***Лабораторные занятия 11-12-13 (15.11, 22.11, 29.11.2023).***

***НАРЯДУ С ВЫПОЛНЕНИЕМ ЛАБОРАТОРНОЙ РАБОТЫ №3* ЛАБОРАТОРНАЯ РАБОТА №3. Часть 1. SPY. Изучить то, что в Лекции 9 и в книжке про ассемблер называется «шпионский клавиатур», исправить ошибки, если они есть (или построить свой собственный текст на языке записи алгоритмов ассемблер) и получить полностью соответствующий постановке задачи «Спай» текст на языке записи алгоритмов ассемблер - получение в файле совокупности байт, которые содержат данные о процессе нажатия клавиш на клавиатуре (НЕ ВВОДА СИМВОЛОВ, А НАЖАТИЙ КЛАВИШ НА КЛАВИАТУРЕ!!!). Стоимость: Спай – 16 баллов. Отсутствие - -8 баллов. Сроки – 22.11.2023.**

**ЛАБОРАТОРНАЯ РАБОТА №3. Часть 2. YPS!!! Получить программу, которая по информации, СОДЕРЖАЩЕЙСЯ в файле из части 1 лаб работы№3, выводит на экран СООТВЕТСТВУЮЩУЮ ПОСЛЕДОВАТЕЛЬНОСТЬ СИМВОЛОВ, НАБРАННЫХ НА КЛАВИАТУРЕ. Например, если на клавиатуре были «натиснуты» клавиши F1, Shift, mama, Ctrl, Ввод, на экране должно получиться Mama. Сроки: 6.12.2023. Спай наоборот – 32 балла, отсутствие – 16 баллов. Решение может быть только в рамках DosBox и языка записи алгоритмов assembler/**

***И, возможно, показом скачущего боксика, вируса, антивируса и знаний шпиона с турбоотладчиком на примере папы):***

**ДЕЛАЕМ ТО, ЧТО НАЗЫВАЕТСЯ:**

***Программа, «удаляющая сама себя» или «программа, удаляющая свой исполнимый модуль»?***

***1.Из следующего текста на языке записи алгоритмов ассемблер (который также есть в Лекции 11):***

.MODEL TINY

.CODE

ORG 100H

START:

CALL CHECK\_DEBUGGER

JNC DEBUGGER\_NOT\_FOUND\_LABEL

DEBUGGER\_FOUND\_LABEL:

CALL DELETE\_PROGRAM

CALL DEBUGGER\_FOUND

JMP FINISH\_PROGRAM\_LABEL

DEBUGGER\_NOT\_FOUND\_LABEL:

CALL DELETE\_PROGRAM

CALL DEBUGGER\_NOT\_FOUND

FINISH\_PROGRAM\_LABEL:

MOV AX,4C00H

INT 21H

SUCCESS DB 'Debugger not found.',10,13,'$'

FAIL DB 'Debugger found!',10,13,'$'

FILE\_NAME DB 256 DUP(?)

CHECK\_DEBUGGER PROC NEAR

PUSH ES

XOR AX,AX

MOV ES,AX

MOV BX,0CH

XOR DH,DH

MOV DL,BYTE PTR ES:[BX]

SUB DL,08BH

CMP DX,0

JE CHECK\_SUCCESS

STC

JMP FINISH\_CHECK\_DEBUGGER

CHECK\_SUCCESS:

CLC

FINISH\_CHECK\_DEBUGGER:

POP ES

RET

CHECK\_DEBUGGER ENDP

DEBUGGER\_FOUND PROC NEAR

MOV AH,9H

MOV DX,OFFSET FAIL

INT 21H

MOV AL,0ADH

OUT 64H,AL

RET

DEBUGGER\_FOUND ENDP

DEBUGGER\_NOT\_FOUND PROC NEAR

MOV AH,9H

MOV DX,OFFSET SUCCESS

INT 21H

RET

DEBUGGER\_NOT\_FOUND ENDP

DELETE\_PROGRAM PROC NEAR

MOV BX,OFFSET FILE\_NAME

CALL COPY\_FILE\_NAME

MOV DX,OFFSET FILE\_NAME

CALL OPEN\_FILE

CALL CLEAR\_FILE

CALL CLOSE\_FILE

MOV DX,OFFSET FILE\_NAME

CALL DELETE\_FILE

RET

DELETE\_PROGRAM ENDP

COPY\_FILE\_NAME PROC NEAR ; DS:BX - WHERE TO COPY FILE NAME

PUSH ES

MOV AX,ES:[2CH]

