БЕЛОРУССКИЙ  ГОСУДАРСТВЕННЫЙ  УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Кафедра технологий программирования

**﻿﻿﻿Защита программ от взлома**

Домашнее задание

Шибко Татьяны Александровны

студентки 4 курса 12 группы

специальность "прикладная информатика"

|  |  |
| --- | --- |
| Дата окончания работы 06.10.2024 г. | Преподаватель: |
|  | Доцент К.А. Зубович |

Минск, 2024

# Вопрос 0.

Перечитайте название Лекции, сформулируйте все известные Вам постулаты из области БИС и укажите, сделав обоснование, какое из них наиболее соответствует названию Лекции.

**Ответ:**

*Постулаты в области БИС.*

Постулат 1. Не существует абсолютных методов защиты данных.

Постулат 2. Построение системы для защиты данных необходимо тогда и только тогда, когда стоимость разработки и внедрения данной системы меньше стоимости вреда, который может быть нанесен заказчику системы из-за её отсутствия.

Постулат 3. Тем, что нельзя измерить, невозможно управлять.

Постулат 4. Чем ближе к аппаратному уровень реализации система защиты данных, тем более надежна система защиты данных при прочих равных условиях.

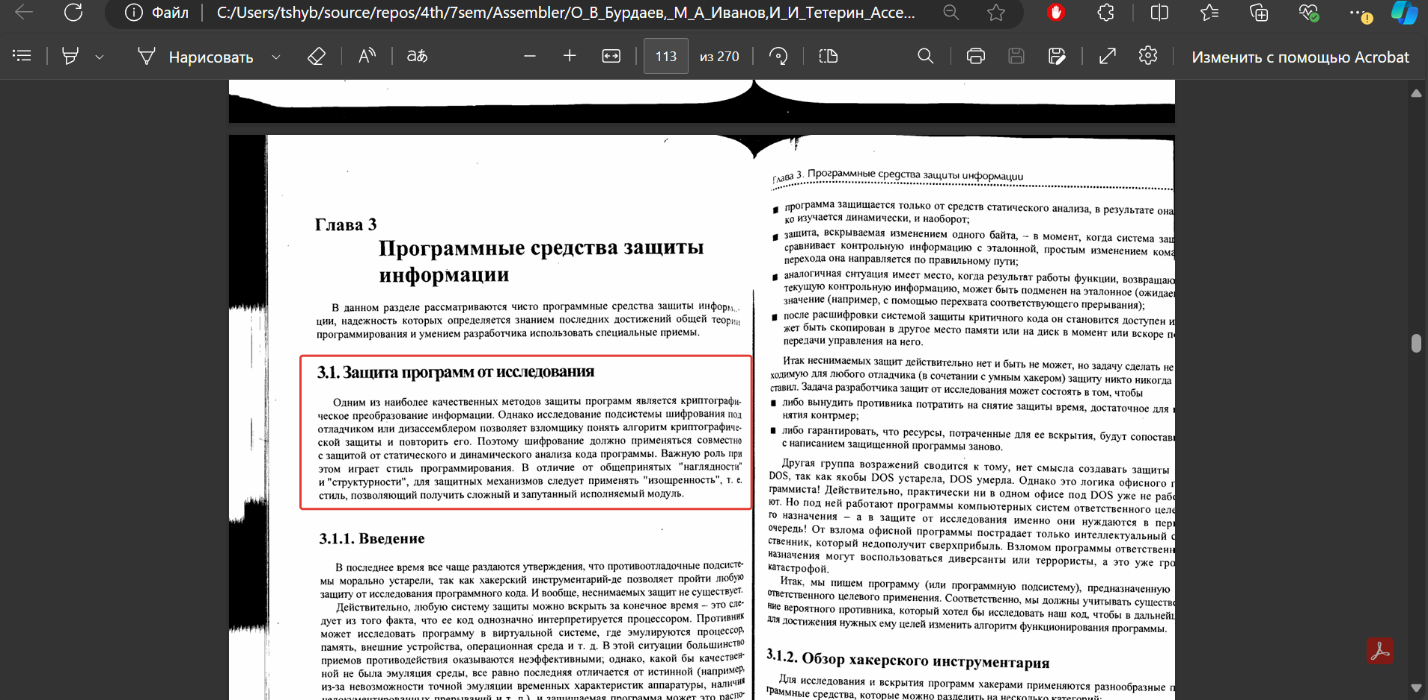
Я выбрала первый постулат, потому что:

* Уязвимости всегда присутствуют: никакая система защиты не может обеспечить полную безопасность.
* Эволюция угроз: методы взлома постоянно развиваются, что требует адаптации систем защиты.
* Многоуровневая защита: осознание того, что абсолютной защиты не существует, побуждает использовать комплексные меры безопасности.

