



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»**

## **ПРАКТИЧЕСКАЯ РАБОТА №5**

### **«Макросредства Ассемблера. Операции ввода/вывода в Ассемблере»**

**ДИСЦИПЛИНА: «Системное программирование»**

Выполнил: студент гр. ИУК4-31Б

  
(подпись)

( Суриков Н.С. )  
(Ф.И.О.)

Проверил:

  
(подпись)

( Амеличева К.А. )  
(Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

**Цель:** Практическое овладение навыками разработки программного кода на языке Ассемблер с использованием макроопределения.

### Задачи:

1. Изучение структуры и операторов макроопределения.
2. Написание макроопределения ввода/вывода.

**Задание 1.** Измените программу, разработанную на практическом занятии №2, заменив повторяющиеся действия макросами вывода строки на экран и установки курсора в заданную позицию.

### Листинг программы 1:

```
1  .model small      ; Определяет модель памяти как "small", где код и данные помещаются в один
    сегмент.
2  .stack 100h       ; Определяет стек размером 256 байт (100h).
3
4  ; Начало сегмента данных.
5  .data
6  message db 'Hard is the first step','$', 10, 13
7  message1 db 'Varro, Mark Ternce','$', 10, 13
8  message2 db '116-27 years. BC','$', 10, 13
9  message3 db 'Surikov','$',10,13
10 message4 db 'IUK4-31B','$',10,13
11 message5 db 'IUK4','$',10,13
12
13 .code
14 Set_cursor MACRO row, col ;Макрос установки курсора
15     push ax
16     push bx
17     push cx
18     push dx
19
20     mov ah, 02 ; Установка курсора
21     mov dh, row ; номер строки в DH
22     mov dl, col ; номер столбца в DL
23     mov bh, 0 ; Указывает страницу экрана (0).
24     int 10h
25
26     pop dx
27     pop cx
28     pop bx
29     pop ax
30 ENDM
31
32 mWriteStr macro string ;Макрос вывода строки
33     push ax
34     push dx
35
36     mov ah, 09h
37     mov dx, offset string
38     int 21h
39
40     pop dx
41     pop ax
42 ENDM
43
44 Clear macro ;Макрос очистки экрана
```

```

45     push ax
46     push bx
47     push cx
48     push dx
49
50     mov ax, 0600h      ; Подготавливает код для очистки экрана (функция 0).
51     mov bh, 2Ch       ; Устанавливает цвет фона и шрифта
52     mov cx, 0000      ; Указывает количество строк для очистки (все).
53     mov dx, 184FH     ; Указывает адрес экрана (184FH – адрес видеопамати).
54     int 10h           ; Вызывает прерывание BIOS для выполнения очистки экрана.
55
56     pop dx
57     pop cx
58     pop bx
59     pop ax
60     ENDM
61
62     start:             ; Метка начала программы.
63     mov ax, @data      ; Загружает адрес сегмента данных в регистр AX.
64     mov ds, ax         ; Устанавливает сегмент данных (DS) в значение AX.
65
66     Clear
67
68     ; Центральное сообщение
69
70     Set_cursor 10, 30
71     mWriteStr message
72
73     Set_cursor 11, 30
74     mWriteStr message1
75
76     Set_cursor 12, 30
77     mWriteStr message2
78
79     ; Вывод информации по углам экрана
80
81     ; Левый верхний угол
82     Set_cursor 0, 0
83     mWriteStr message3
84
85     ; Правый верхний угол
86     Set_cursor 0, 72
87     mWriteStr message4
88
89     ; Левый нижний угол
90     Set_cursor 24, 0
91     mWriteStr message5
92
93     ; Правый нижний угол
94     Set_cursor 24, 75
95
96     mov ah, 09h        ; Подготавливает функцию вывода строки.
97     mov al, '!'        ; Выводимый символ
98     mov bl, 10101100b  ; Атрибут(цвет, фон, мерцание)
99     mov cx, 5          ; Коэффициент повторения
100    int 10h            ; Вызывает прерывание BIOS для установки курсора.
101
102    mov ah, 7h          ; Подготавливает функцию для ожидания нажатия клавиши.
103    int 21h            ; Вызывает прерывание DOS для ожидания нажатия клавиши.
104
105    mov ax, 4c00h       ; Завершает программу и возвращает управление операционной
системе.
106    int 21h            ; Вызывает прерывание DOS для завершения программы.
107
108    end                start

