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

Студентка гр. 0381 _____ Степанова Е.М.

Преподаватель _____ Губкин А.Ф.

Санкт-Петербург

2022

Цель работы.

Исследование различий в структурах исходных текстов модулей типов .COM и .EXE, структур файлов загрузочных модулей и способов их загрузки в основную память.

Основные теоретические положения.

В работе используются следующие процедуры:

TETR_TO_HEX – переводит в 10-ые цифры в символьный код.

BYTE_TO_HEX – переводит байт в 16-ой системе счисления в символьный код.

WRD_TO_HEX – переводит слово в символьный код.

BYTE_TO_DEC – переводит байт в 10-ую систему счисления.

Данные процедуры взяты из модуля «Общие сведения»

Также были написаны следующие процедуры:

print – выводит сообщение в консоль ms-dos.

pc_type – получает информацию о типе IBM PC, при помощи сравнений с таблицей (Рисунок 1) выдаёт информацию.

versia_info – получает информацию о версии ms-dos при помощи прерывания int 21h функцией 30h.

PC	FF
PC/XT	FE,FB
AT	FC
PS2 модель 30	FA
PS2 модель 50 или 60	FC
PS2 модель 80	F8
PCjr	FD
PC Convertible	F9

Рисунок 1.

В ходе работы также были описаны строки для вывода:

type_PC db 'IBM PC Type: PC', 0dh, 0ah, '$'

type_PC_XT db 'IBM PC Type: PC/XT', 0dh, 0AH, '$'

type_AT db 'IBM PC Type: AT', 0dh, 0ah, '$'

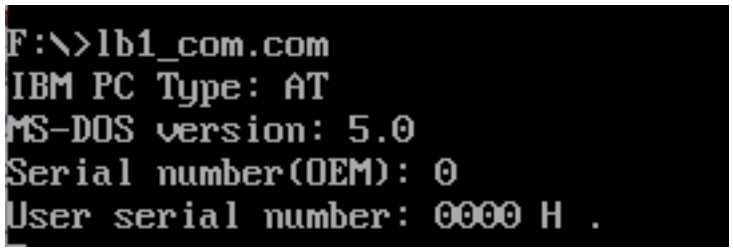
type_PS2_30 db 'IBM PC Type: PS2 model 30', 0dh, 0ah, '$'

```
type_PS2_50_60 db 'IBM PC Type: PS2 model 50/60', 0dh, 0ah, '$'
type_PS2_80 db 'IBM PC Type: PS2 model 80', 0dh, 0ah, '$'
type_PCjr db 'IBM PC Type: PCjr', 0dh, 0ah, '$'
type_PC_Convertible db 'IBM PC Type: PC Convertible', 0dh, 0ah, '$'

version db 'MS-DOS version:  .', 0dh, 0ah,'$'
serial_number db 'Serial number(OEM): .', 0dh,0ah,'$'
user_number db 'User serial number:       H .$'
```

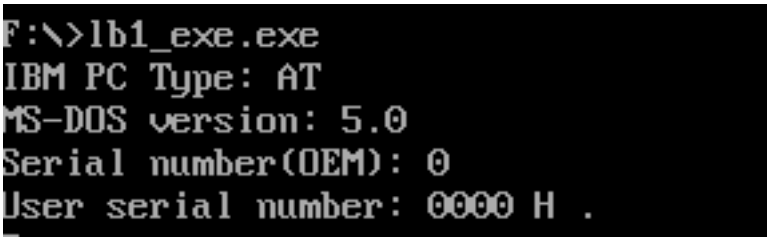
Выполнение работы.

I. Был написан и отлажен исходный .com модуль , определяющий тип PC и версию ms-dos. При запуске данного .com модуля выводится следующая информация:



При создании данного .com модуля был создан «плохая» .exe программа:

II. Был написан текст исходного .exe модуля, который выполняет ту же работу, что и .com модуль, но являет уже “хорошей” .exe программой. Было разделение сегментов кода и данных.



Ответы на вопросы п. 3-6 представлены в Приложении А.

Выводы.

Были исследованы различия в структурах исходных текстов модулей типов .COM и .EXE, структур файлов загрузочных модулей и способов их загрузки в основную память.

