



Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК4 «Программное обеспечение ЭВМ, информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №5

«Создание макросов для ввода и вывода данных»

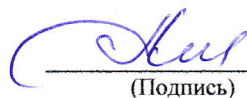
ДИСЦИПЛИНА: «Системное программирование»

Выполнил: студент гр. ИУК4-32Б


(Подпись)

(Зудин Д.В.)
(Ф.И.О.)

Проверил:


(Подпись)

(Амеличева К.А.)
(Ф.И.О.)

Дата сдачи (защиты):

16.11.2022

Результаты сдачи (защиты):

- Балльная оценка:

3 балла + 1 балл

- Оценка:

зачтено

Калуга, 2022 г.

Цель: практическое овладение навыками разработки программного кода на языке Ассемблер. Изучение приёмов разработки макроопределений.

Постановка задачи:

1. Создать макросы для ввода и вывода чисел (двух, трех и четырехзначных).
2. С использованием макросов выполнить задание, соответствующее варианту.
3. Исходные данные вводятся с клавиатуры (n, c, d).
4. Результаты выводятся на экран.

Вариант №18

Формулировка задания №1

Ввести с клавиатуры последовательность из N чисел, размером в слово. Значение N задается с клавиатуры, но должно быть не больше 15. Найти сумму квадратов всех отрицательных элементов последовательности. Результат вывести на экран.

Листинг файла io.asm

```
Set_cursor MACRO row, col      ;Макрос установки курсора
    push ax
    push bx
    push cx
    push dx

    mov bh, 3fh                ;атрибут нормальный ч/б
    mov cx, 0000                ;координаты от 00,00
    mov dx, 184fh                ;до 24,79 (весь экран)
    int 10h

    mov ah, 02                  ;Установка курсора
    mov bh, 00                  ;страница
    mov dh, row                 ;номер строки в DH
    mov dl, col                 ;номер столбца в DL
    int 10h

    pop dx
    pop cx
    pop bx
    pop ax
ENDM

mWriteStr macro string          ;Макрос вывода строки
    push ax
    push dx

    mov ah, 09h                ;Функция вывода
    mov dx, offset string
    int 21h

    pop dx
    pop ax
ENDM
```

```

Clear macro                                ;Макрос очистки экрана
    push ax
    push bx
    push cx
    push dx

    mov ah, 6h
    mov al, 0h
    mov bh, 3fh
    mov cx, 0000h
    mov dx, 184fh
    int 10h

    pop dx
    pop cx
    pop bx
    pop ax
ENDM

mReadAX10 macro buffer, sizee ;Макрос ввода 10-чного числа в регистр AX
local input, startOfConvert, endOfConvert
    push bx                ;Данные в стек
    push cx
    push dx

input:
    mov [buffer], sizee    ;Задаём размер буфера
    mov dx, offset [buffer] ;Поместить в регистр dx строку по адресу buffer
    mov ah, 0Ah            ;Чтение строки из консоли
    int 21h                ;Прерывание DOS

    mov ah, 02h            ;Вывод символа на экран
    mov dl, 0Dh            ;Перевод каретки на новую строку
    int 21h                ;Прерывание DOS

    mov ah, 02h            ;Вывод символа на экран
    mov dl, 0Ah            ;Чтение строки из консоли
    int 21h                ;Прерывание DOS

    xor ah, ah             ;Очистка регистра ah
    cmp al, [buffer][1]    ;Проверка на пустую строку
    jz input               ;Переход, если строка пустая

    xor cx, cx             ;Очистка регистра cx
    mov cl, [buffer][1]    ;инициализация переменной-счётчика

    xor ax, ax             ;Очистка регистра ax
    xor bx, bx             ;Очистка регистра bx
    xor dx, dx             ;Очистка регистра dx
    mov bx, offset [buffer][2] ;Поместить начало строки в регистр bx
    cmp [buffer][2], '-'   ;Проверка на знак числа
    jne startOfConvert     ;Переход, если число неотрицательное
    inc bx                 ;Инкремент регистра bx
    dec cl                 ;Декремент регистра-счётчика cl

startOfConvert:
    mov dx, 10             ;Поместить в регистр ax число 10
    mul dx                 ;Умножение на 10 перед сложением с младшим
разрядом
    cmp ax, 8000h          ;Проверка числа на выход за границы
    jae input              ;Переход, если число выходит за границы

