Пароль от архива с исходным текстом и исполнимым модулем: **041**

В тексте программы на языке записи алгоритмов ассемблер записаны следующие строки:

.data

Zu db “ЯД”

ZuZU dw 255 dup (?)

.code

…

LEA DX, Zu

MOV ah, 10

INT 21h

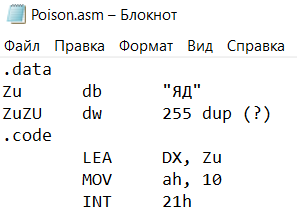
**Упражнение 601**: Сколько символов можно будет ввести пользователю в процессе выполнения функции 10 21h прерывания. И сколько раз можно будет стукнуть по различным клавишам на клавиатуре?

**Ответ на вопрос (12.10.2021)**: «Функция прерывания 21h вообще говоря предназначена для того, чтобы читать запись файла произвольной длины. То есть, по идее, стукать по клавиатуре можно сколько угодно раз до тех пор, пока не будет нажат Enter. Меня несколько смутило, что мы добавляем какую-то ерунду (Яд) в регистр DX (а это регистр данных вроде бы), так что что-то могло сломаться и пойти не так, но без отладки сложно сказать, что именно. Возможно, после добавления адреса области памяти, в которой находится «ЯД», в регистр DX мы не сможем стукать по клавиатуре сколько угодно раз до нажатия Enter….»

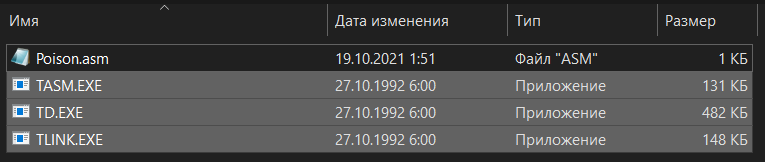
**Домашнее задание** – до следующего вторника – решение в виде трёх файлов прислать на почту Zubovich@mail.ru Используя оболочку DosBox получить программу, которая позволяет определить, сколько на самом деле символов можно ввести с клавиатуры и сколько раз можно стукнуть по клавишам на клавиатуре. После чего, используя какой-либо отладчик ПОКАЗАТЬ на Экране, что находится в области памяти, начиная с адреса Zu. Свинья: отладчик представляет собой программное средство td.exe, но можно использовать любой другой…

**Ответ на домашнее задание**

Для начала я создал файл с именем Posion.asm, внутри которого поместил исходный текст программы:

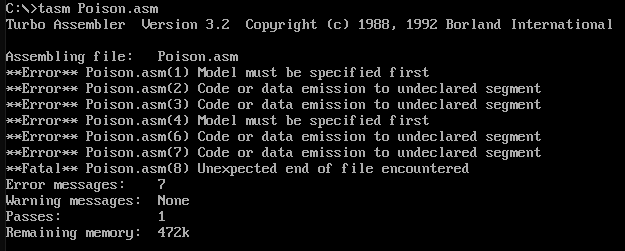
****

Также я поместил в папку с файлом (внутри которого помещён исходный текст) следующие исполнимые модули:

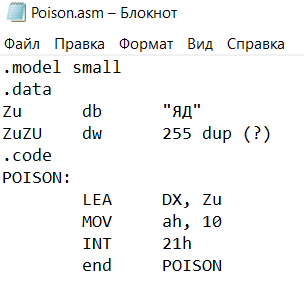


Они нам пригодятся для того, чтобы получить объектный модуль (с помощью TASM.EXE), а затем исполнимый модуль программы (с помощью TLINK.EXE), а также для того, чтобы совершить отладку (с помощью TD.EXE).