ПРИЛОЖЕНИЕ А

ВОПРОСЫ

1. **Отличия исходных текстов .com и .exe программ:**

1) Сколько сегментов должна содержать com-программа?

Только один сегмент, так как стек задается автоматически, а код и данные, находясь вместе, не разделяются на разные сегменты.

2) Сколько сегментов должна содержать exe-программа?

Exe – программа должна содержать не менее одного сегмента. Сегменты данных, стека и кода описываются отдельно (сегмент стека можно не задавать, ms-dos автоматически выделит место под стек).

3) *Какие директивы должны обязательно быть в тексте com-программы*

Директива org 100h, которая обеспечивает смещение в 256 байт, чтобы не попасть в область PSP и директива assume, которая позволяет указать для сегментов кода и данных на один общий сегмент в программе.

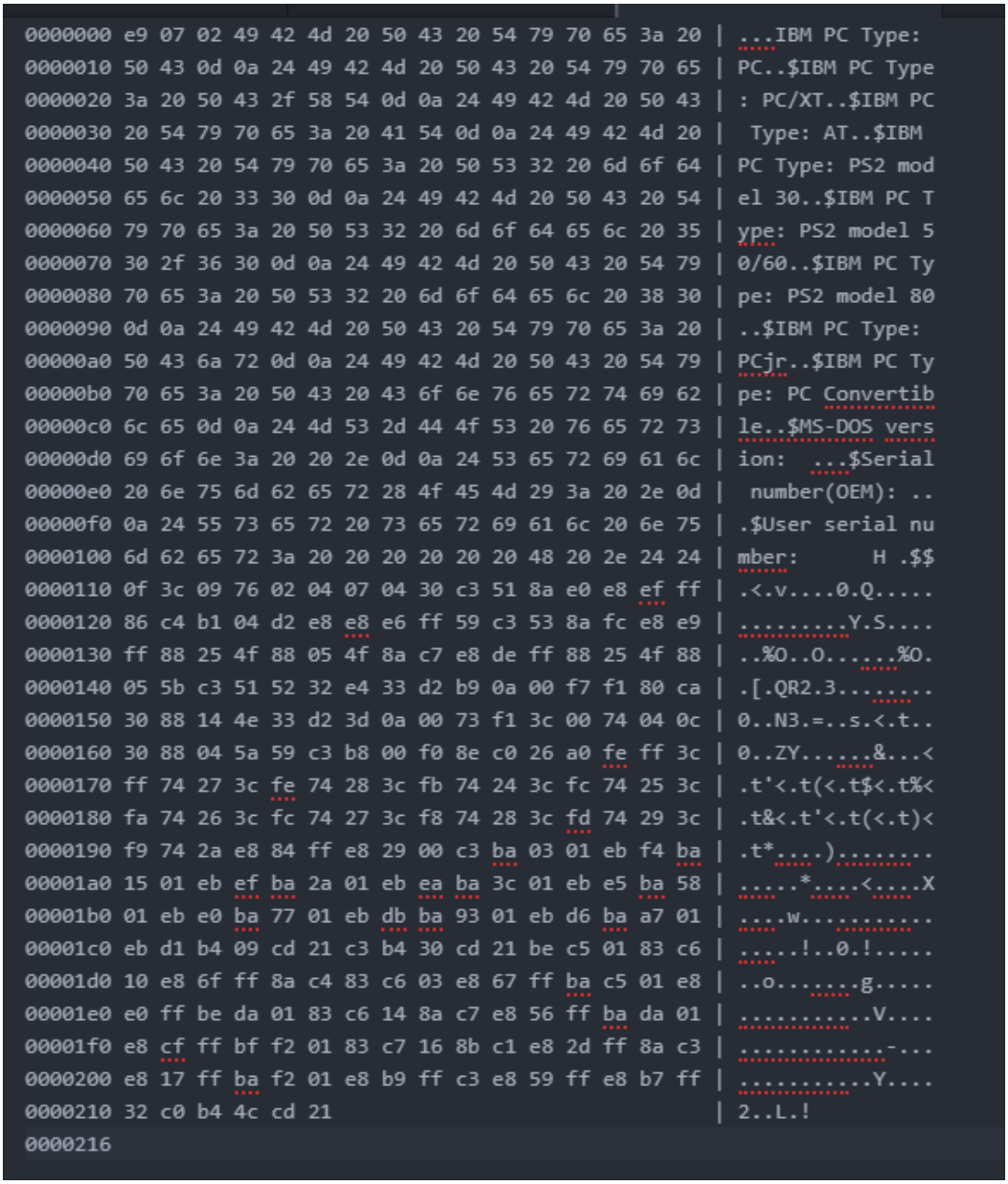
4) Все ли форматы команд можно использовать в com-программе?

