**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 15**

**студента гр. И-2-18**

Мартыненко Романа Сергеевича

По дисциплине: Программирование встроенных систем

Тема занятия: Ввод данных с помощью ОС и BIOS

Цель занятия: научиться выводить текст на экран используя различные функции операционной системы

1. **Задание**

Составить программу ввода любого числа размером до 2-х байт с помощью функций DOS или BIOS и вывести введенное число на экран (при выводе на экран использовать программный код лабораторной работы No2).

При вводе числа осуществить проверку на вводимые символы, т.е. отображать только символы, соответствующие кодам цифр. Ввод завершить каким-либо признаком конца ввода числа (перед вводом числа вывести сообщение типа «Нажатие на клавишу Enter завершает ввод числа»). Обязательным условием выполнения работы является возможность обработки введенного числа в коде программы. При вводе числа с клавиатуры и выводе числа на экран необходимо учесть, что вводимые и выводимые символы ASCII-кодах.

1. **Листинг файла** Lab\_15.asm

.model small

.286

.stack 100h

.data                   ;Идентификатор сегмента данных

    Message db 'Pressing the Enter key completes the input of the number:',10,13,'$'

    Merr\_big db 'The number is big!',10,13,'$'

    Merr\_empty db 'You did not enter a number!',10,13,'$'

    Message\_good db 'The number of successfully stored in the memory area "data" - number',10,13,'$'

    muller dw 10

    number dw 0

    b db '     ',10,13,'$'  ;Переменная в которую мы запишем переменную a в виде строки

    c dw 10                 ;Делитель 10

.code

easy\_video macro

    mov ah, 0Fh     ;уточнить параметры видеорежима

    int 10h         ;

    mov ah, 03h     ;читать позицию курсора

    int 10h         ;выход: dh,dl - позиция курсора

    endm

start:

    mov ax, @data           ;Передаем @data в ax

    mov ds, ax              ;Передаем ax в ds

    mov ax, 0

    mov si, 0

    mov dx, offset Message  ;В dx записываем значение переменной Message

    mov ah,9h               ;в ah передаем код -9h (Вывод на экран)

    int 21h                 ;Вызываем прерывание для выполнения команды

    easy\_video

    jmp k1

clearsimvol:    ; Стираем символ и переводим курсор на 1 шаг назад

    easy\_video  ; Получаем полажение курсора

    cmp dl, 0

    je k1       ; Если 0 то переходим на k1

    dec dl

    mov ah, 02h ;переводим пурсор назад на 1

    int 10h

    mov dl, 20h ;очищаем символ при помощи пробела

    mov ah, 02h

    int 21h

    easy\_video  ; Получаем полажение курсора

    dec dl

    mov ah, 02h ;переводим пурсор назад на 1

    int 10h

    ;чистим стек от лишнего элемента

    dec si

    pop ax

    jmp k1

k1:    ; обработка ввода с клавиатуры

    mov ah,08h

    int 21h

    cmp al, 8   ; При Backspace

    je clearsimvol

    cmp al, 13  ; При enter перейти в k2

    je k2

    cmp al, '0' ; пропускать ввод если это не числа от 0-9

    jl k1

    cmp al, '9'

    jg k1

    cmp si, 5   ; ограничение ввода в 5 символов

    je k1

    mov ah, 0

    push ax     ; отправка символа в стек

    inc si

    mov dl, al

    mov ah, 02h ; отрисовка символа

    int 21h

    jmp k1

k2:

    mov dl, 10

    mov ah, 02h

    int 21h

    mov di, 0

    mov bx, 0   ; в bx получим итоговое число которое запишем в number

    mov dx, 0

    cmp si, 0

    je err\_empty    ; ошибка, если ничего не ввели, но нажали enter

    pop ax      ;забираем символ из стека

    push ax

    sub ax,'0'  ;преобразуем к числу

    add bx, ax

k3:

    cmp si, 0

    je rezult   ; если опустошили стек, то выводим результат

    pop ax      ;забираем символ из стека

    dec si

    sub ax,'0'  ;преобразуем к числу

    mov cx, di  ; передаем количество раз умножения на 10 (1 для десятков, 2 для сотен и ...)