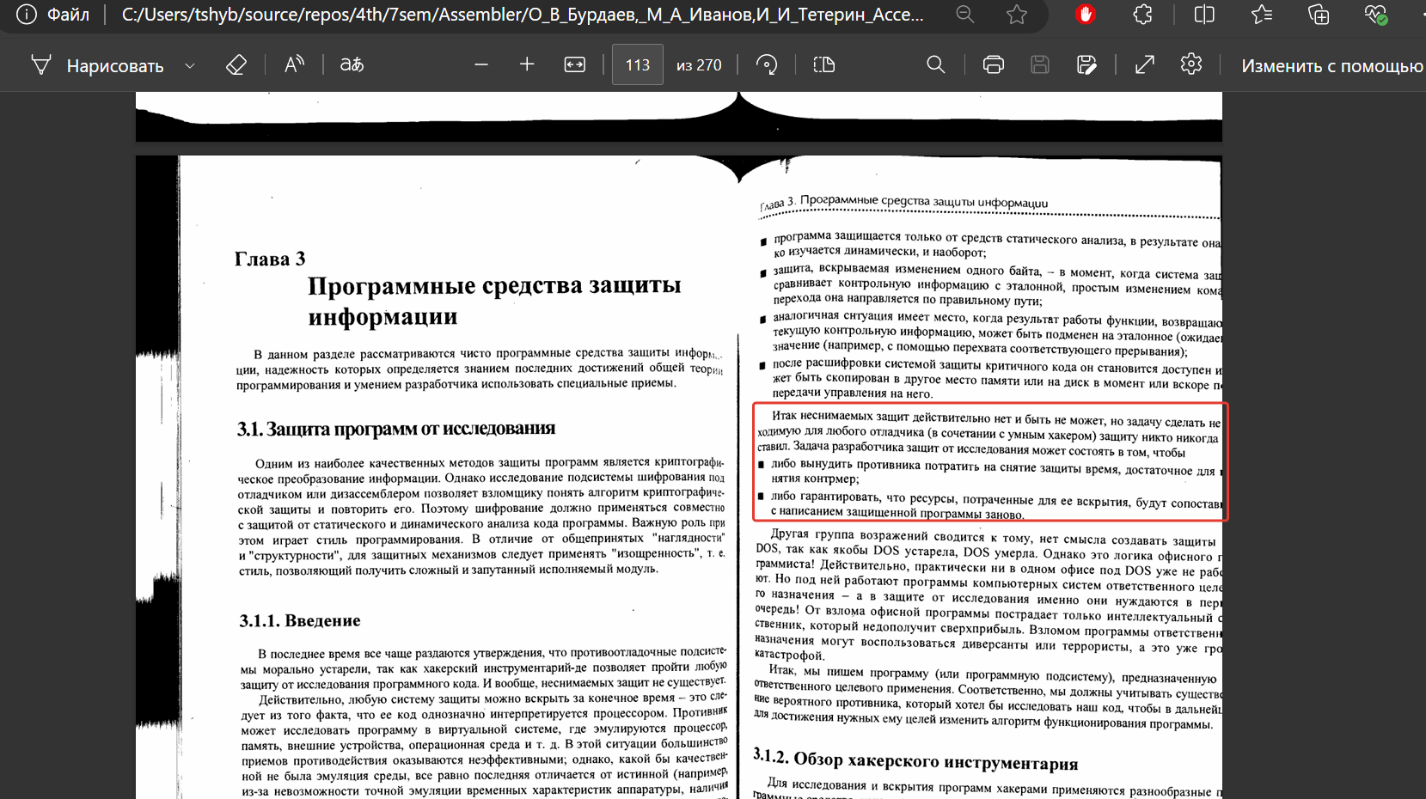
# Вопрос 1.

На каких страницах книги записаны набранные мелким шрифтом абзацы.

**Ответ:** Первый абзац – страница 226



2 абзац – страница 228



# Вопрос 2.

Дать толкование Crack-а на английском языке, взяв его из английской Wikipedia.

**Ответ:** *Crack is a*[*Unix*](https://en.wikipedia.org/wiki/Unix)[*password cracking*](https://en.wikipedia.org/wiki/Password_cracking)*program designed to allow*[*system administrators*](https://en.wikipedia.org/wiki/System_administrator)*to locate users who may have weak*[*passwords*](https://en.wikipedia.org/wiki/Password)*vulnerable to a*[*dictionary attack*](https://en.wikipedia.org/wiki/Dictionary_attack)*. Crack was the first standalone password cracker for Unix systems and the first to introduce programmable dictionary generation as well.*

*Safe-cracking is the process of opening a*[*safe*](https://en.wikipedia.org/wiki/Safe)*without either the combination or the key.*

# Вопрос 3:

КАК ТАК? Записать пример из книги про ассемблер.