Я попробовал получить объектный модуль из имеющегося исходного текста, но ничего не получилось:



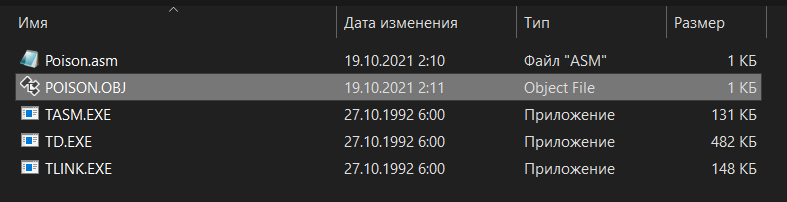
Поэтому я внёс некоторые изменения в исходный текст:



После чего снова попробовал получить объектный модуль:



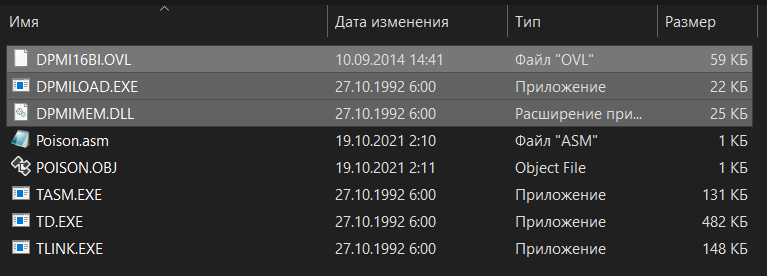
Объектный модуль успешно получен:



Теперь я попробовал получить исполнимый модуль:



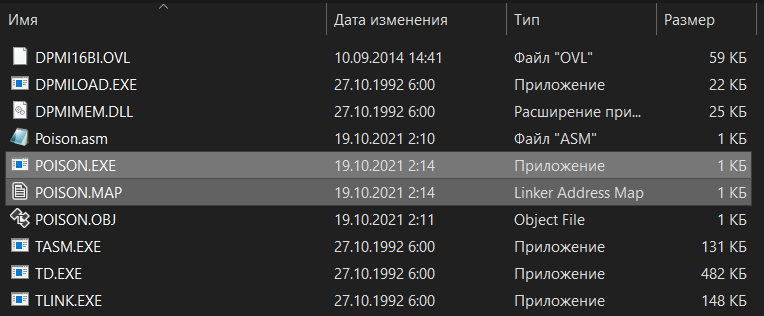
Я понял, что мне не хватает следующих компонент:



После чего я снова попробовал получить исполнимый модуль:



Успешно получен исполнимый модуль:



После этого я попробовал его запустить и провести некоторые эксперименты:



Я попросту вводил поочерёдно символы с клавиатуры…

На 5-ом круге, когда я захотел ввести ‘]’, у меня уже не получилось…

Один мой круг состоял из 33 символов: от ‘q’ до ‘/’.

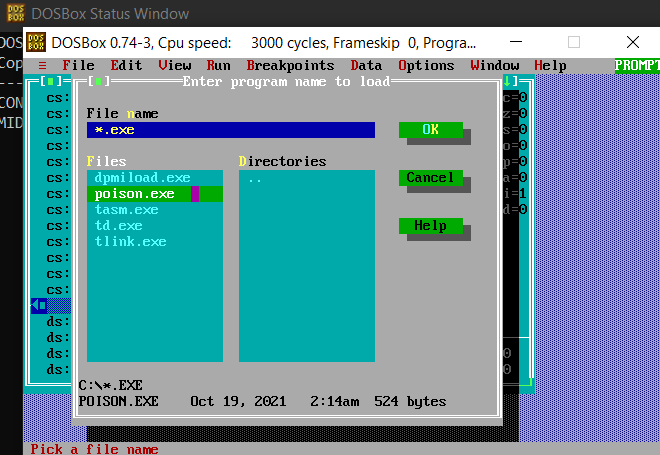
Таким образом я проделал 4 полных круга и ещё 11 символов от 5-ого круга, итого:

33\*4 + 11 = 132 + 11 = 143 символа.

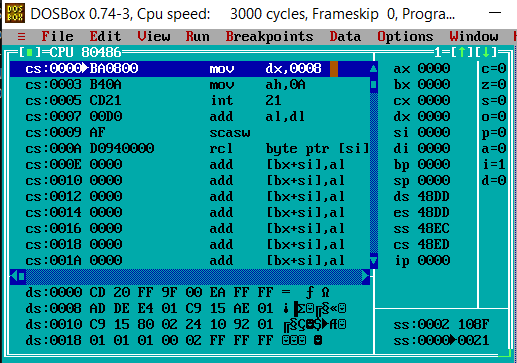
Максимальное количество раз, которое можно стукнуть по клавишам на клавиатуре в процессе выполнения функции 10 21h прерывания, равно **143 символам**.