```

## Результат работы программы 1:



**Задание 2.** Измените программу, разработанную в лабораторной работе №3 «Выполнение арифметических операций над числами без знака и со знаком», дополнив ее 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 2:

```
1  .model small
2  .stack 100h
3  .data
4      buffer db 5, 0, 5 dup(0)           ; Буфер для ввода числа (максимум 5 цифр)
5      mes_a  db 'Enter the number a: ', '$'
6      mes_b  db 'Enter the number b: ', '$'
7      mes_c  db 'Enter the number c: ', '$'
8      mes_d  db 'Enter the number d: ', '$'
9      a      dw 0                         ; Переменная для хранения первого числа
10     b      dw 0                         ; Переменная для хранения второго числа
11     c      dw 0                         ; Переменная для хранения третьего числа
12     d      dw 0                         ; Переменная для хранения четвертого числа
13
14  .code
15     ; Макрос для вычисления  $y = 2*b + 2*a$ 
16     mCalc_Y1 MACRO a, b
17         push bx           ; Данные в стек
18         push cx
19         push dx
20
21         mov ax, a         ; загрузить a в ax
22         mov bx, b         ; загрузить b в bx
23         shl ax, 1         ; ax = 2*a
```

```

24         shl     bx, 1      ; bx = 2*b
25         add     ax, bx     ; ax = 2*a + 2*b
26
27         pop     dx
28         pop     cx
29         pop     bx
30     ENDM
31
32     ; Макрос для вычисления  $y = ((a + 3*b) / c) + 4$ 
33     mCalc_Y2 MACRO a, b, c
34         push    bx          ; Данные в стек
35         push    cx
36         push    dx
37
38         mov     bx, a       ; загрузить a в bx
39         mov     ax, b       ; загрузить b в ax
40         mov     cx, 3       ; загрузить 3 в cx
41
42         imul    cx          ; ax = b * 3 (ax = 3*b)
43         add     ax, bx      ; ax = a + 3*b
44
45         xor     dx, dx      ; очистить для деления
46         xor     cx, cx
47
48         mov     cx, c       ; загрузить c в cx
49         cmp     cx, 0
50
51         je      DIV_BY_ZERO ; если c = 0, переход к обработке деления на ноль
52         cwd
53         idiv    cx          ; деление ax на c, результат в ax/dx
54         add     ax, 4       ; добавить 4 к результату
55         jmp     END_CALC_Y2
56     DIV_BY_ZERO:
57         xor     ax, ax      ; если деление на ноль, сохранить 0 в res
58     END_CALC_Y2:
59         pop     dx
60         pop     cx
61         pop     bx
62     ENDM
63
64     mReadAX macro buffer, sizee ;Макрос ввода 10-чного числа в
регистр AX
65         local input, startOfConvert, endOfConvert
66         push    bx          ;Данные в стек
67         push    cx
68         push    dx
69     input:
70         mov     [buffer], sizee ;Задаём размер буфера
71         mov     dx, offset [buffer] ;Поместить в регистр dx строку по
адресу buffer
72         mov     ah, 0Ah      ;Чтение строки из консоли
73         int     21h          ;Прерывание DOS
74
75         mov     ah, 02h      ;Вывод символа на экран
76         mov     dl, 0Dh      ;Перевод каретки на новую строку
77         int     21h          ;Прерывание DOS
78
79         mov     ah, 02h      ;Вывод символа на экран
80         mov     dl, 0Ah      ;Чтение строки из консоли
81         int     21h          ;Прерывание DOS
82
83         xor     ah, ah       ;Очистка регистра ah
84         cmp     al, [buffer][1] ;Проверка на пустую строку
85         jz      input       ;Переход, если строка пустая
86
87         xor     cx, cx       ;Очистка регистра cx
88         mov     cl, [buffer][1] ;инициализация переменной-
счётчика
89
90         xor     ax, ax       ;Очистка регистра ax