Не могут использоваться команды с указанием сегментов, так как отсутствует таблица настроек, по которой осуществляется поиск абсолютных адресов сегментов(например mov или seg).

2. **Отличия форматов файлов .com и .exe модулей:**

1) Какова структура файла com? С какого адреса располагается код?

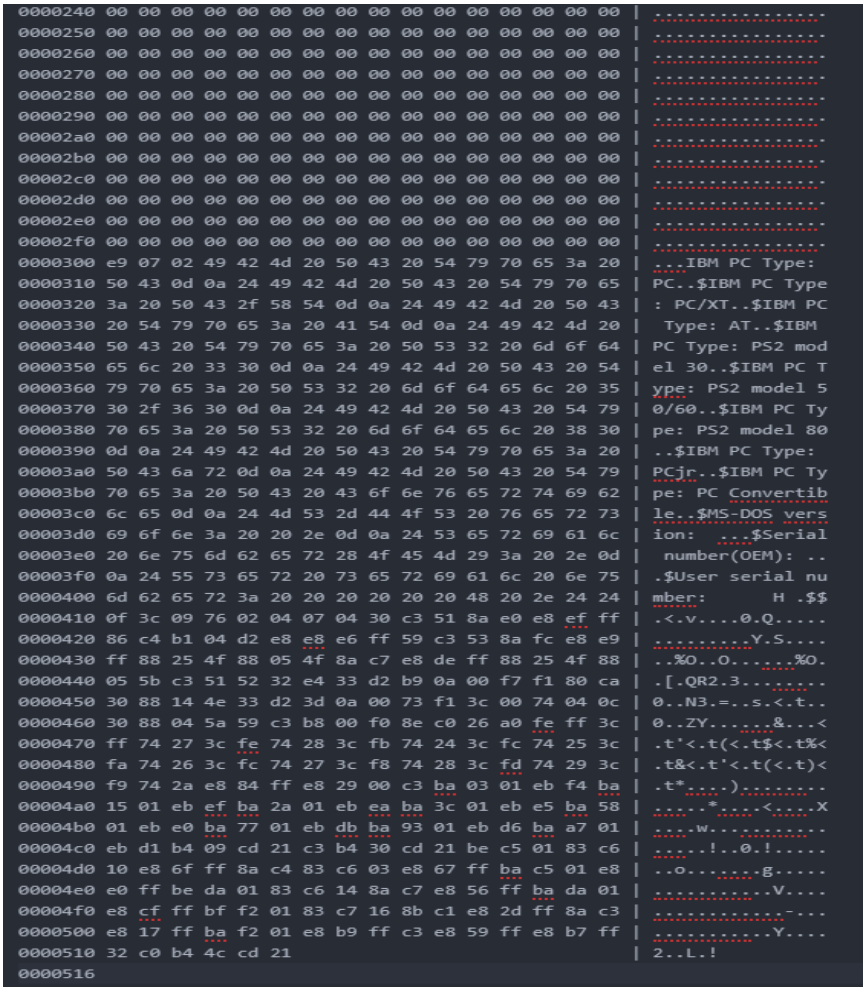
Файл com состоит из одного сегмента, который включает в себя сегменты кода и данных. Файл ограничен размером одного сегмента и не превышает 64Кб. Сегмент стека генерируется автоматически. Код начинается с адреса 0h,но при загрузке модуля устанавливает смещение в 100h.



2) *Какова структура файла «плохого» exe? С какого адреса располагается код? Что располагается с адреса 0?*

Код и данные находятся в одном сегменте, что является ошибкой для .exe, так как эти сегменты должны быть разделены. Код начинается с адреса 300h, а с адреса 0h располагается заголовок модуля. Символы mz указывают на то, что это 16-битный формат исполняемого файла с расширением .exe.