**Ответ:**

=== Пример 3. 6. ====================================

cs:

pushf

pop ax

pushf

pop bx

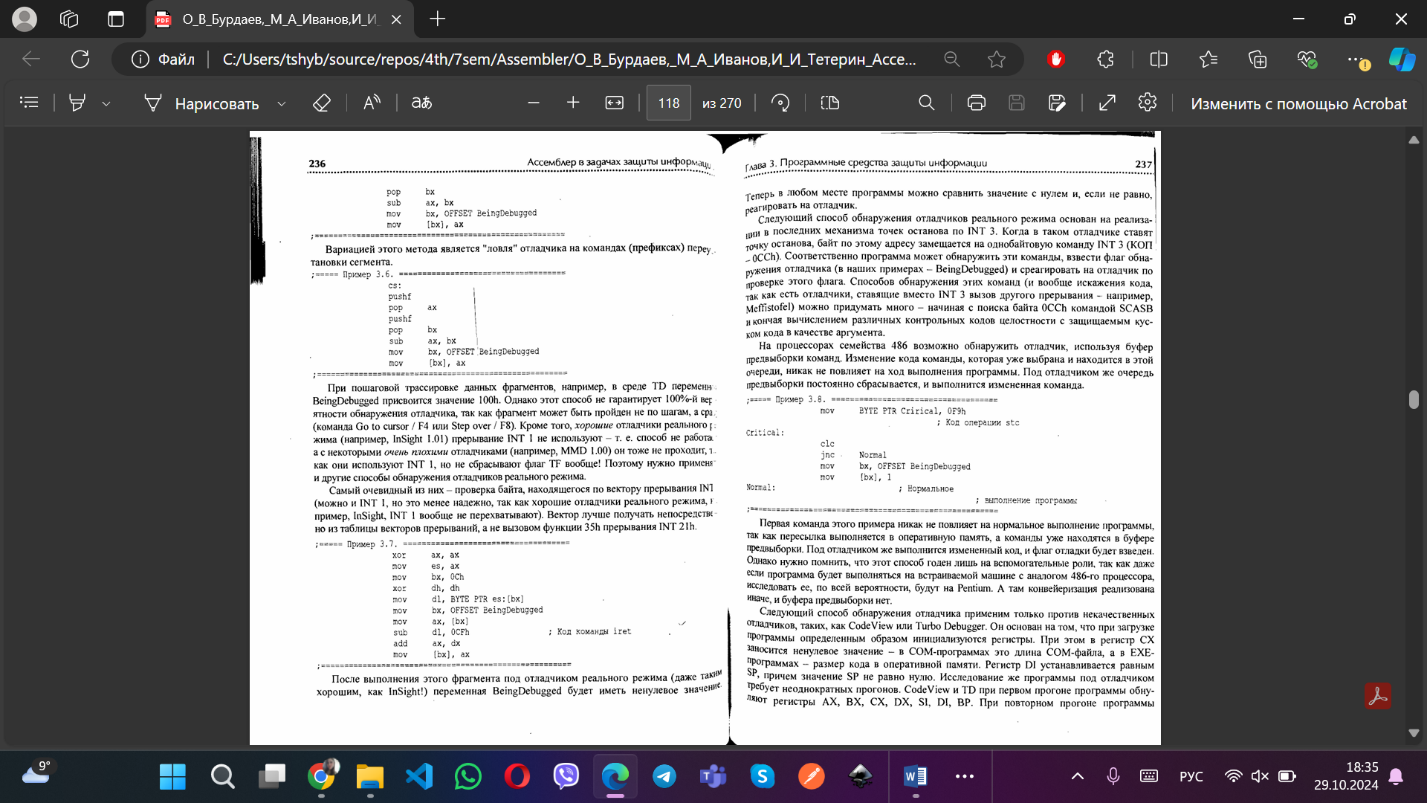
sub ax, bx

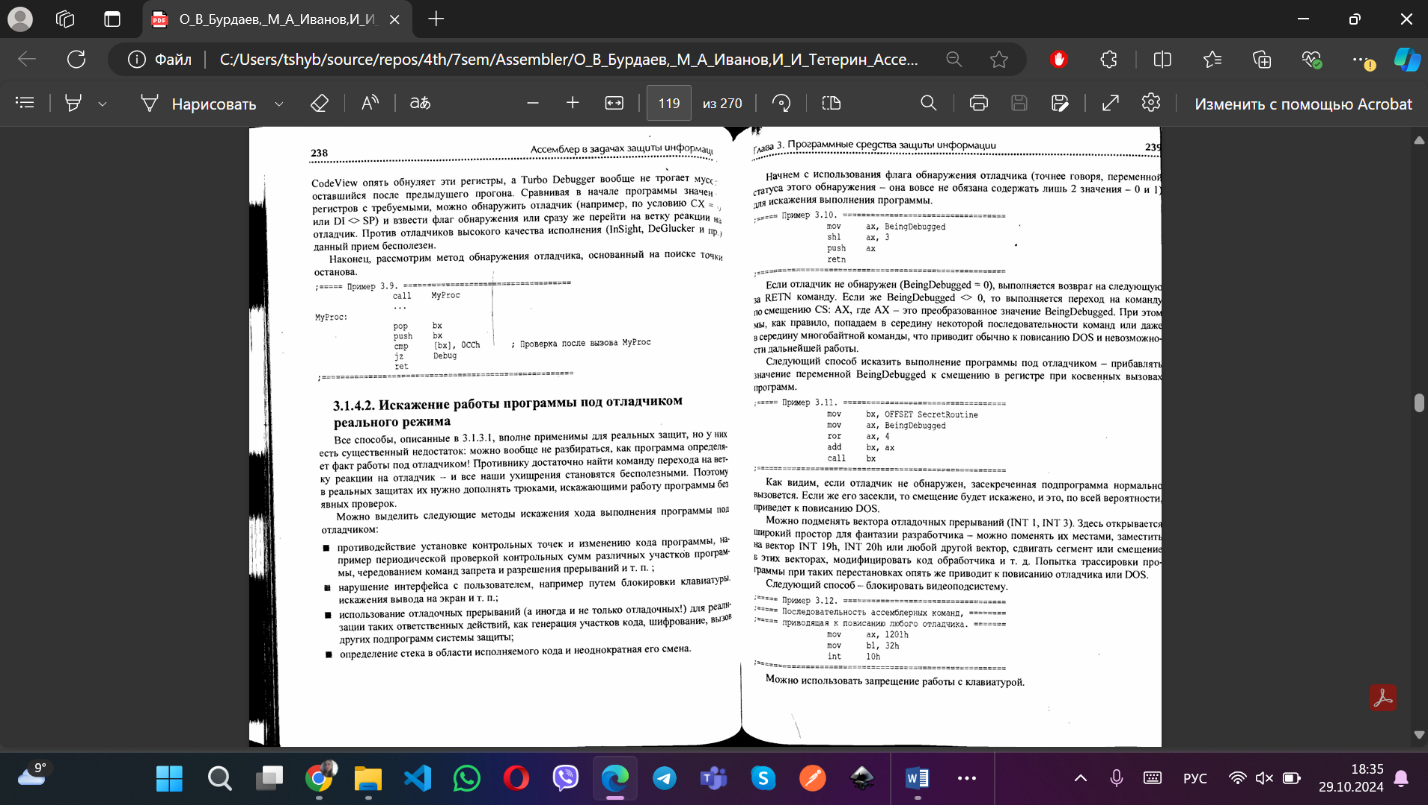
mov bx, OFFSET BeingDebugged

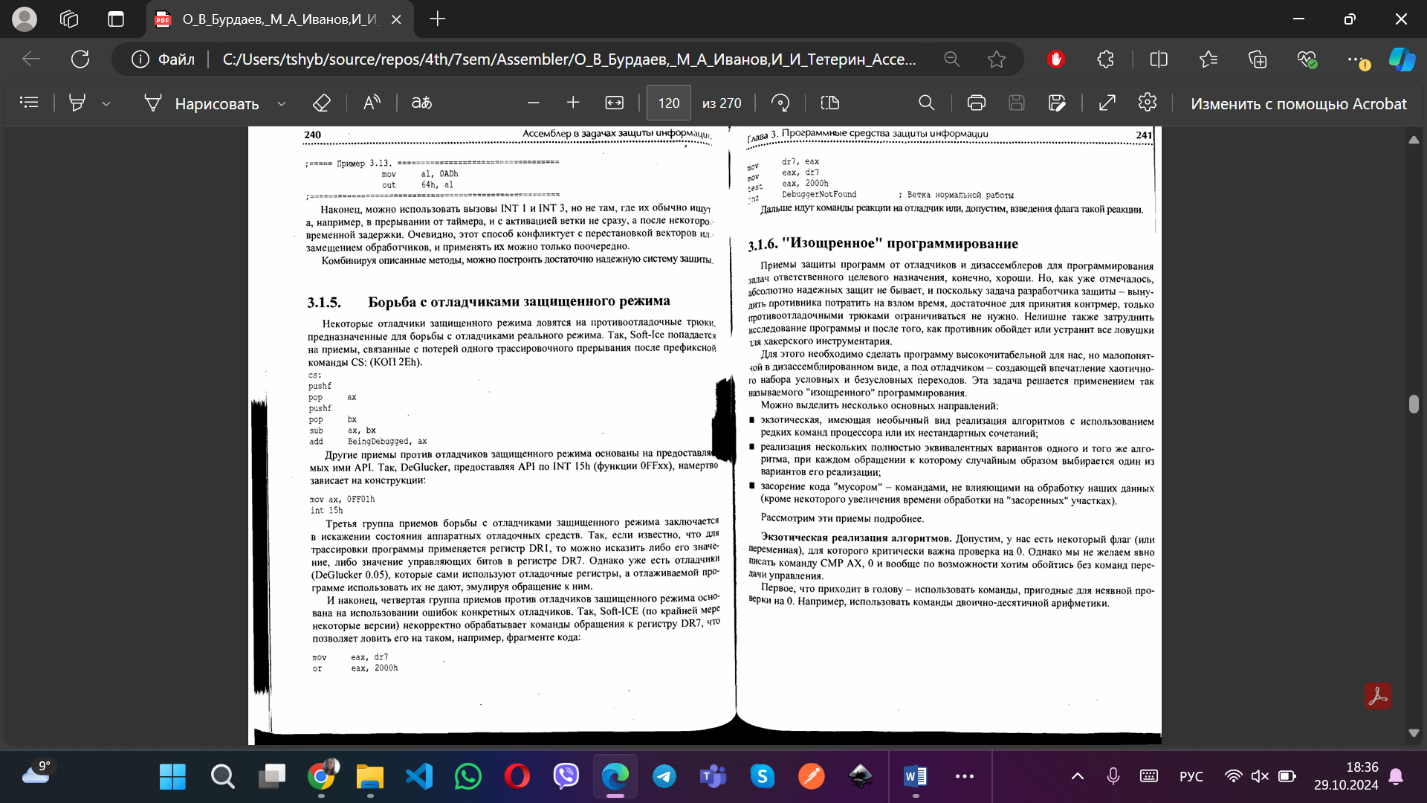
mov [bx], ax

На страницах 236-238 Вашей нелюбимой книжки про ассемблер описано несколько способов определения того, находится ли Ваша программа под отладчиком. Надо просто почитать. Как и пункты 3.1.4.2 и 3.1.5. Всё равно, как и я, ничего не поймёте. А можете просто открыть книжку на указанной странице и тут же её закрыть, но подтвердив факт этого процесса открытия-закрытия, прислав мне на почту «вырезку» из книжки, начиная со страницы 236 и заканчивая «Изощренным программированием». И это был вопрос 4. Вырезку сопроводить указанием времени, когда эта «вырезка» была сделана.