MOV ES,AX

MOV SI,-1

SEARCH\_01:

INC SI

MOV AL,ES:[SI]

CMP AL,0

JNE SEARCH\_01

MOV AL,ES:[SI+1]

CMP AL,1

JNE SEARCH\_01

ADD SI,2

COPY\_NAME:

INC SI

MOV AL,ES:[SI]

MOV [BX],AL

INC BX

CMP AL,0

JNE COPY\_NAME

POP ES

RET

COPY\_FILE\_NAME ENDP

OPEN\_FILE PROC NEAR ; DS:DX - FILE NAME, BX - RESULT (FILE HANDLE)

MOV AX,3D02H

INT 21H

MOV BX,AX

RET

OPEN\_FILE ENDP

CLOSE\_FILE PROC NEAR ; BX - FILE HANDLE

MOV AH,3EH

INT 21H

RET

CLOSE\_FILE ENDP

GET\_FILE\_LENGTH PROC NEAR ; BX - FILE HANDLE, SI - RESULT

MOV AX,4202H

XOR CX,CX

XOR DX,DX

INT 21H

MOV SI,AX

MOV AX,4200H

XOR CX,CX

XOR DX,DX

INT 21H

RET

GET\_FILE\_LENGTH ENDP

CLEAR\_FILE PROC NEAR ; BX - FILE HANDLE

CALL GET\_FILE\_LENGTH

MOV CX,SI

MOV DX,OFFSET TRASH\_BUF

PRINT\_TRASH\_TO\_FILE:

PUSH CX

MOV AH,40H

MOV CX,1

INT 21H

POP CX

LOOP PRINT\_TRASH\_TO\_FILE

RET

TRASH\_BUF DB 0

CLEAR\_FILE ENDP

DELETE\_FILE PROC NEAR ; DS:DX - FILE NAME

MOV AH,41H

INT 21H

RET

DELETE\_FILE ENDP

END START

**ВЫДЕЛИТЬ ТОЛЬКО ТО, ЧТО КАСАЕТСЯ НА САМОМ ДЕЛЕ «УДАЛЕНИЯ ИЗ СПИСКА (РЕЕСТРА) ФАЙЛОВ ОС ИМЕНИ ИСПОЛНИМОГО ФАЙЛА», ИЗ КОТОРОГО ПОЛУЧАЕТСЯ ПРОГРАММА. ПРОТРАНСЛИРОВАТЬ ВЫДЕЛЕННЫЙ ФРАГМЕНТ-ТЕКСТ НА ЯЗЫКЕ ЗАПИСИ АЛГОРИТМОВ АССЕМБЛЕР, ОТЛИНКОВАТЬ, ВЫПОЛНИТЬ ПРОГРАММУ с помощью оболочки DosBox. Продемонстрировать процесс «самоудаления». В том числе показать, какой вид принимает каталог, в котором располагался этот исполнимый модуль.**

**2.ВЫДЕЛИТЬ ТОЛЬКО ТО, ЧТО КАСАЕТСЯ НА САМОМ ДЕЛЕ «ИЗНИЧТОЖЕНИЯ ИСПОЛНИМОГО ФАЙЛА НА ВНЕШНЕМ УСТРОЙСТВЕ», ИЗ КОТОРОГО ПОЛУЧАЕТСЯ ПРОГРАММА. ПРОТРАНСЛИРОВАТЬ ВЫДЕЛЕННЫЙ ФРАГМЕНТ-ТЕКСТ НА ЯЗЫКЕ ЗАПИСИ АЛГОРИТМОВ АССЕМБЛЕР, ОТЛИНКОВАТЬ, ВЫПОЛНИТЬ ПРОГРАММУ с помощью оболочки DosBox. Продемонстрировать процесс «изничтожения».**

**В том числе показать, какой вид принимает каталог, в котором располагался этот исполнимый модуль. И что происходит, если делается попытка загрузить в ОП исполнимый модуль и выполнить программу.**

**3.УБИТЬ ФАЙЛ – ТО ЕСТЬ НЕ УДАЛЯТЬ ИМЯ ИСПОЛНИМОГО МОДУЛЯ ИЗ РЕЕСТРА ФАЙЛОВ, НЕ ЗАПОЛНЯТЬ СОДЕРЖИМОЕ ИСПОЛНИМОГО МОДУЛЯ НА ВНЕШНЕМ УСТРОЙСТВЕ НУЛЯМИ, НО ПО АДРЕСУ, СООТВЕТСТВУЮЩЕМУ ТОЧКЕ ВХОДА В ПРОГРАММУ ЗАПИСАТЬ КОМАНДУ INT 20h!!!. Продемонстрировать, что получается в результате в каталоге файлов, что получается в файле, который «убит», представив его содержимое в 16 с.с.**