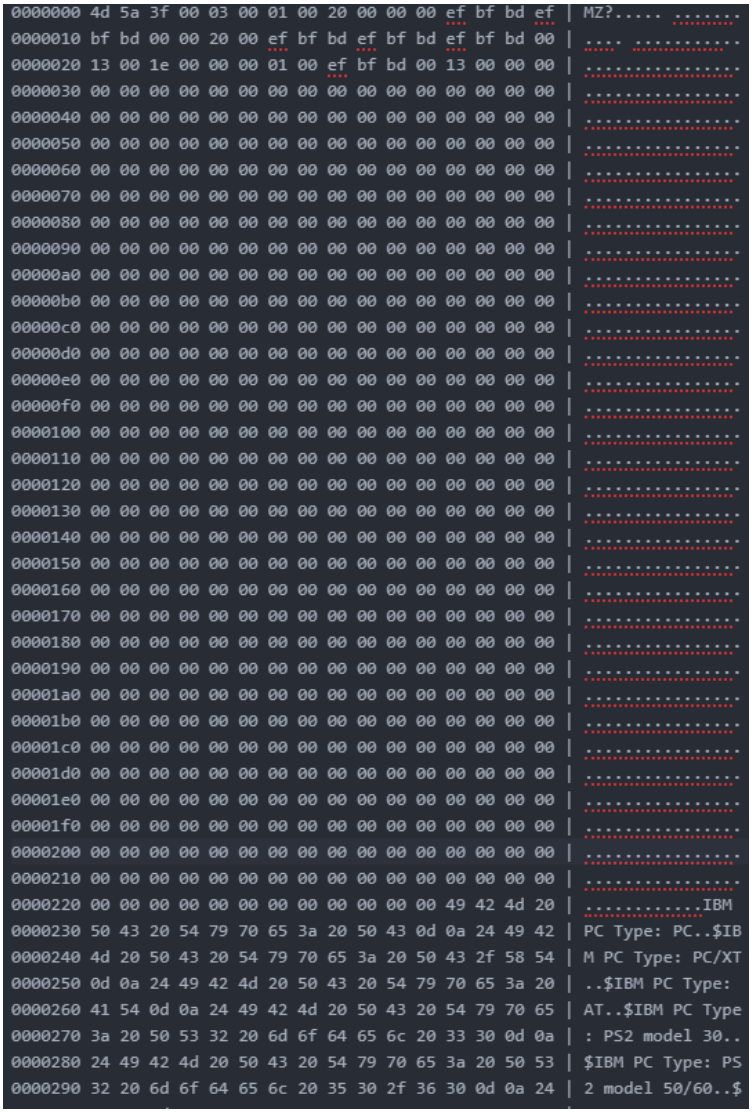
```
0000000 4d 5a 16 01 03 00 00 00 20 00 00 00 ff ff 00 00 | MZ...... ........
0000010 00 00 17 bc 00 01 00 00 1e 00 00 00 01 00 00 00 | ................
0000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00000a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00000b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00000c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00000d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00000e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00000f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00001a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00001b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00001c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00001d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00001e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00001f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
```