**Ответ:** дата и время отображаются в правом нижнем углу







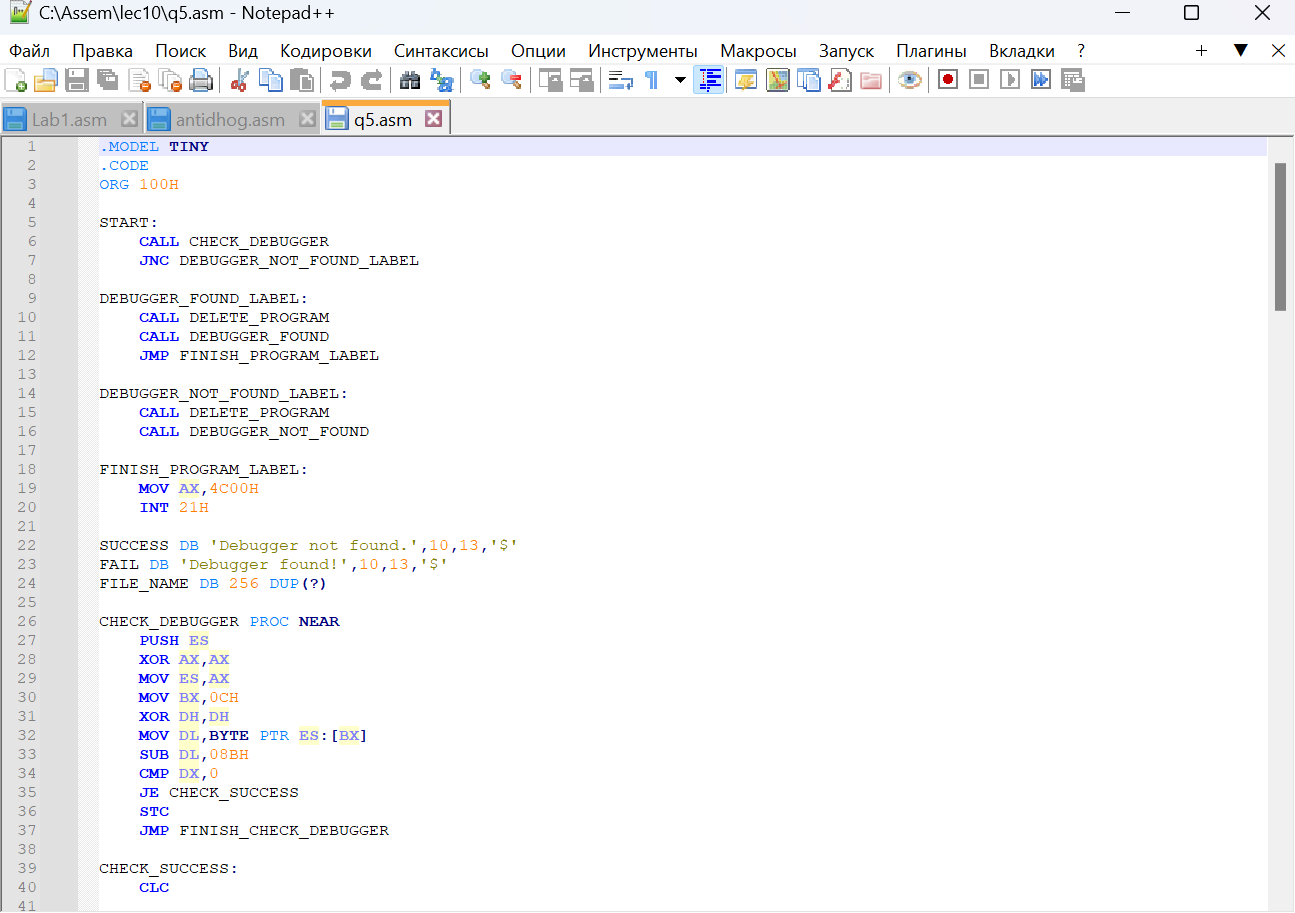
Здесь же и сейчас приведём ещё один текст на ЯЗА ассемблер, который после трансляции и линковки превращается в исполнимый модуль, а он (исполнимый модуль) при загрузке его в ОП превращается в программу.

# **Вопрос 5:**

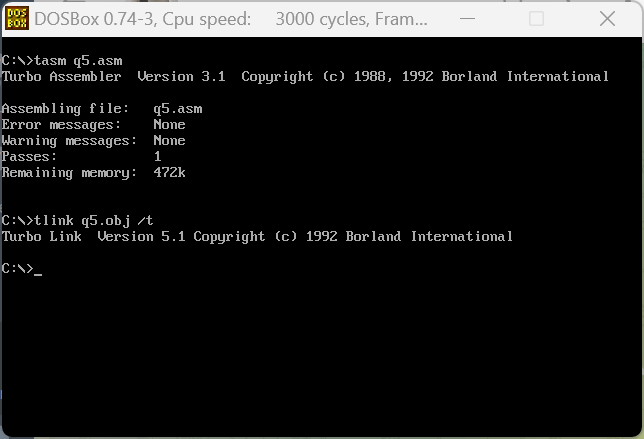
В word-файле, который Вам предстоит сформировать ДО КОНЦА НЕДЕЛИ, ПРЕДСТАВИТЬ ОТЧЕТ О НАПИСАНИИ текста, ТРАНСЛЯЦИИ, ЛИНКОВКИ, ИСПОЛНЕНИЯ полученной программы И ДЕМОНСТРАЦИИ РЕЗУЛЬТАТОВ В РАМКАХ ИСПОЛЬЗОВАНИЯ ОТЛАДЧИКА И оболочки DosBox. Показать, что на самом деле клавиатура блокируется (КАК показать? Ответ на этот вопрос как в анекдоте: «Ты же умная, придумай сама» и, в особенности, что понимается под словами «удаляет сама себя» (действительно ли «программа удаляет сама себя» или же она УДАЛЯЕТ ИЗ РЕЕСТРА ФАЙЛОВ файловой системы имя файла, представляющее собой «исполнимый модуль», или же делает что-то ещё: типа «убивает» и (или) изничтожает).