```

```

    mov dl, [bx]           ;Поместить в регистр dl следующий символ
    sub dl, '0'           ;Перевод его в числовой формат

    add ax, dx             ;Прибавляем его к конечному результату
    cmp ax, 8000h         ;Проверка числа на выход за границы
    jae input             ;Переход, если число выходит за границы

    inc bx                ;Переход к следующему символу
    loop startOfConvert   ;Цикл

    cmp [buffer][2], '-'  ;Проверка на знак числа
    jne endOfConvert      ;Переход, если число неотрицательное
    neg ax                ;Инвертирование числа

endOfConvert:
    pop dx                ;Данные из стека
    pop cx
    pop bx
endm

mWriteAX10 macro          ;Макрос вывода 10-чного числа из регистра AX
local convert, write
    push ax               ;Данные в стек
    push bx
    push cx
    push dx
    push di

    mov cx, 10            ;cx - основание системы счисления
    xor di, di            ;di - количество цифр в числе
    or ax, ax             ;Проверка числа на ноль
    jns convert           ;Переход, если число положительное
    push ax               ;Регистр ax в стек

    mov dx, '-'           ;Поместить в регистр dx символ '-'
    mov ah, 02h           ;Вывод символа на экран
    int 21h              ;Прерывание DOS

    pop ax                ;Регистр ax из стека
    neg ax                ;Инвертирование отрицательного числа

convert:
    xor dx, dx            ;Очистка регистра dx

    div cx                ;После деления dl = остатку от деления ax на cx
    add dl, '0'           ;Перевод в символьный формат
    inc di                ;Увеличение количества цифр в числе на 1
    push dx               ;Регистр dx в стек

    or ax, ax             ;Проверка числа на ноль
    jnz convert           ;Переход, если число не равно нулю

write:
    pop dx                ;dl = очередной символ

    mov ah, 02h           ;Вывод символа на экран
    int 21h              ;Прерывание DOS
    dec di                ;Повторение, пока di != 0
    jnz write

    pop di                ;Данные из стека
    pop dx
    pop cx

```

```

        pop bx
        pop ax
endm

```

Листинг программы для задания №1

```

include io.asm                ;Подключение файла макросов
.model small                  ;Количество сегментов - 2
.stack 100h                   ;Размер стека - 256 б
.data                         ;Сегмент данных
    mes_n      db 'Enter the size of the array N: ', '$'
    mes_res    db 'Result: ', '$'
    new_line    db 13,10,'$'
    _enter      db 'Enter $'
    arr_num     db ' array number: $'
    arr_eq      db 'arr: $'
    space       db ' $'
    comma       db ', $'
    o_sq_br     db '[$'
    c_sq_br     db ']$'
    buf db ?                  ;Буфер для ввода числа с клавиатуры
    n  dw ?                  ;N - размер массива
    arr dw 15 dup (?)        ;Массив arr
    res dw ?                 ;Результат

.code                         ;Сегмент кода
start:
    mov ax, @data             ;Поместить адрес сегмента данных
    mov ds, ax                ;в регистр ds

    xor ax, ax                 ;Обнуление регистра ax
    mWriteStr mes_n            ;Введите размер массива N
    mReadAX10 buf, 3           ;Ввод числа n в регистр ax
    mWriteStr new_line         ;Вывод новой строки
    mov bx, ax                 ;Поместить в регистр bx значение регистра ax
    mov n, bx                  ;Поместить в переменную n значение регистра bx

    mov cx, n                  ;Поместить в регистр cx значение переменной n
    mov di, 2

11:
    mWriteStr _enter           ;Вывод Enter
    mov si, di                 ;Поместить в регистр si значение регистра di
    inc si                     ;Увеличить si на 1
    mov ax, si                 ;Поместить в регистр ax значение регистра si
    push bx                    ;Поместить bx в стек
    mov bx, 2                  ;
    xor dx, dx                 ;
    div bx                     ;Арифметика индекса для вывода
    mov si, ax                 ;
    pop bx                     ;Извлечь bx из стека

    mov ax, si                 ;Поместить в регистр ax значение регистра si
    mWriteAX10                 ;Вывод номера элемента
    mWriteStr arr_num          ;Вывод array number
    mReadAX10 buf, 7           ;Ввод элемента массива в регистр ax
    mov arr[di], ax            ;Поместить в i-ый элемент массива значение
    регистра ax
    inc di                     ;Увеличить di на 2
    inc di

    dec cx                     ;Уменьшить cx на 1
    cmp cx, 0                  ;Сравнение cx с 0