mull:   ;цикл умножения

    mul muller

    loop mull

    cmp dx, 0

    jne err\_big ;переполнение при умножении

    add bx, ax

    jc err\_big  ;переполнение при сложении

    inc di

    jmp k3

rezult:     ; код взят из лаб 8 (перевод числа в строку), для того чтобы проверить работу 1 части кода

    mov number, bx

    mov dx, offset Message\_good  ;В dx записываем значение переменной Message

    mov ah,9h               ;в ah передаем код -9h (Вывод на экран)

    int 21h                 ;Вызываем прерывание для выполнения команды

    mov si, 4            ;Записываем 4 в si

    mov ax, number           ;Записываем a в ax

l1:                     ;Метка начала блока k1

    xor dx, dx          ;Обнуление регистра dx (быстрее чем mov dx, 0)

    div c               ;Делим содержимое ax на c

    add dl, '0'         ;Прибовляем к младшему байту dx '0' (Преобразовали число к символу)

                        ;'0' = 48 (48-57 это номера символов от 0-9 в таблице ascii)

    mov [b + si], dl    ;Записываем в строку полученный символ

    dec si              ;Отнимаем 1 от регистра si

    cmp ax, 0           ;Сравниваем ax с 0

    jne l1              ;Если ax не 0, то выполняем (k1) деление до тех пор пока число не кончится

    mov dx, offset b    ;В dx записываем значение переменной b

    mov ah, 9h          ;в ah передаем код -9h (Вывод на экран)

    int 21h             ;Вызываем прерывание для выполнения команды

    jmp konec

err\_big:

    mov dx, offset Merr\_big  ;В dx записываем значение переменной Message

    mov ah,9h               ;в ah передаем код -9h (Вывод на экран)

    int 21h                 ;Вызываем прерывание для выполнения команды

    jmp konec

err\_empty:

    mov dx, offset Merr\_empty  ;В dx записываем значение переменной Message

    mov ah,9h               ;в ah передаем код -9h (Вывод на экран)

    int 21h                 ;Вызываем прерывание для выполнения команды

    jmp konec

konec:

    mov ah,4ch

    int 21h

end start

1. **Результат выполнения лабораторной работы**

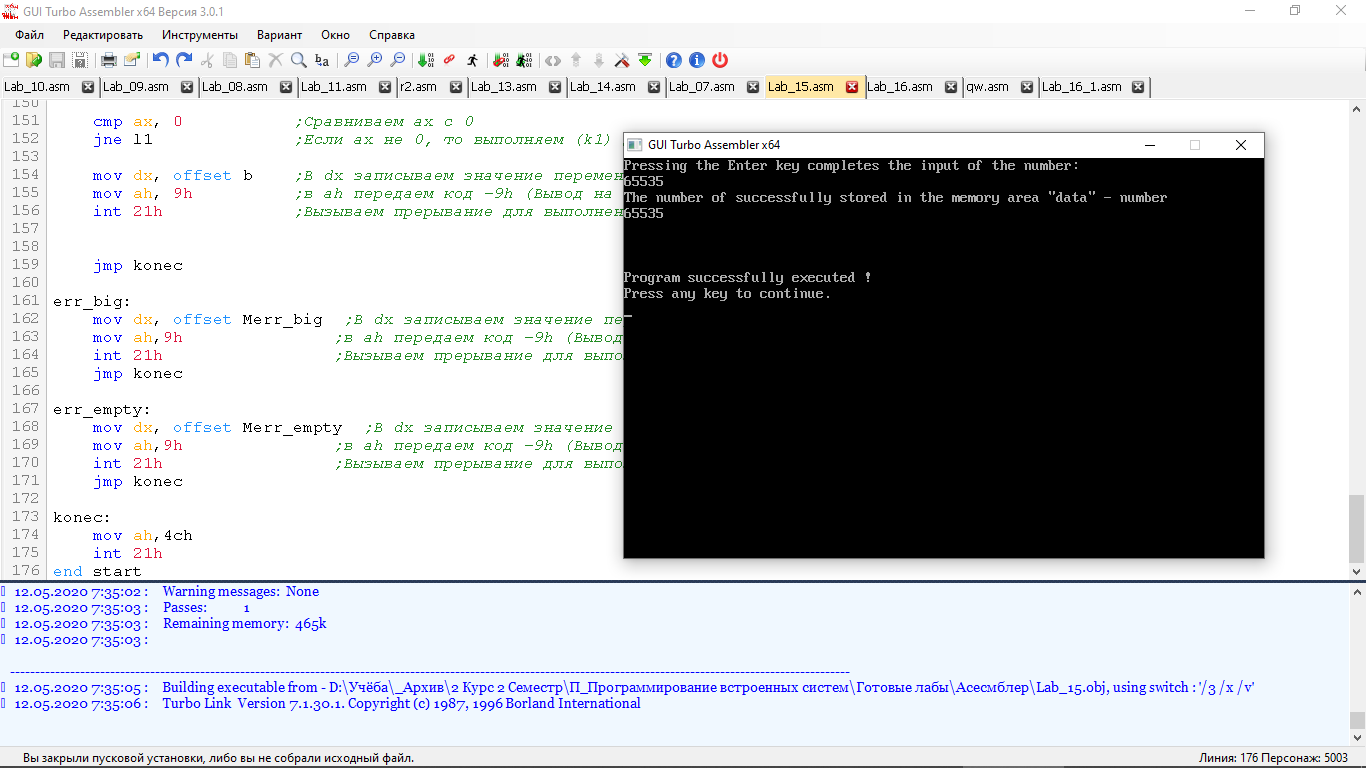


Рис 1. (Скриншот результата работы кода файла Lab\_15.asm – когда число введено правильно)