```

```

91          xor    bx, bx                ;Очистка регистра bx
92          xor    dx, dx                ;Очистка регистра dx
93
94          mov    bx, offset [buffer][2] ;Поместить начало строки в
регистр bx
95          cmp    [buffer][2], '-'      ;Проверка на знак числа
96          jne    startOfConvert        ;Переход, если число
неотрицательное
97          inc    bx                    ;Инкремент регистра bx
98          dec    cl                    ;Декремент регистра-счетчика cl
99          startOfConvert:
100         mov    dx, 10                ;Поместить в регистр ax число 10
101         mul    dx                    ;Умножение на 10 перед сложением
с младшим разрядом
102         cmp    ax, 8000h              ;Проверка числа на выход за
границы
103         jae    input                  ;Переход, если число выходит за
границы
104         mov    dl, [bx]                ;Поместить в регистр dl следующий
символ
105         sub    dl, '0'                 ;Перевод его в числовой формат
106         add    ax, dx                  ;Прибавляем его к конечному
результату
107         cmp    ax, 8000h              ;Проверка числа на выход за
границы
108         jae    input                  ;Переход, если число выходит за
границы
109         inc    bx                    ;Переход к следующему символу
110         loop   startOfConvert          ;Цикл
111         cmp    [buffer][2], '-'      ;Проверка на знак числа
112         jne    endOfConvert           ;Переход, если число
неотрицательное
113         neg    ax                    ;Инвертирование числа
114         endOfConvert:
115         pop     dx                    ;Данные из стека
116         pop     cx
117         pop     bx
118     endm
119
120     mWriteAX macro                    ;Макрос вывода 10-чного числа из регистра AX
121         local convert, write
122         push    ax                    ;Данные в стек
123         push    bx
124         push    cx
125         push    dx
126         push    di
127
128         mov     cx, 10                ;cx - основание системы счисления
129         xor     di, di                ;di - количество цифр в числе
130         or      ax, ax                ;Проверка числа на ноль
131         jns     convert               ;Переход, если число положительное
132         push    ax                    ;Регистр ax в стек
133         mov     dx, '-'               ;Поместить в регистр dx символ '-'
134         mov     ah, 02h               ;Вывод символа на экран
135         int     21h                   ;Прерывание DOS
136         pop     ax                    ;Регистр ax из стека
137         neg     ax                    ;Инвертирование отрицательного числа
138     convert:
139         xor     dx, dx                ;Очистка регистра dx