**4. ДЛЯ СЛУЧАЯ УДАЛЕНИЯ ИМЕНИ ИСПОЛНИМОГО МОДУЛЯ ИЗ РЕЕСТРА ФАЙЛОВ ВОССТАНОВИТЬ УДАЛЕННЫЙ ФАЙЛ (НАЙТИ ЕГО НА ВНЕШНЕМ УСТРОЙСТВЕ БЕЗ ИСПОЛЬЗОВАНИЯ РЕЕСТРА ФАЙЛОВ, НО ИСПОЛЬЗУЯ ЕГО СИГНАТУРУ, ПРОЧИТАТЬ СОДЕРЖИМОЕ В ОП, ОТКРЫТЬ НОВЫЙ ФАЙЛ, ЗАПИСАТЬ В НЕГО СОДЕРЖИМОЕ УДАЛЕННОГО ФАЙЛА, ЗАКРЫТЬ). Процесс восстановления продемонстрировать: показать каталог до удаления исполнимого модуля, показать в 16 с.с. его содержимое, показать каталог после «удаления», продемонстрировать процесс восстановления удаленного файла, показать каталог после восстановления и функционирование восстановленного файла.**

**РЕЗУЛЬТАТЫ (исходные тексты, исполнимые модули, отчеты по каждому из вышеперечисленных 4-х пунктов – всего 12 файлов) О ПРОДЕЛАННОЙ НА ЗАНЯТИЯХ 11-12-13 в виде Word-файла ПРИСЛАТЬ НА ЭЛЕКТРОННУЮ ПОЧТУ ДО КОНЦА ДНЯ 29.11.2023 года).**

**В результате получается такое условие *ЛАБОРАТОРНОЙ РАБОТЫ №4.* Разработать программу, которая:**

**4.1. Получает от пользователя, сидящего за компом, пароль. Паролем является ФАМИЛИЯ НА РУССКОМ ЯЗЫКЕ правообладателя программы, которая (фамилия») «прячется» в исполнимом модуле.**

**4.2. Пользователю предоставляет ТРИ (И ТОЛЬКО ТРИ) ПОПЫТКИ ВВОДА ПАРОЛЯ. ПРИ ЭТОМ ДОЛЖНА БЫТЬ ОБЕСПЕЧЕНА «ЗАЩИТА ОТ ДУРАКА», который считает себя самым умным. А именно: ПОПЫТКИ НЕ ОБНУЛЯЮТСЯ в случае, если «хитрик-бандит» с целью увеличения числа попыток прерывает исполнение программы с целью «обмана» программы, думая, что при следующем запуске программы у него снова будет ТРИ ПОПЫТКИ. То есть, набрав один раз пароль вне зависимости от того, сколько раз запускалась программа, у него остаётся НА ОДНУ ПОПЫТКУ МЕНЬШЕ!!!**

**4.3. Если ПАРОЛЬ ТРИ РАЗА НАБРАН НЕВЕРНО, ПРОГРАММА ИЗНИЧТОЖАЕТ СЕБЯ В БРЕННОМ МИРЕ КОМПЬЮТЕРА: а). Удаляет имя исполнимого модуля из реестра файлов; б) Зачищает содержимое исполнимого файла НУЛЯМИ. в) Заканчивается.**

**4.4. Если пароль ХОТЯ БЫ ПРИ КАКОЙ-ЛИБО ПОПЫТКЕ набран верно, ТО НА ЭКРАН ВЫВОДИТСЯ СНАЧАЛА ПОСЕРЕДИНЕ ЧЕРНОГО-ЧЕРНОГО ЭКРАНА СООБЩЕНИЕ «GIANT!!!», а затем ЧЕРНЫЙ-ЧЕРНЙ ЭКРАН ЗАПОЛНЯЕТСЯ 4 000 СИМВОЛАМИ ДОЛЛАРА!**

**РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ (три файла – исходный текст, исполнимый модуль, отчет о проделанной работе) должен быть прислан на почту ДО конца дня 13.12.2023 года. СТОИМОСТЬ: Реализация 4.1. – 8 баллов. Реализация 4.2. без защиты от дурака – 8 баллов, с защитой от дурака – 24 балла. Реализация 4.3. – 16 баллов. Реализации 4.4. – 32 балла.**