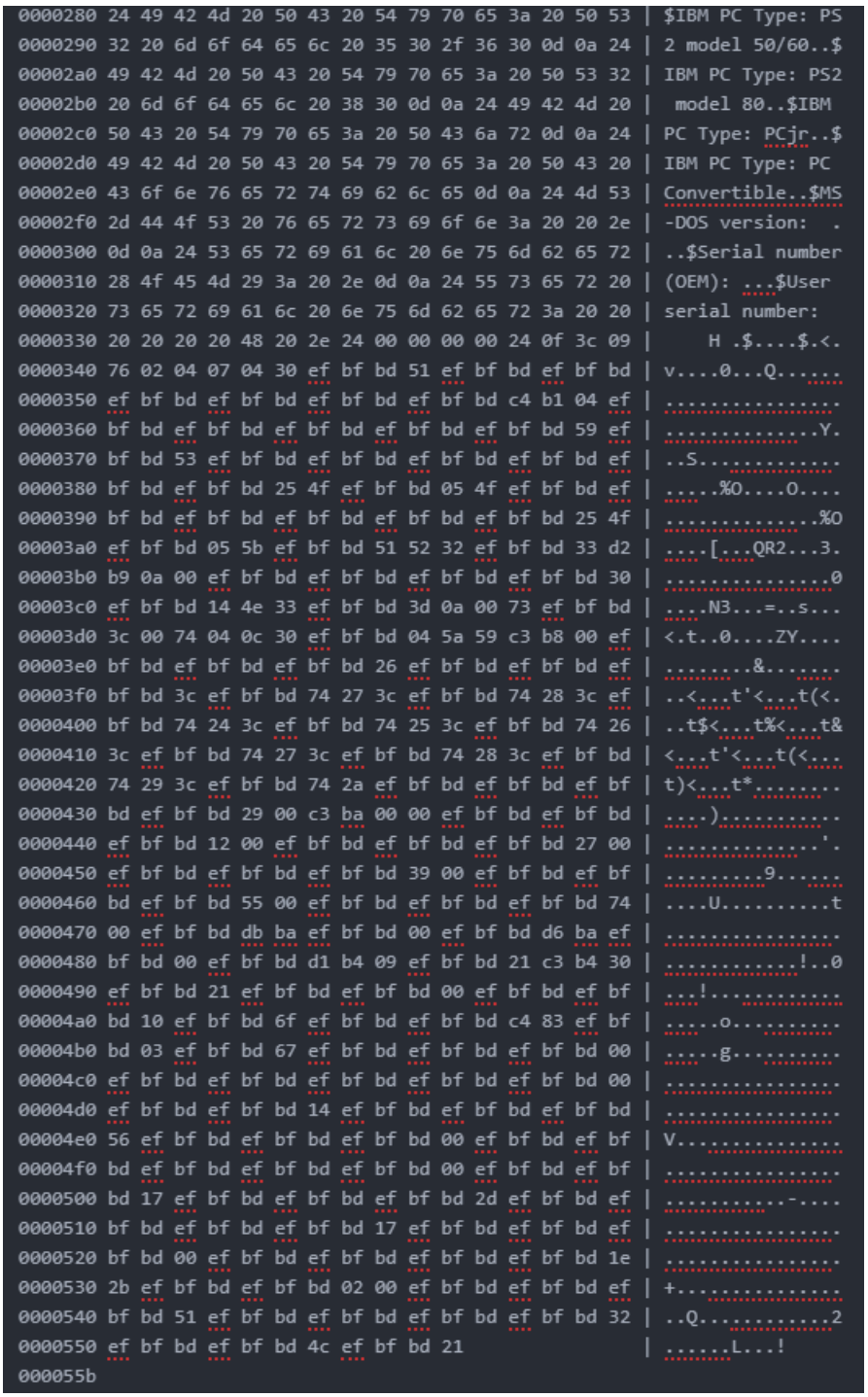


```
0000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000270 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000280 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000290 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00002a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00002b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00002c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00002d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00002e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
00002f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000300 e9 07 02 49 42 4d 20 50 43 20 54 79 70 65 3a 20 | ...IBM PC Type:
0000310 50 43 0d 0a 24 49 42 4d 20 50 43 20 54 79 70 65 | PC..$IBM PC Type
0000320 3a 20 50 43 2f 58 54 0d 0a 24 49 42 4d 20 50 43 | : PC/XT..$IBM PC
0000330 20 54 79 70 65 3a 20 41 54 0d 0a 24 49 42 4d 20 |  Type: AT..$IBM
0000340 50 43 20 54 79 70 65 3a 20 50 53 32 20 6d 6f 64 | PC Type: PS2 mod
0000350 65 6c 20 33 30 0d 0a 24 49 42 4d 20 50 43 20 54 | el 30..$IBM PC T
0000360 79 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c 20 35 | ype: PS2 model 5
0000370 30 2f 36 30 0d 0a 24 49 42 4d 20 50 43 20 54 79 | 0/60..$IBM PC Ty
0000380 70 65 3a 20 50 53 32 20 6d 6f 64 65 6c 20 38 30 | pe: PS2 model 80
0000390 0d 0a 24 49 42 4d 20 50 43 20 54 79 70 65 3a 20 | ..$IBM PC Type:
00003a0 50 43 6a 72 0d 0a 24 49 42 4d 20 50 43 20 54 79 | PCjr..$IBM PC Ty
00003b0 70 65 3a 20 50 43 20 43 6f 6e 76 65 72 74 69 62 | pe: PC Convertib
00003c0 6c 65 0d 0a 24 4d 53 2d 44 4f 53 20 76 65 72 73 | le..$MS-DOS vers