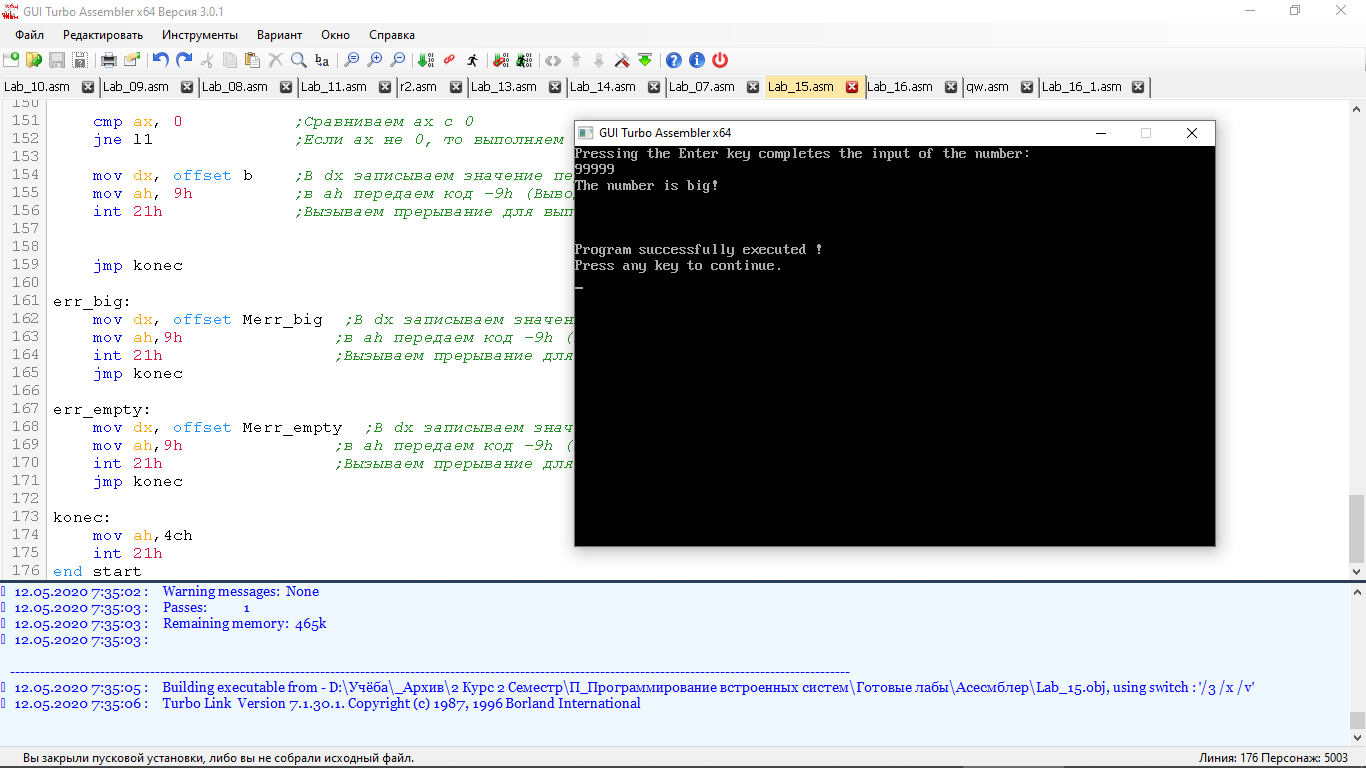


Рис 2. (Скриншот результата работы кода файла Lab\_15.asm – когда введено число больше 2-х байт)

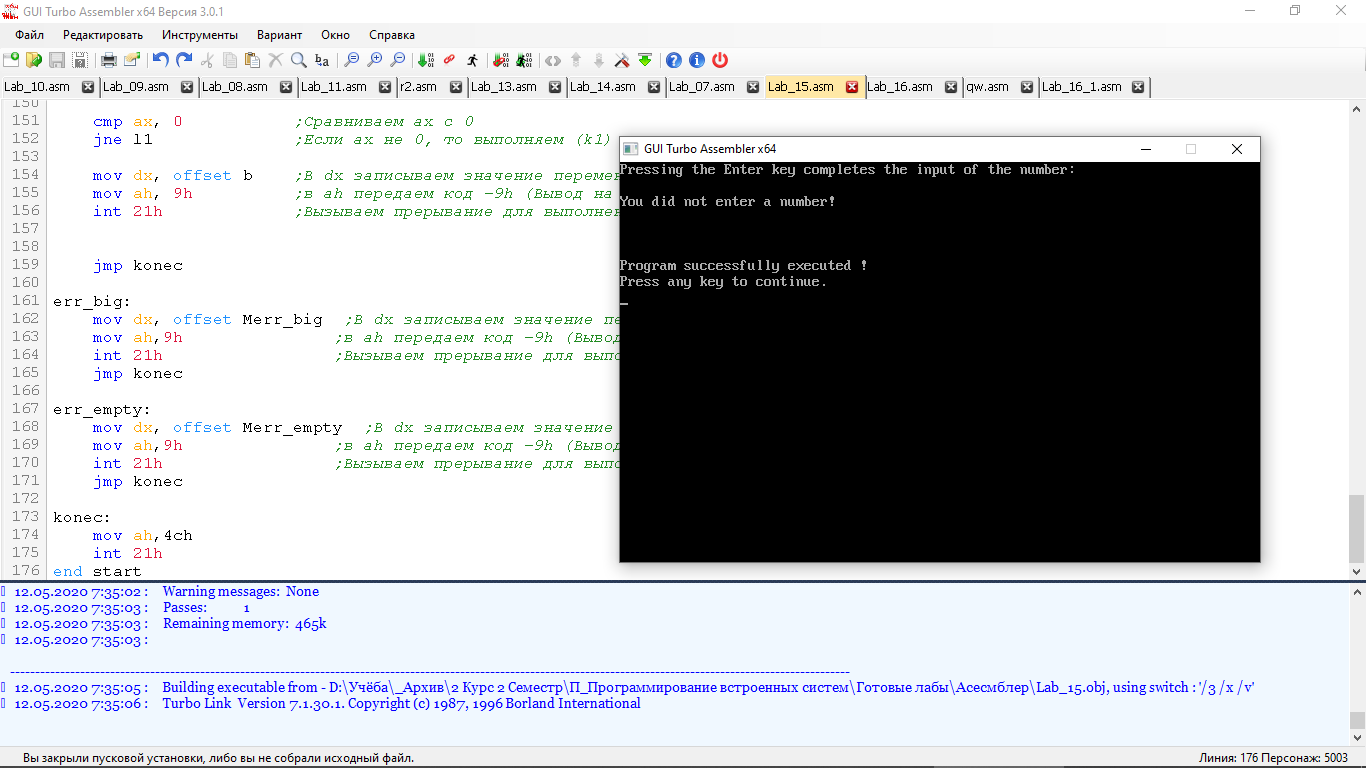


Рис 3. (Скриншот результата работы кода файла Lab\_15.asm – когда число не введено вообще)