140         div     cx                    ;После деления dl = остатку от деления ax на cx
141         add     dl, '0'               ;Перевод в символьный формат
142         inc     di                    ;Увеличение количества цифр в числе на 1
143         push    dx                    ;Регистр dx в стек
144         or      ax, ax                ;Проверка числа на ноль
145         jnz     convert               ;Переход, если число не равно нулю
146     write:
147         pop     dx                    ;dl = очередной символ
148         mov     ah, 02h               ;Вывод символа на экран
149         int     21h                   ;Прерывание DOS
150         dec     di                    ;Повторение, пока di != 0

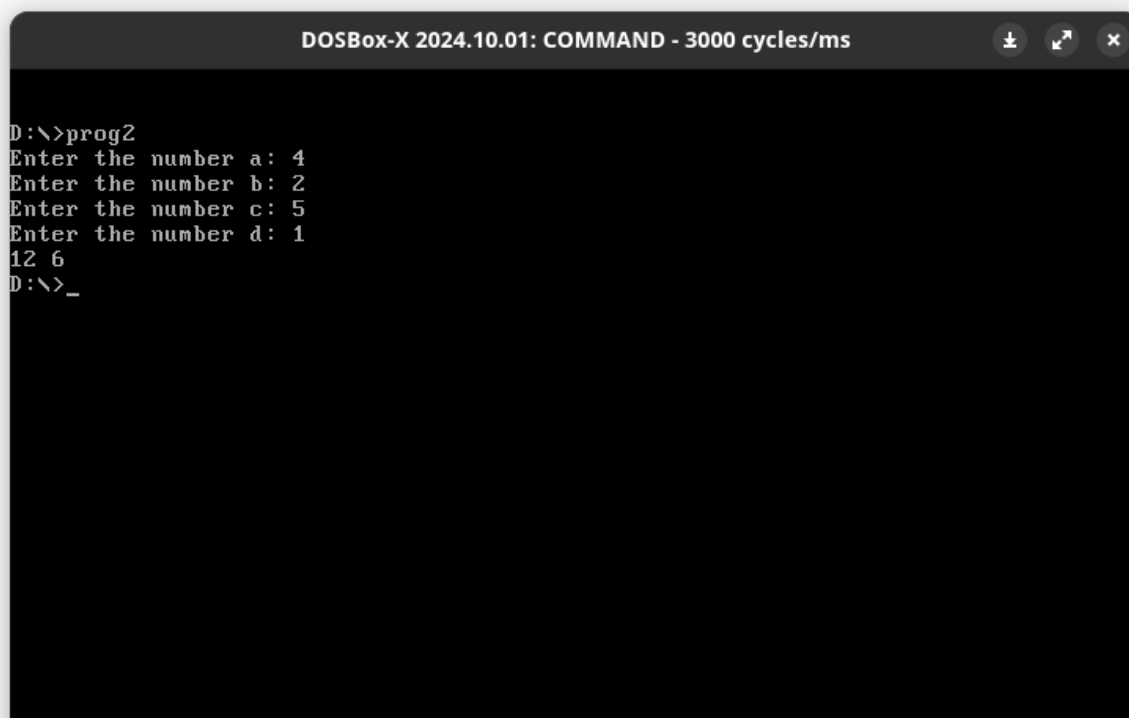
```

```

151             jnz     write
152
153             pop     di                ;Данные из стека
154             pop     dx
155             pop     cx
156             pop     bx
157             pop     ax
158     endm
159
160     mWriteStr macro string                ;Макрос вывода строки
161             push ax
162             push dx
163
164             mov     ah, 09h
165             mov     dx, offset string
166             int     21h
167
168             pop     dx
169             pop     ax
170     ENDM
171
172     start:
173             mov     ax, @data
174             mov     ds, ax
175
176     ; Ввод переменной a
177             xor     ax, ax
178             mWriteStr     mes_a
179             mReadAX      buffer, 5
180             mov     a, ax
181
182     ; Ввод переменной b
183             xor     ax, ax
184             mWriteStr     mes_b
185             mReadAX      buffer, 5
186             mov     b, ax
187
188     ; Ввод переменной c
189             xor     ax, ax
190             mWriteStr     mes_c
191             mReadAX      buffer, 5
192             mov     c, ax
193
194     ; Ввод переменной d
195             xor     ax, ax
196             mWriteStr     mes_d
197             mReadAX      buffer, 5
198             mov     d, ax
199
200
201             xor     ax, ax
202
203
204             mCalc_Y1  a, b ; Пример 1:  $y = 2*b + 2*a$ 
205             mWriteAX; Вывод результата
206
207             mCalc_Y2 a, b, c ; Пример 2:  $y = ((a + 3*b) / c) + 4$ 
208             mWriteAX      ; Вывод результата
209
210     ; Завершение программы
211             mov     ax, 4c00h
212             int     21h
213     end start

```

## Результат работы программы 2:



```
DOSBox-X 2024.10.01: COMMAND - 3000 cycles/ms
D:\>prog2
Enter the number a: 4
Enter the number b: 2
Enter the number c: 5
Enter the number d: 1
12 6
D:\>_
```

**Вывод:** в ходе выполнения практического задания были получены навыки разработки программного кода на языке Ассемблер с использованием макросов.