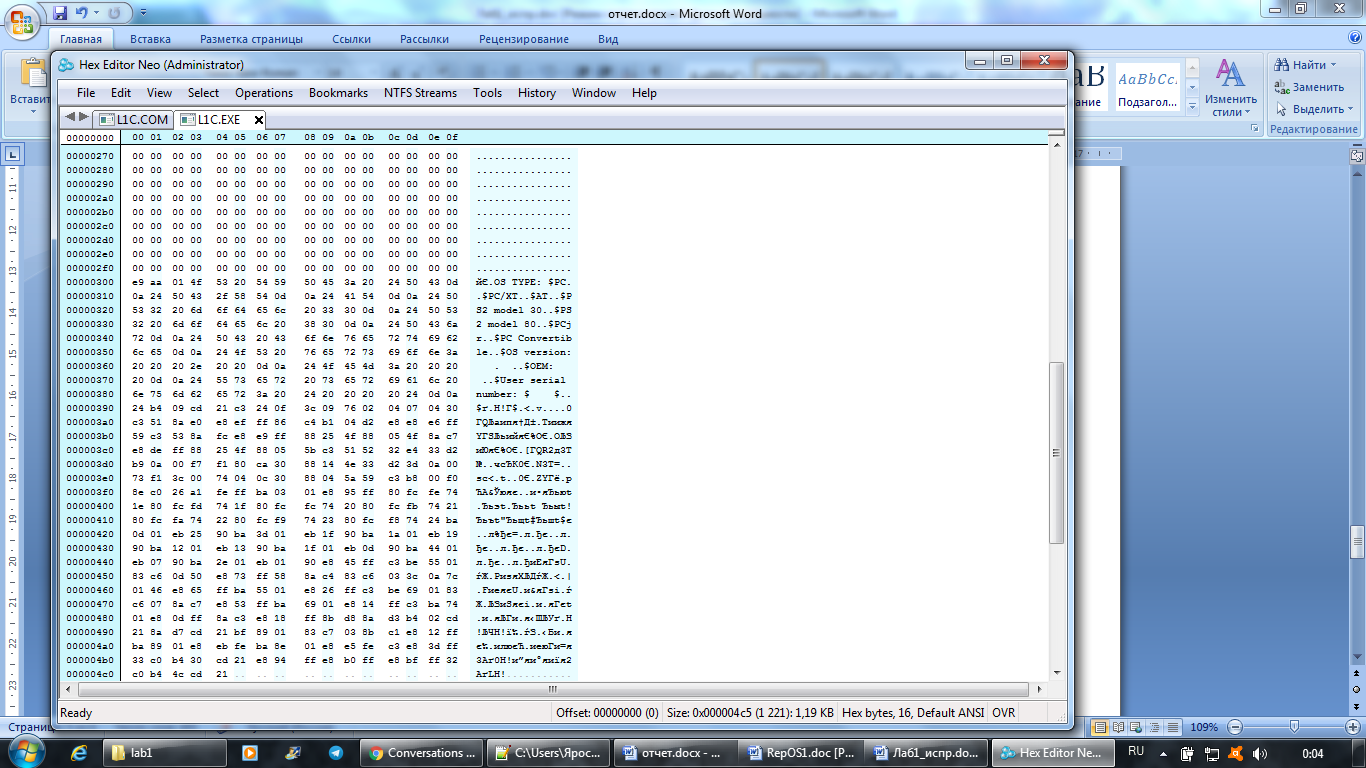
В тексте COM-программы обязательно должна быть директива ORG 100h, т.к. в первых 256 байтах находится PSP.