```

```

        je endl1          ;Переход, если cx = 0
        jmp l1            ;Безусловный переход

endl1:
        mWriteStr arr_eq
        mWriteStr o_sq_br
        mov di, 2
        mov n, bx         ;Поместить в переменную n значение регистра bx
        mov cx, n         ;Поместить в регистр cx значение переменной n

12:
        mov ax, arr[di]   ;Поместить в регистр ax i-ый элемент массива
        inc di            ;Увеличить di на 1
        inc di
        mWriteAX10        ;Вывод элемента массива

        dec cx            ;Уменьшить cx на 1
        cmp cx, 0         ;Сравнение cx с 0
        jne sep           ;Переход, если cx != 0
        jmp next          ;Переход, если cx = 0

sep:
        mWriteStr comma
        mWriteStr space   ;Разделители символов при выводе
        jmp 12

next:
        mWriteStr c_sq_br
        mWriteStr new_line
        mov di, 2
        mov n, bx         ;Поместить в переменную n значение регистра bx
        mov cx, n         ;Поместить в регистр cx значение переменной n
        xor bx, bx        ;Обнуление регистра bx

13:
        cmp arr[di], 0    ;Сравнение элемента массива с 0
        jl square         ;Переход, если arr[i] < 0
        jmp endl          ;Безусловный переход

square:
        mov ax, arr[di]   ;Поместить в регистр ax i-ый элемент массива
        mul ax            ;Поместить в регистр ax arr[i]^2
        add bx, ax        ;Прибавить к bx значение регистра ax

endl:
        inc di
        inc di
        dec cx            ;Уменьшить cx на 1
        cmp cx, 0         ;Сравнение cx с 0
        jne 13            ;Переход, если cx != 0

        mov ax, bx        ;Поместить в регистр ax значение регистра bx
        mWriteStr mes_res
        mWriteAX10        ;Вывод результата на экран
        mov res, ax       ;Результат

        mov ax, 4c00h     ;Завершение
        int 21h           ;программы
end start                ;Закреть программу

```

Результат выполнения программы для задания №1

```
Enter the size of the array N: 5
Enter 1 array number: -2
Enter 2 array number: 2
Enter 3 array number: 3
Enter 4 array number: -3
Enter 5 array number: -4
arr: [-2, 2, 3, -3, -4]
Result: 29
```

Формулировка задания №2

Измените программу, разработанную на практическом занятии №2, заменив повторяющиеся действия макросами вывода строки на экран и установки курсора в заданную позицию.

Листинг программы для задания №2

```
include io.asm ;Подключение файла макросов
.model small
.stack 100h
.data
    ru1 db "Труден лишь первый шаг $"
    ru2 db "Варрон Марк Теренций $"
    ru3 db "116-27 гг. до н.э. $"
    en1 db "| Hard is the first step $"
    en2 db "| Varro, Mark Terence $"
    en3 db "| 116-27 years BC $"
    surname db "Zudin $"
    grp db "IUK4-32B $"
    faculty db "IUK $"
    symbol db '!'
.code
start:
    mov ax, @data
    mov ds, ax

    Clear ;Очистка экрана

    Set_cursor 0, 0
    mWriteStr surname ;Вывод фамилии

    Set_cursor 0, 72
    mWriteStr grp ;Вывод группы

    Set_cursor 10, 17
    mWriteStr ru1 ;Вывод первой строки
    Set_cursor 10, 37
    mWriteStr en1

    Set_cursor 11, 17
    mWriteStr ru2 ;Вывод второй строки
    Set_cursor 11, 37
    mWriteStr en2

    Set_cursor 12, 17
```

```

mWriteStr ru3          ;Вывод третьей строки
Set_cursor 12, 37
mWriteStr en3

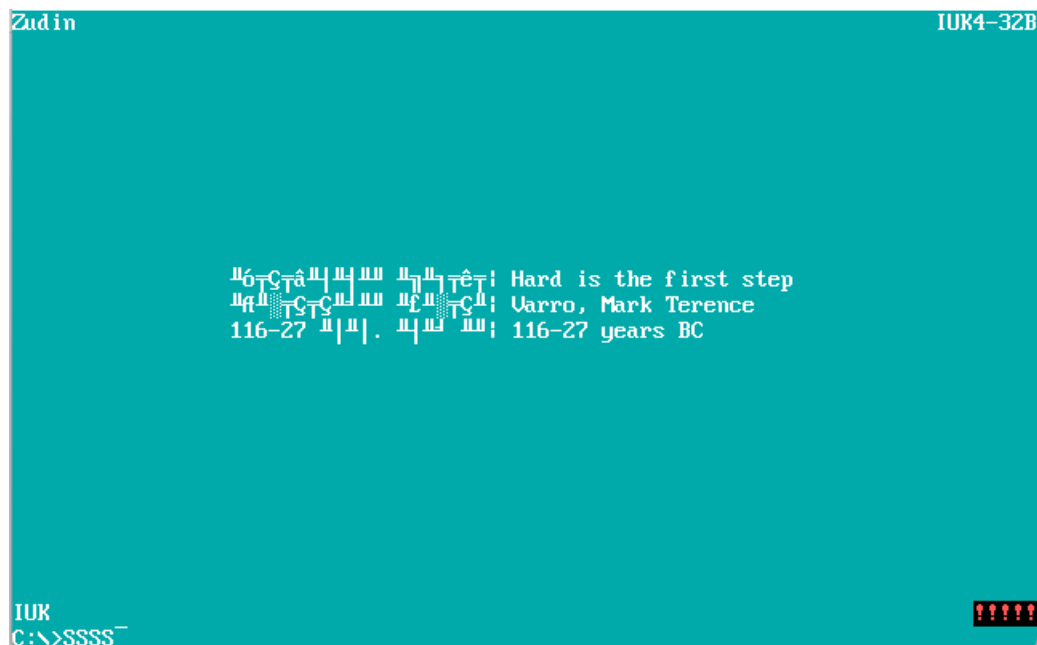
Set_cursor 23, 0
mWriteStr faculty      ;Вывод факультета

Set_cursor 23, 75
mov ah, 09h
mov al, '!'
mov bl, 10001100b
mov cx, 5              ;Вывод пяти знаков "!"
int 10h

mov ax, 4c00h
int 21h
end start