00003d0 69 6f 6e 3a 20 20 2e 0d 0a 24 53 65 72 69 61 6c | ion:  ....$Serial
00003e0 20 6e 75 6d 62 65 72 28 4f 45 4d 29 3a 20 2e 0d |  number(OEM): ..
00003f0 0a 24 55 73 65 72 20 73 65 72 69 61 6c 20 6e 75 | .$User serial nu
0000400 6d 62 65 72 3a 20 20 20 20 20 20 48 20 2e 24 24 | mber:      H .$$
0000410 0f 3c 09 76 02 04 07 04 30 c3 51 8a e0 e8 ef ff | .<.v....0.Q.....
0000420 86 c4 b1 04 d2 e8 e8 e6 ff 59 c3 53 8a fc e8 e9 | .........Y.S....
0000430 ff 88 25 4f 88 05 4f 8a c7 e8 de ff 88 25 4f 88 | ..%O..O......%O.
0000440 05 5b c3 51 52 32 e4 33 d2 b9 0a 00 f7 f1 80 ca | .[.QR2.3........
0000450 30 88 14 4e 33 d2 3d 0a 00 73 f1 3c 00 74 04 0c | 0..N3.=..s.<.t..
0000460 30 88 04 5a 59 c3 b8 00 f0 8e c0 26 a0 fe ff 3c | 0..ZY......&...<
0000470 ff 74 27 3c fe 74 28 3c fb 74 24 3c fc 74 25 3c | .t'<.t(<.t$<.t%<
0000480 fa 74 26 3c fc 74 27 3c f8 74 28 3c fd 74 29 3c | .t&<.t'<.t(<.t)<
0000490 f9 74 2a e8 84 ff e8 29 00 c3 ba 03 01 eb f4 ba | .t*.....).......
00004a0 15 01 eb ef ba 2a 01 eb ea ba 3c 01 eb e5 ba 58 | .....*....<....X
00004b0 01 eb e0 ba 77 01 eb db ba 93 01 eb d6 ba a7 01 | ....w...........
00004c0 eb d1 b4 09 cd 21 c3 b4 30 cd 21 be c5 01 83 c6 | .....!..0.!.....
00004d0 10 e8 6f ff 8a c4 83 c6 03 e8 67 ff ba c5 01 e8 | ..o.......g.....
00004e0 e0 ff be da 01 83 c6 14 8a c7 e8 56 ff ba da 01 | ..........V.....
00004f0 e8 cf ff bf f2 01 83 c7 16 8b c1 e8 2d ff 8a c3 | ............-...
0000500 e8 17 ff ba f2 01 e8 b9 ff c3 e8 59 ff e8 b7 ff | ...........Y....
0000510 32 c0 b4 4c cd 21                               | 2..L.!
0000516
```