**Ответ:**

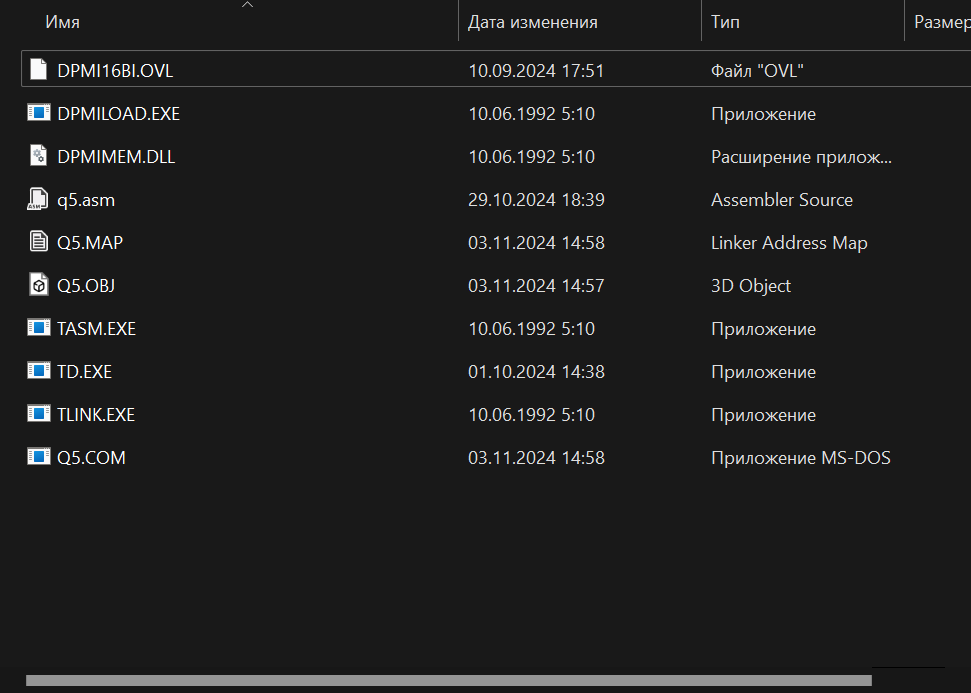
Код программы из лекции 10:



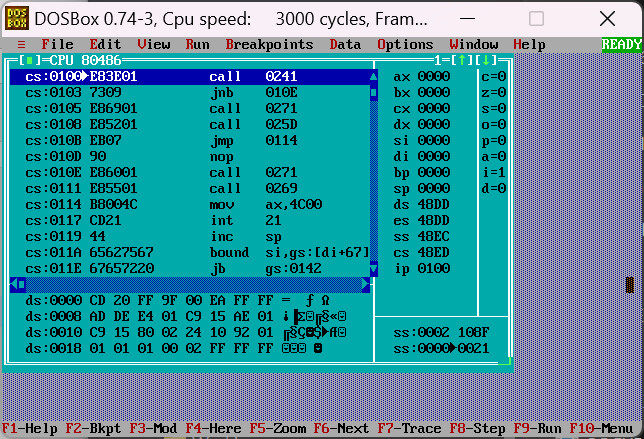
Создаём исполнимый файл:



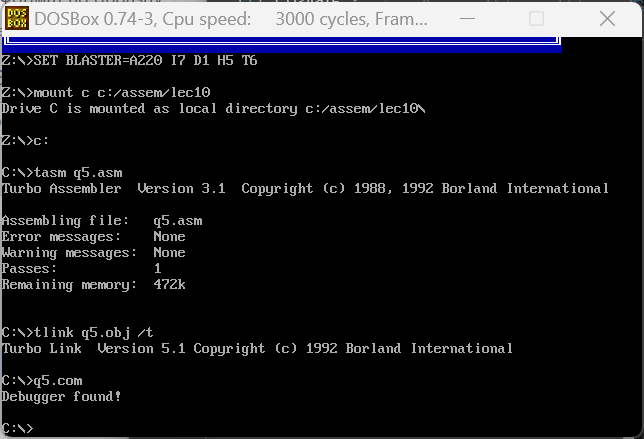
Проверяем, что всё создалось до запуска:



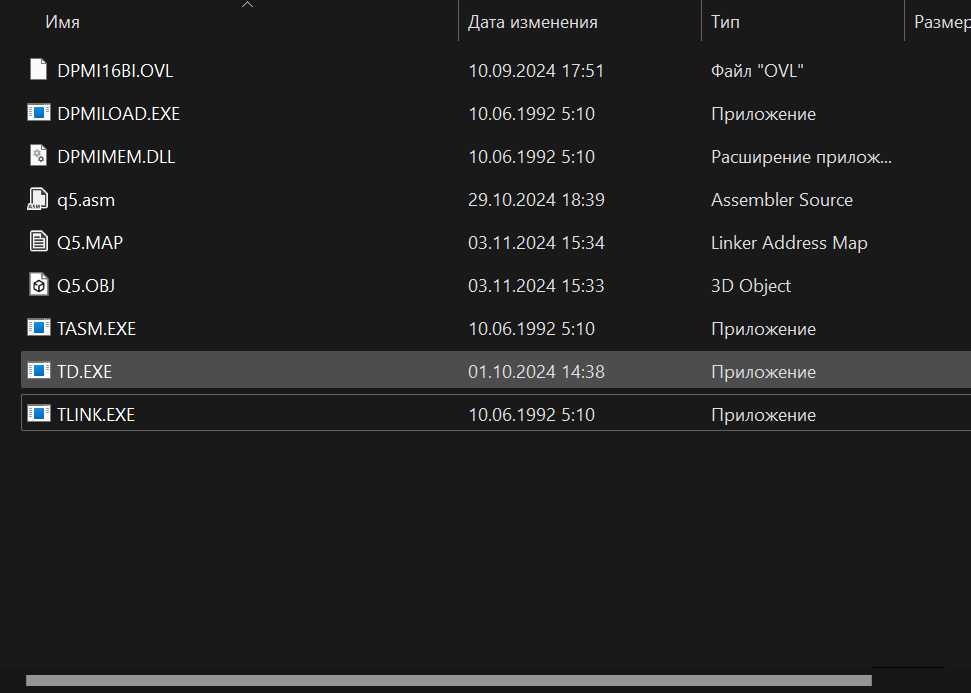
Открываем его в турбо-отладчике:



На моменте запуска через animate процесс просто зациклился, поэтому я запускала через q5.com



После выполнения программы:



# **Вопрос 6.**

Указать номер примера из книги про ассемблер, который был использован при написании процедуры CHECK\_DEBUGGER. И какое прерывание анализируется в этом случае…. Указать его номер.

**Ответ:** Номер примера из книги про ассемблер, который был использован при написании процедуры CHECK\_DEBUGGER: 3.7. Номер прерывания, которое анализируется в этом случае: INT 03h

# Вопрос 7.

Дописать вместо трёх точек событие, которое называется прерыванием с номером 1.

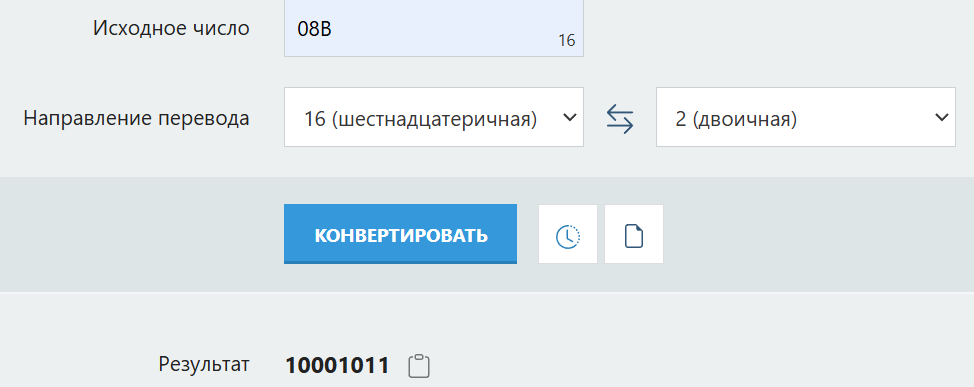
**Ответ:** По адресу 0:4 располагается адрес функции – обработчика прерывания с номером 1 – события так называемое прерывание пошагового режима, вырабатывающееся после выполнения каждой машинной команды в случае, когда в слове флагов установлен бит пошаговой трассировки TF.

# Вопрос 8:

Обманул-не-обманул?

**Ответ:** *«Затем от значения первого байта вектора прерывания с номером 3 отнимается значение 08Bh. Значение 08Bh представляет собой в двоичной системе значение, равное = 0100 1011, а если доступ к этому байту осуществить как к коду команды, то оказывается, что это команда MOV.»*

Вы написали, что значение 08Bh представляет собой в двоичной системе: 01001011. Но это далеко не так.



# Вопрос 9.

Что такое РЕГИСТР ФЛАГОВ. Используя средства Microsoft Word нарисовать его и указать в 10 с.с. НОМЕР БИТА, который «отвечает» ЗА ПЕРЕНОС.

**Ответ:** Флаг – это бит, принимающий значение 1 («флаг установлен»), если выполнено некоторое условие, и значение 0 («флаг сброшен») в противном случае. Процессор имеет регистр флагов, содержащий набор флагов, отражающий текущее состояние процессора.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ бита** | **Обозначение** | **Название** | **Описание** | **Тип флага** |
| 0 | CF | Carry Flag | Флаг переноса | Состояние |