```

Результат выполнения программы для задания №2



Формулировка задания №3

Измените программу, разработанную в лабораторной работе №3 «Выполнение арифметических операций над числами без знака и со знаком», дополнив её макросами «Ввода целого числа в регистр AX в 10-ричной системе счисления», и «Вывода целого числа в регистр AX в 10-ричной системе счисления», приведенными ниже.

Листинг программы для задания №3

```

include io.asm          ;Подключение файла макросов
.model small            ;Количество сегментов - 2
.stack 100h             ;Размер стека - 256 б
.data                   ;Сегмент данных
    mes_a db 'Enter the number a: ', '$'
    mes_b db 'Enter the number b: ', '$'

```



```

mes_x db 'Enter the number x: ', '$'
mes_y db 'Result y: ', '$'
buf db ? ;Буфер для ввода числа с клавиатуры
x dw ? ;Переменная x
a dw ? ;Переменная a
b dw ? ;Переменная b
y dw ? ;Переменная y - результат вычислений

.code ;Сегмент кода
start:
    mov ax, @data ;Поместить адрес сегмента данных
    mov ds, ax ;в регистр ds

    xor ax, ax ;Обнуление регистра ax
    mWriteStr mes_a ;Введите число a
    mReadAX10 buf, 5 ;Ввод числа a в регистр ax
    mov bx, ax ;Поместить в регистр bx значение регистра ax

    mWriteStr mes_b ;Введите число b
    mReadAX10 buf, 5 ;Ввод числа b в регистр ax
    mov cx, ax ;Поместить в регистр cx значение регистра ax

    mWriteStr mes_x ;Введите число x
    mReadAX10 buf, 5 ;Ввод числа x в регистр ax
    mov dx, ax ;Поместить в регистр dx значение регистра ax

    mov a, bx ;Поместить в переменную a значение регистра bx
    mov b, cx ;Поместить в переменную b значение регистра cx
    mov x, dx ;Поместить в переменную x значение регистра dx
    xor ax, ax ;Обнуление регистра ax
    xor bx, bx ;Обнуление регистра bx
    xor cx, cx ;Обнуление регистра cx
    xor dx, dx ;Обнуление регистра dx

    mov ax, a ;Поместить в регистр ax значение переменной a
    mul a ;Поместить в регистр ax значение a^2
    add ax, x ;Поместить в регистр ax значение (x + a^2)
    mov bx, ax ;Поместить в регистр bx значение регистра ax

    mov ax, b ;Поместить в регистр ax значение переменной b
    mov dl, 3 ;Поместить в регистр dl значение 3
    mul dl ;Поместить в регистр ax значение 3b

    xchg bx, ax ;Поменять местами значения регистров ax и bx
    xor dx, dx ;Обнуление регистра dx
    idiv bx ;Поместить в регистр ax значение (x + a^2) / 3b
    mov bx, ax ;Поместить в регистр bx значение регистра ax

    mov ax, x ;Поместить в регистр ax значение переменной x
    mul x ;Поместить в регистр ax значение x^2
    mov cx, 2 ;Поместить в регистр cx значение 2
    xor dx, dx ;Обнуление регистра dx
    div cx ;Поместить в регистр ax значение x^2 / 2

    xchg bx, ax ;Поменять местами значения регистров ax и bx
    sub ax, bx ;Поместить в регистр ax значение ((x + a^2) /
3b) - (x^2 / 2)

    mWriteStr mes_y ;Результат
    mWriteAX10 ;Вывод результата
    mov y, ax ;Поместить в переменную y значение регистра ax

    mov ax, 4c00h ;Завершение

```

```
int 21h ;программы  
end start ;Закреть программу
```

Результат выполнения программы для задания №3

```
Enter the number a: 4  
Enter the number b: 1  
Enter the number x: 5  
Result y: -5
```

Выводы:

В ходе выполнения работы были сформированы практические навыки разработки программного кода на языке Ассемблера; изучены приёмы разработки макроопределений.