3) Какова структура файла «хорошего» exe? Чем он отличается от файла «плохого» exe?

В exe-программе код, данные и стек – отдельные сегменты. exe-фа л имеет заголовок, которы используется при его загрузке. Заголовок состоит из форматированно части, содержаще сигнатуру и данные, необходимые для загрузки exe-файла, и таблицы для настро ки адресов. В отличии от «плохого» exe в «хорошем» exe присутствуют три сегмента: сегмент кода, сегмент данных и сегмент стека, а «плохо » exe содержит один сегмент, совмещающи код и данные. Также в «плохом» exe адресация кода начинается с 300h, так как он получается из .com фа ла, в котором изначально сегмент кода смещён на 100h, а при создании «плохого» exe к этому смещению добавляется размер заголовка.



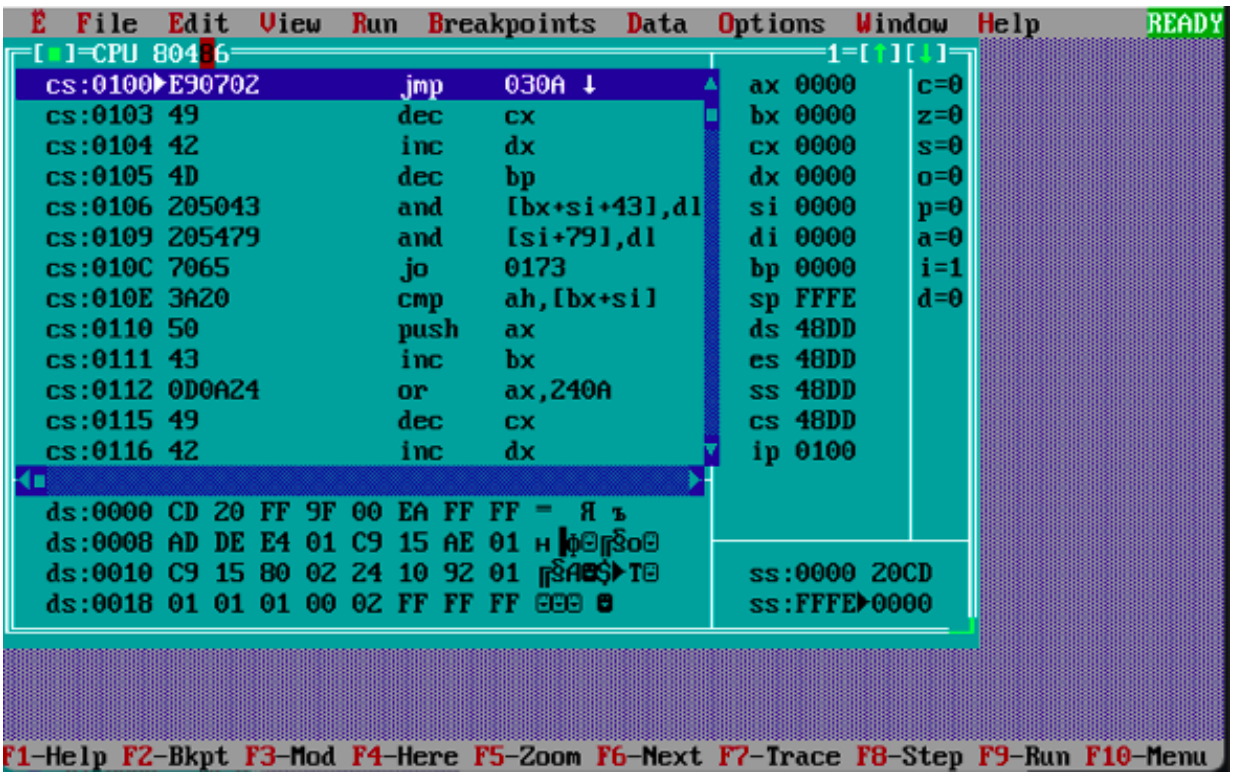


3. Загрузка .com модуля в основную память:

1) *Какой формат загрузки модуля com? С какого адреса располагается код?*

Определяется сегментный адрес участка ОП, у которого достаточно места для загрузки программы, образ com-файла считывается с диска и помещается в память, начиная с PSP:0100h. После загрузки образа com-

программы сегментные регистры CS,SS,DS и ES указывают на PSP, SP указывает на конец сегмента PSP, слово 00H помещено в стек, IP содержит 100H в результат команды JMP PSP:100h.



2) *Что располагается с адреса 0h?*

Программный сегмент PSP, размером 256 байт, зарезервированный ОС.

3) *Какие значения имеют сегментные регистры? На какие области памяти ониуказывают?*

Сегментные регистры CS, DS, ES и SS указывают на PSP и имеют значение 48DD.