1. **Все ли форматы команд можно использовать в COM-программе?**

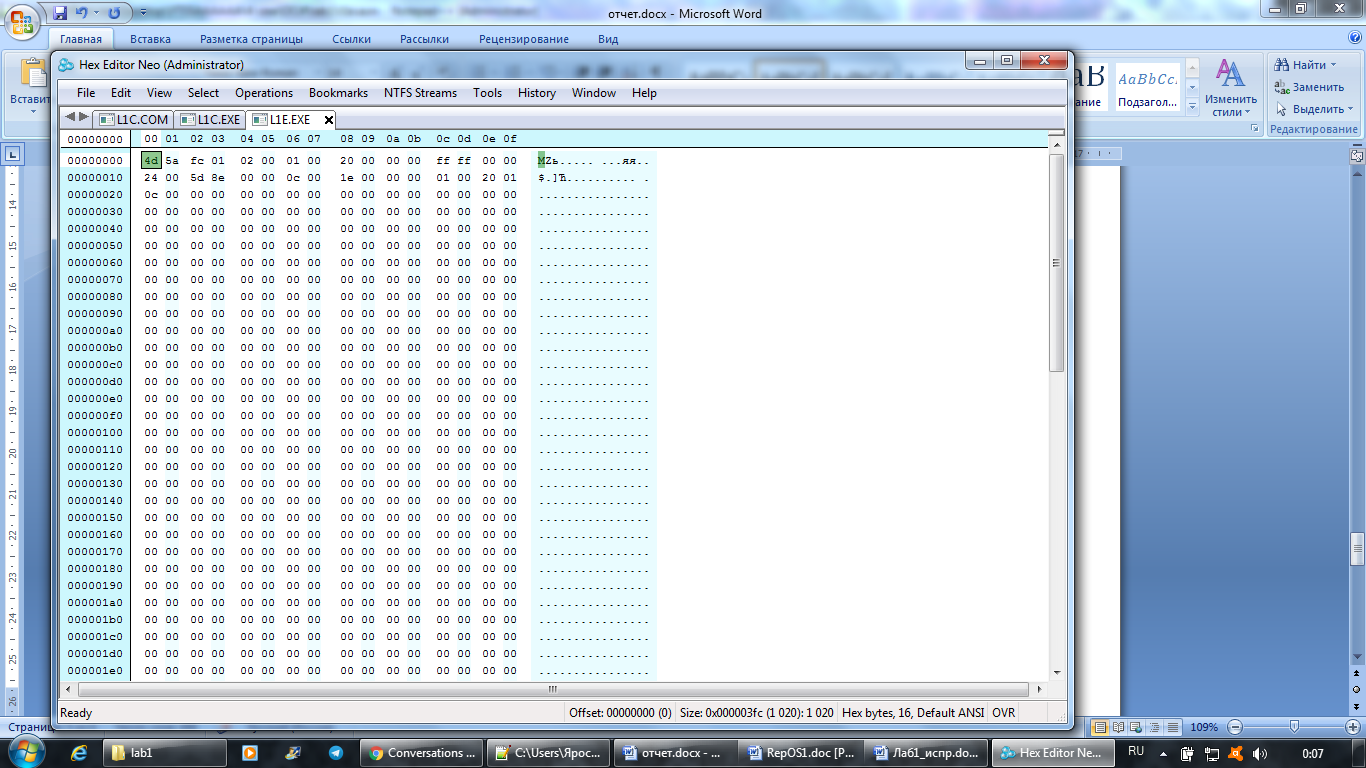
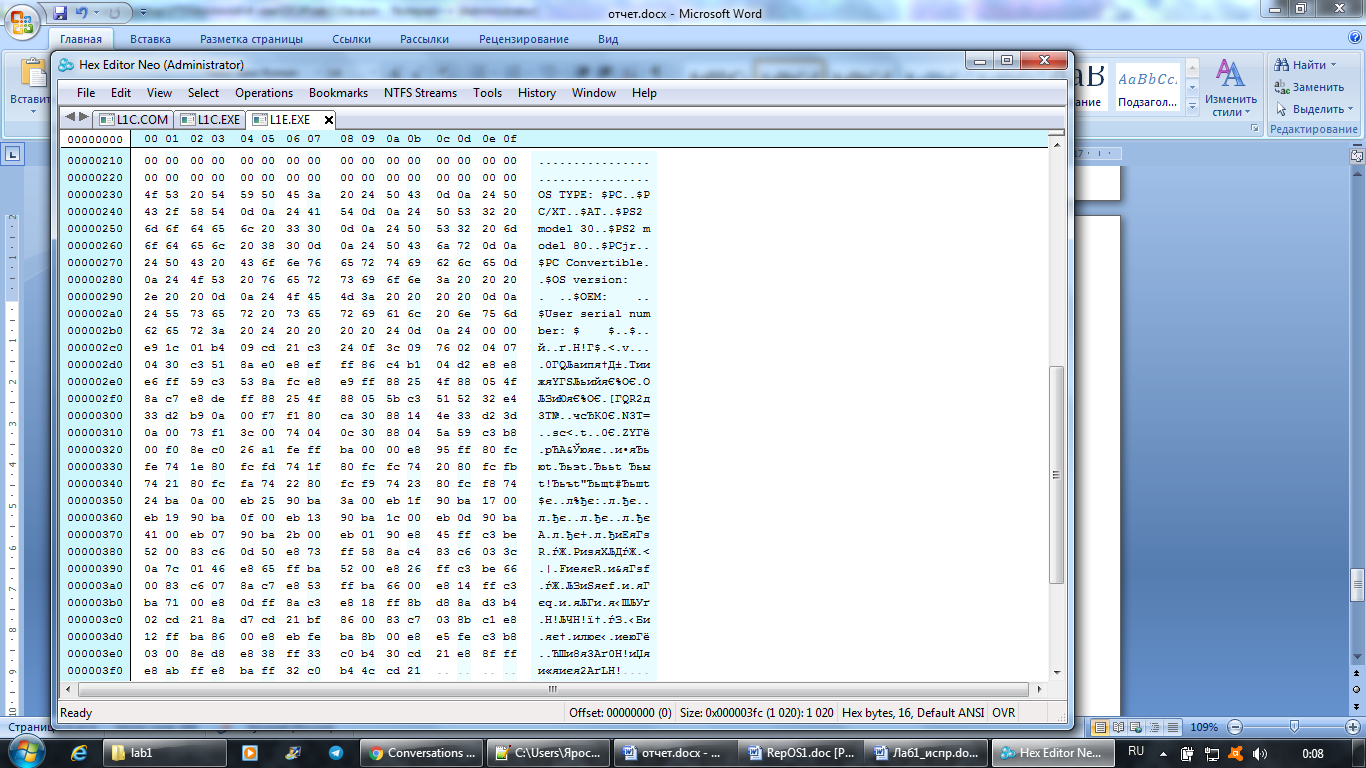
В СОМ-программе нельзя использовать команды вида mov register, segment, т.к. в них используется имеющаяся только в ЕХЕ-файлах relocation table.