Теперь поэкспериментируем с отладчиком…

Для этого я запустил TD.EXE и выбрал в качестве исполнимого модуля для отладки мой POSION.EXE:

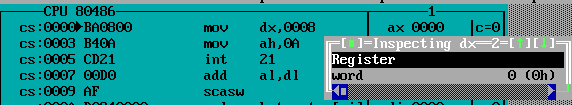


При загрузке исполнимого модуля в оперативную память я увидел следующее:

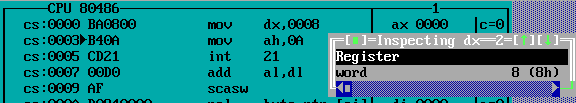


Там, где в исходном тексте стояла «Zu», отладчик показывает «0008».

Регистр данных DX до того, как мы положили в него данные из «Zu»:



Регистр данных DX после того, как мы положили в него данные из «Zu»:



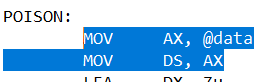
Т.е. «Zu» добавил в регистр DX значение «0008».

Как мы знаем, регистр DX является регистром данных (16-битным), который в свою очередь состоит из регистров DH и DL (в каждом по 8 бит).

При выполнении 10-ой функции прерывания 21h, регистр DS:DX служит адресом входного буфера (об этом я прочитал в справке данной функции где-то на помойке).

Выходит, что при задании адреса входного буфера «0008» мы повлияли на 10-ую функцию прерывания 21h таким образом, что она позволила нам ввести лишь 143 символа.

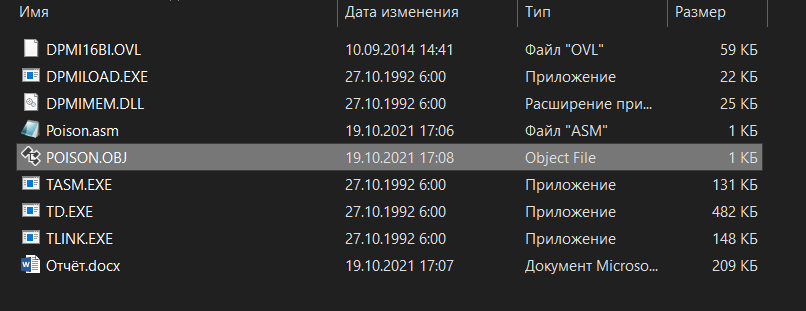
Позже я добавил в исходный текст следующие две строки:



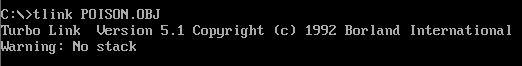
Я сделал это с целью положить данные в регистр DX (ведь он тоже связан со входным буфером).

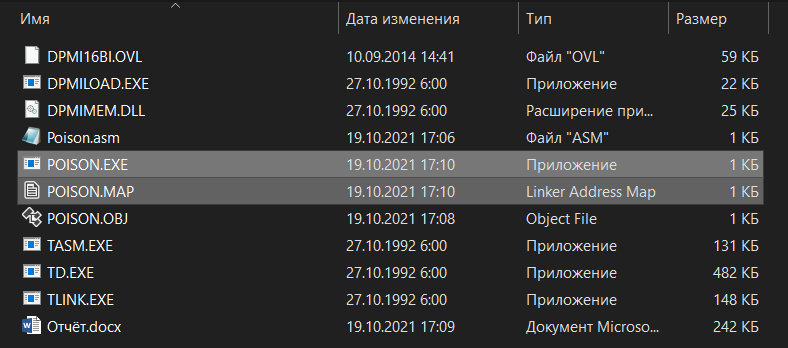
После чего я получил объектный модуль:



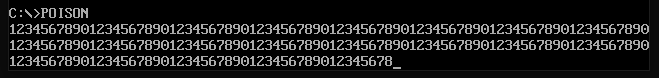


А затем я получил исполнимый модуль:





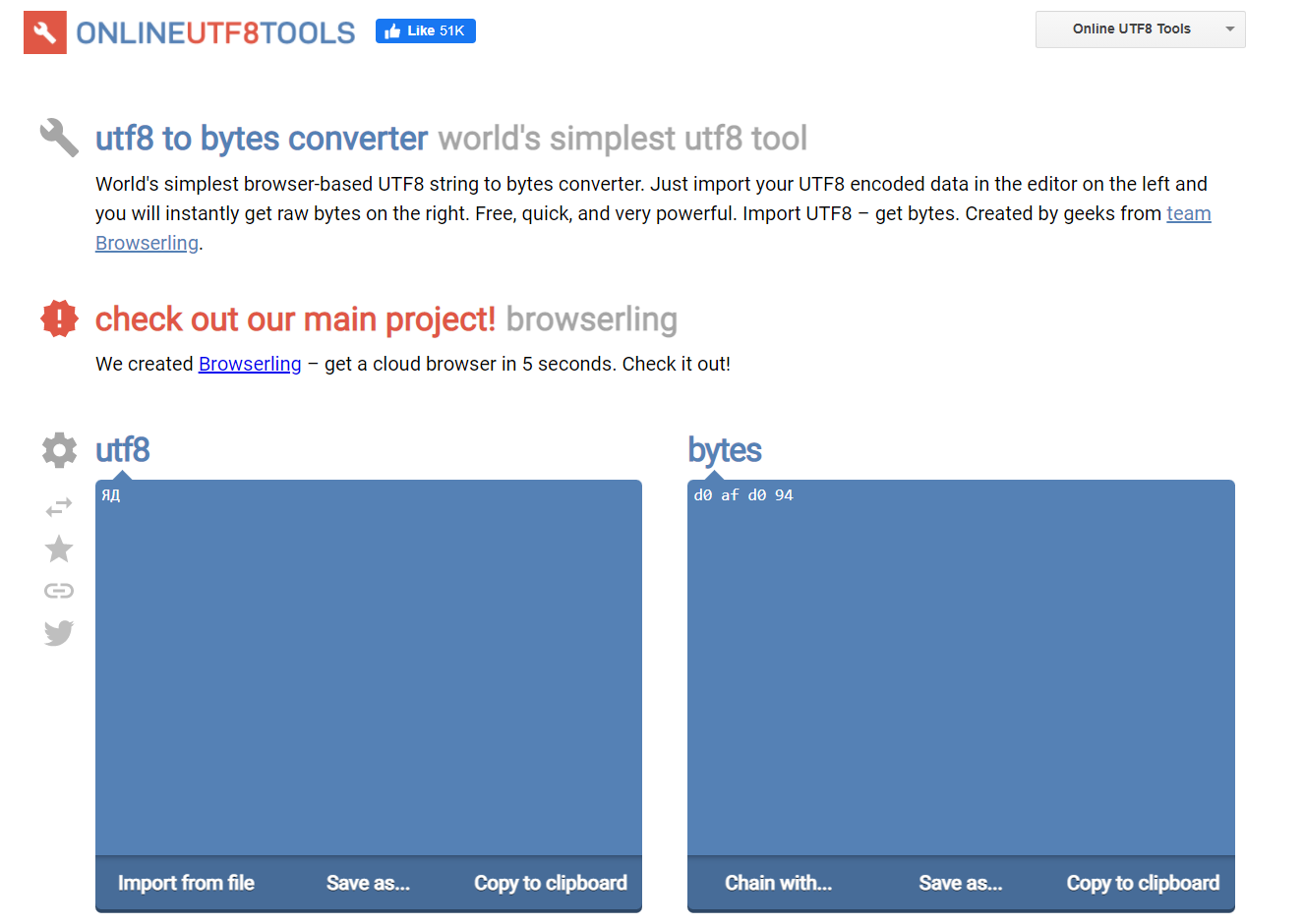
Затем я снова попробовал посчитать, сколько клавиш (не считая функциональных) я смогу нажать:



Для удобства я вводил цифры от «1» до «0», таким образом я 20 раз ввёл 10 цифр, и ещё 8 цифр с 21-ого раза.

Таким образом, я смог ввести **208** символов, что в 16-ричном виде соответствует числу **D0**.

Позже я залез на помойку и выяснил, что в кодировке UTF-8, которую я использовал в своём блокноте слово «ЯД» соответствует следующему набору байтов:



**D0 AF D0 94**

Надо, же, какое совпадение! Первая байт слова «ЯД» – **D0** – соответствует числу символов, которые я мог ввести во время выполнения программы, а именно **208**.

Пароль от архива с исходным текстом и исполнимым модулем: **041**