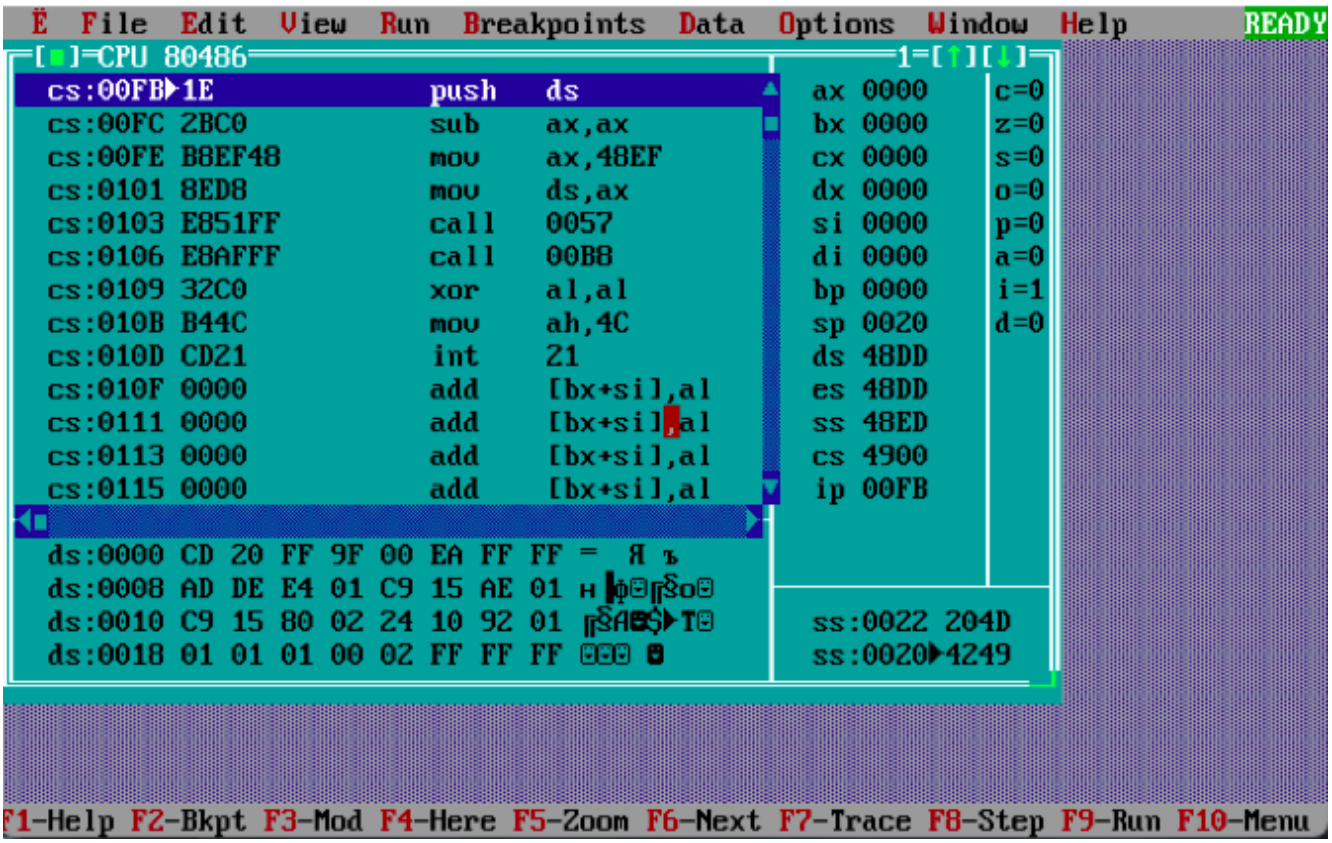
4) *Как определяется стек? Какую область памяти он занимает? Какие адреса?*

Стек генерируется автоматически при создании com-программы. SS – на начало (0h), регистр SP – на конец стека (FFFEh). Адрес стека расположен от 0h – FFFEh.

4. Загрузка «хорошего» .exe модуля в основную память:

1) *Как загружается «хороший» exe? Какие значения имеют сегментные регистры?*

Exe-файл загружается с адреса PSP:0100h. В процессе загрузки считывается информация заголовка в начале файла и выполняется перемещение адресов сегментов, то есть DS и ES устанавливаются на начало сегмента PSP, SS – на начало сегмента стека, CS – на начало сегмента команд. В IP загружается смещение точки входа в программу, которая берется из метки после директивы END. Причем дополнительный программный сегмент (PSP) присутствует в каждом exe-файле.



2) *На что указывают регистры DS и ES?*

Они указывают на начало сегмента PSP.

3) *Как определяется стек?*

Стек определяется директивой stack, после которой задается размер стека. При исполнении регистр SS указывает на начало этого стека, а SP – на конец стека.

4) *Как определяется точка входа?*

При помощи директивы END.