**Лабораторная № 1**

**Цель** лабораторной работы № 1: получить навыки написания консольных приложений на ассемблере.

**Задание к** работе. Написать на языке ассемблер программу, позволяющую:

1) ввести (взять из файла) несколько переменных;

2) вычислить



подставляя в неё

n-48

d-12

m-1

y-1982

3) Вывести результат на экран (в файл).

Исполнительный файл должен иметь расширение exe/com. Для его получения использовать компилятор tasm и компоновщик tlink .

Результат работы программы на рис. 1.

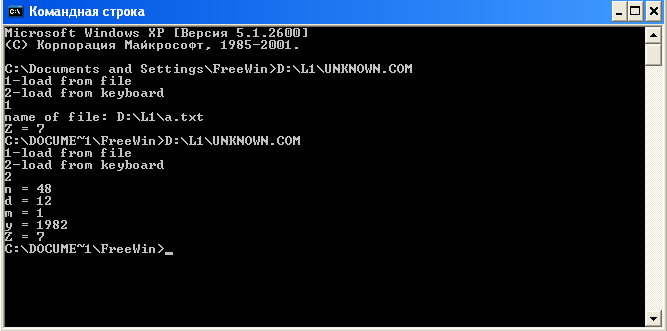


рис. 1

**Исходный код**

;#make\_COM#

CSEG segment

assume ss:CSEG, es:CSEG, ds:CSEG, cs: CSEG

ORG 100H

START:

mov dx, offset mesLoad ;загружаем меню

mov ah, 09h

int 21h

mov ah, 01h ;ждём нажатия клавиши

int 21h

cmp al, 32h ;если выбран ввод с клавиатуры

je LoadKeyBoard ;переходим на метку LoadKeyBoard

;====================================================

mov dx, offset endl ;перенос строки

mov ah, 09h

int 21h

mov dx, offset mesNameFile;запрос на ввод имени файла

mov ah, 09h

int 21h

mov dx, offset path; ;сюда сохраняем имя файла

mov ah, 0ah

int 21h

mov bx, 0 ;ставим знак конца строки

mov bl, [path+1]

mov path[bx+2], '0'

mov ax,3d00h ; открываем для чтения

lea dx,path + 2 ; DS:dx указатель на имя файла

int 21h ; в ax деcкриптор файла

jnc FileOK ; если не поднят флаг С, то ошибки открытия нет

mov dx, offset mesError

mov ah, 09h ;иначе сообщаем об ошибке

int 21h

MOV AH, 4Ch ;и выходим

INT 21h

int 20h

FileOK:

mov bx,ax ; копируем в bx указатель файла

mov Handle,ax ; копируем в Handle указатель файла

xor cx,cx ;указатели на начало файла

xor dx,dx ;указатели на начало файла

mov ax,4200h

int 21h ; идем к началу файла

mov bx, Handle ; копируем в bx указатель файла

call LoadFile ;загружаем из файла первое число

mov n, ax ;в n

call LoadFile ;загружаем из файла второе число

mov d, ax ;в d

call LoadFile ;загружаем из файла третье число

mov m, ax ;в m

call LoadFile ;загружаем из файла четвёртое число

mov y, ax ;в y

mov bx, Handle ; копируем в bx указатель файла

;close: ; закрываем файл, после чтения

mov ah,3eh

int 21h

;===================================================

jmp formula

LoadKeyBoard:

mov dx, offset endl

mov ah, 09h

int 21h

;Вводим n===============

LEA DX, nCout

MOV AH, 9h

INT 21h

;Вводим строку

LEA DX, msg

MOV AH, 0ah

INT 21h

;Ставим знак конца строки

mov bx, 0

mov bl, [msg+1]

mov msg[bx+2], '$'

call IntegerIn

mov n,ax

;переводим строку

LEA DX, endl

MOV AH, 9h

INT 21h

;Вводим d===============

LEA DX, dCout

MOV AH, 9h

INT 21h

;Вводим строку

LEA DX, msg

MOV AH, 0ah

INT 21h

;Ставим знак конца строки

mov bx, 0

mov bl, [msg+1]

mov msg[bx+2], '$'

call IntegerIn

mov d,ax

;переводим строку

LEA DX, endl

MOV AH, 9h

INT 21h

;Вводим m==================

LEA DX, mCout

MOV AH, 9h

INT 21h

;Вводим строку

LEA DX, msg

MOV AH, 0ah

INT 21h

;Ставим знак конца строки

mov bx, 0

mov bl, [msg+1]

mov msg[bx+2], '$'

call IntegerIn

mov m,ax

;переводим строку

LEA DX, endl

MOV AH, 9h

INT 21h

;Вводим y===============

LEA DX, yCout

MOV AH, 9h

INT 21h

;Вводим строку

LEA DX, msg

MOV AH, 0ah

INT 21h

;Ставим знак конца строки

mov bx, 0

mov bl, [msg+1]

mov msg[bx+2], '$'

call IntegerIn

mov y,ax

;========================================

formula:

cmp d, 15 ;

jg Dgreate ;если d>15, тогда на Dgreate

mov ax, m ;m в ax

imul n ;m\*n

mov a, ax ;результат в a

mov ax, y ;y в ax

idiv n ;y/n

mov b, ax ;результат в b

mov ax, a ;m\*n в ax

sub ax, [b] ;m\*n-y/b

mov [z], ax ;Z

jmp finish ;переход на метку finish

Dgreate:

mov ax, y ;y в ax

sub ax, m ;(y-m)

mov a, ax ;результат в a

mov ax, y ;y в ax

sub ax, d ;(y-d)

mov b, ax ;результат в b

mov ax, a ;(y-m) в ax

imul [b] ;(y-m)\*(y-d)

mov [z], ax ;Z

finish:

mov dx, offset endl ;перенос строки

mov ah, 09h

int 21h

mov dx, offset result ;Z =

mov ah, 09h

int 21h

mov ax, [z]

call IntegerOut ;выводим результат

MOV AH, 4Ch ;выход

INT 21h

int 20h

IntegerIn proc

mov x,0

mov ax,10 ;умножаем на 10 каждый цикл

mov ch, 0 ;обнуляем счётчик

mov dh, 0 ;в dl будем заносить числа в [bx]

mov cl, [msg+1] ;длина строки

lea bx, msg+2 ;первый символ

mov dl, [bx] ;заносим в [dl]

cmp dl, '-' ;если -

je GetInteger ;число целое

loopGetUnsigned:

mul x ;умножаем на 10

mov x, ax ;запоминаем результат

mov dl, [bx] ;символ в dl

sub dl, '0' ;получаем число

add x, dx ;прибавляем к результату

inc bx ;переходим к следующему числу

mov ax,10 ;восстанавливаем множитель

loop loopGetUnsigned

mov ax, x ;результат в ax

ret

GetInteger:

mov ax,10 ;умножаем на 10 каждый цикл

mov ch, 0 ;обнуляем счётчик

mov dh, 0 ;в dl будем заносить числа в [bx]

dec cl ;пропускаем '-'

inc bx ;переходим к следующему числу

loopGetInteger:

mul x ;умножаем на 10

mov x, ax ;запоминаем результат

mov dl, [bx] ;символ в dl

sub dl, '0' ;получаем число

add x, dx ;прибавляем к результату

inc bx ;переходим к следующему числу

mov ax,10 ;восстанавливаем множитель

loop loopGetInteger

mov ax, x ;результат в ax

neg ax ;меняем знак

ret

IntegerIn endp

IntegerOut proc

xor cx,cx ;обнуляем счетчик цифр

mov bx,10 ;в bx помещаем делитель

cmp ax,0 ;проверяем знак числа

jge mIntegerOut ;если неотрицательное – на m

neg ax ;иначе – меняем знак числа

push ax ;сохраняем число перед вызовом

;функции, использующей ax

mov ah,2 ;функцией 02 выводим знак '-'

mov dl,'-'

int 21h

pop ax ;восстанавливаем число в ax

mIntegerOut:

inc cx ;считаем количество

;получающихся цифр

xor dx,dx ;преобразуем делимое к 32

;разрядам

div bx ;получаем очередную цифру

push dx ;сохраняем ее в стеке

or ax,ax ;проверяем есть ли еще цифры

jnz mIntegerOut ;если да – на метку m

;при выходе из цикла в стеке лежат цифры, в cx – их

;количество

m1IntegerOut: pop dx ;извлекаем цифру из стека

add dx,'0' ;преобразуем в код символа

mov ah,2 ;функцией 02 выводим на экран

int 21h

loop m1IntegerOut ;повторяем cx раз

ret ;возвращаемся из процедуры

IntegerOut endp

LoadFile proc

mov [x], 0 ;дополнительная переменная для запоминания промежуточного результата

mov [zn], 0 ;знак числа

out\_str:

mov ah,3fh ; будем читать из файла

mov cx,1 ; 1 байт

lea dx,buf ; в память buf

int 21h

cmp ax,cx ; если достигнуть EoF или ошибка чтения

jnz close ; то закрываем файл закрываем файл

cmp [buf], 10 ;возврат каретки

je close

cmp [buf], 13 ;перенос каретки

je trash

cmp [buf], '-' ;проверяем знак

je znak

mov ax, 10 ;будем умножаеть на 10

mov bx, [x] ;последовательно каждый цикл

mul bx

mov bx, ax ;результат в bx

add bl, [buf] ;прибавляем следующее число

sub bl, '0' ;вычитаем код '0'

mov [x], bx ;сохраняем промежуточное значение

trash:

mov bx, Handle ; копируем в bx указатель файла

jmp out\_str

close:

cmp [zn], 1 ;если число отрицательное

je chan ;меняем знак

mov ax, [x] ;результат в ax

ret

chan:

mov ax, [x] ;результат в ax

neg ax ;меняем знак

ret

znak :

mov zn, 1 ;отмечаем, что число отрицательное

jmp trash ;возвращаемся

LoadFile endp

;Data

mesNameFile db 'name of file: $'

Handle dw 0

mesLoad db '1-load from file',10,13, '2-load from keyboard',10,13,'$'

mesError db 10,13,'file not found$'

msg DB 255,?,255 dup(?)

endl DB 10,13,'$'

mCout DB 'm = $'

nCout DB 'n = $'

dCout DB 'd = $'

yCout DB 'y = $'

result DB 'Z = $'

a dw 0

b dw 0

x dw 0

m dw 0

n dw 0

y dw 0

d dw 0

z dw 0

zn db 0

;path db 'a.txt',0 ; имя файла для октрытия

buf db ?

path db 255,?,255 dup(0)

CSEG ends

end START