Содержание СОМ-модуля.

Содержание «плохого» ЕХЕ-модуля

Содержание «хорошего» ЕХЕ-модуля.

1. **Какова структура файла СОМ? С какого адреса располагается код?**

COM-файл содержит только код и данные. В файле код располагается с нулевого адреса.

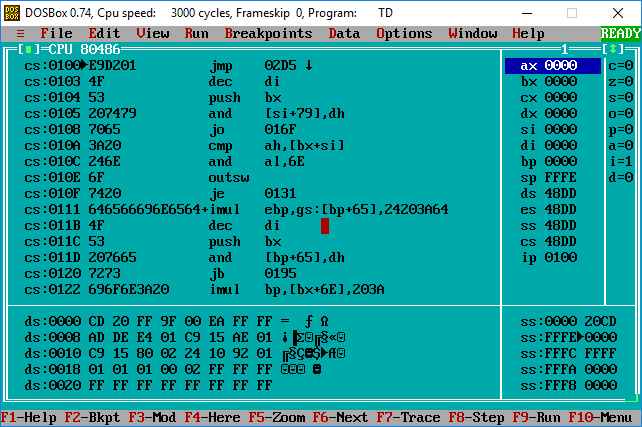
1. **Какова структура файла «плохого» EXE? С какого адреса располагается код? Что располагается с адреса 0?**

В «плохом» EXE файле данные и код содержатся в одном сегменте. Код располагается с адреса 300h. С адреса 0h располагается Relocation Table

1. **Какова структура файла «хорошего» EXE? Чем он отличается от файла «плохого» EXE?**

В отличие от «плохого» EXE, в «хорошем» код, стек и данные выделены в отдельные сегменты. Код начинается с 230h байта.

Для «хорошего» EXE в директиве org 100h нет необходимости, т.к. загрузчик автоматически расположит программу после PSP.



Загрузка COM модуля в основную память

1. **Какой формат загрузки COM модуля? С какого адреса располагается код?**

Код располагается с адреса 100h. После загрузки COM-программы в основную память сегментные регистры указывают на начало PSP.

1. **Что располагается с 0 адреса?**

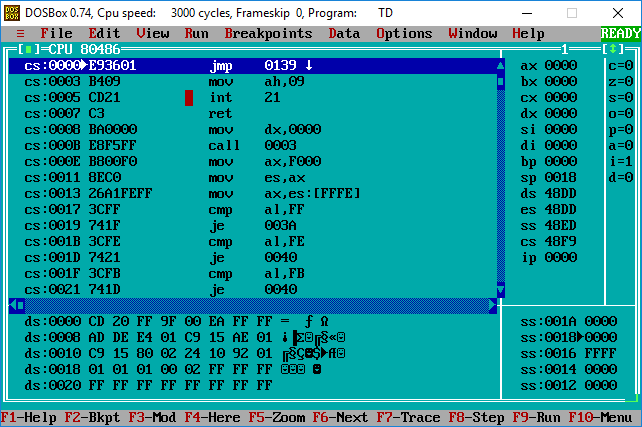
Начало PSP

1. **Какие значения имеют сегментные регистры? На какие области памяти они указывают?**

48DDh, На начало PSP

1. **Как определяется стек? Какую область памяти он занимает? Какие адреса?**

Стек занимает весь сегмент COM-программы, его начало находится в конце сегмента. SS указывает на начало сегмента, а SP=FFFEh – на его конец. Стек может дойти до кода/данных программы при достаточном количестве элементов.



Загрузка «хорошего» EXE модуля в основную память

1. **Как загружается «хороший» EXE? Какие значения имеют сегментные регистры? На что указывают регистры DS и ES?**

DS и ES имеют значение 48DD и указывают на начало PSP, SS=48ED – начало сегмента стека, CS=48F9 – начало сегмента команд.

1. **Как определяется стек?**

Выделением сегмента. В регистры SS и SP записываются адрес начала сегмента стека и его вершины соответственно.

1. **Как определяется точка входа?**

Точка входа в программу определяется с помощью команды END.