|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 вывод данных»**

**ДИСЦИПЛИНА: «Системное программирование»**



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. ИУК4-31Б | |  |  | ( | Суриков Н. С. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |
| Проверил: | |  |  | ( | Амеличева К. А. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |

|  |  |
| --- | --- |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | |
|  | - Балльная оценка:  - Оценка: |

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

**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 №3**

Ввести с клавиатуры последовательность из N чисел, размером в слово. Значение N задается с клавиатуры, но должно быть не больше 10. Найти, сколько положительных, отрицательных и нулевых элементов введенной последовательности удовлетворяет условию: c<=a[i]<=d. Значение c, d задается с клавиатуры. Результат вывести на экран.

**Словесный алгоритм:**

**Макрос ввода**

1. Чтение ввода:

Установить размер буфера.

Поместить адрес буфера в регистр DX.

Вызвать прерывание 21h для чтения строки из консоли.

Вывести перевод каретки и новую строку.

1. Проверка на пустую строку:
   * Если введённая строка пустая, вернуться к началу ввода.
2. Подготовка к конвертации:
   * Очистить регистры AX, BX, DX.
   * Поместить адрес начала строки в регистр BX.
   * Проверить, начинается ли строка с символа '-' (отрицательное число). Если да, перейти к следующему символу.
3. Конвертация строки в число:
   * Установить основание (10) в регистр DX.
   * В цикле:
   * Умножить текущее значение AX на 10.
   * Проверить, не превышает ли значение AX предел (8000h). Если да, вернуться к вводу.
   * Получить следующий символ из строки, преобразовать его в числовой формат и добавить к AX.
   * Повторять до тех пор, пока не будут обработаны все символы.
4. Обработка знака числа:
   * Если число отрицательное, инвертировать значение в AX.

**Макрос вывода**

1. Проверка числа:
   * Установить основание (10) в CX.
   * Проверить, является ли число нулём. Если да, перейти к выводу.
2. Обработка отрицательного числа:
   * Если число отрицательное, вывести символ '-' и инвертировать значение в AX.
3. Конвертация числа в строку:
   * В цикле:
   * Разделить AX на 10, сохраняя остаток в DX.
   * Преобразовать остаток в символьный формат и сохранить его.
   * Увеличить счётчик цифр.
   * Повторять, пока AX не станет равным нулю.
4. Вывод цифр:
   * В цикле:
   * Извлекать символы из стека и выводить их по одному на экран.

**Листинг файла макросов:**

1 *; Макрос ввода 10-чного числа в регистр АХ*

2 mReadAX macro buffer, sizee

3 local input, startOfConvert, endOfConvert

4 push bx *; Данные в стек*

5 push cx

6 push dx

7 input:

8 mov [buffer], sizee *; Задаём размер буфера*

9 mov dx, offset [buffer] *; Поместить в регистр dx строку по адресу buffer*

10 mov ah, 0Ah *; Чтение строки из консоли*

11 int 21h *; Прерывание DOS*

12

13 mov ah, 02h *; Вывод символа на экран*

14 mov dl, 0Dh *; Перевод каретки на новую строку*

15 int 21h *; Прерывание DOS*

16

17 mov ah, 02h *; Вывод символа на экран*

18 mov dl, 0Ah *; Чтение строки из консоли*

19 int 21h *; Прерывание DOS*

20

21 xor ah, ah *; Очистка регистра ah*

22 cmp al, [buffer][1] *; Проверка на пустую строку*

23 jz input *; Переход, если строка пустая*

24

25 xor cx, cx *; Очистка регистра cx*

26 mov cl, [buffer][1] *; инициализация переменной-счётчика*

27

28 xor ax, ax *; Очистка регистра ax*

29 xor bx, bx *; Очистка регистра bx*

30 xor dx, dx *; Очистка регистра dx*

31

32 mov bx, offset [buffer][2] *; Поместить начало строки в регистр bx*

33 cmp [buffer][2], '-' *; Проверка на знак числа*

34 jne startOfConvert *; Переход, если число неотрицательное*

35 inc bx *; Инкремент регистра bx*

36 dec cl *; Декремент регистра-счетчика cl*

37 startOfConvert:

38 mov dx, 10 *; Поместить в регистр ax число 10*

39 mul dx *; Умножение на 10 перед сложением с младшим разрядом*

40 cmp ax, 8000h *; Проверка числа на выход за границы*

41 jae input *; Переход, если число выходит за границы*

42 mov dl, [bx] *; Поместить в регистр dl следующий символ*

43 sub dl, '0' *; Перевод его в числовой формат*

44 add ax, dx *; Прибавляем его к конечному результату*

45 cmp ax, 8000h *; Проверка числа на выход за границы*

46 jae input *; Переход, если число выходит за границы*

47 inc bx *; Переход к следующему символу*

48 loop startOfConvert *; Цикл*

49 cmp [buffer][2], '-' *; Проверка на знак числа*

50 jne endOfConvert *; Переход, если число неотрицательное*

51 neg ax *; Инвертирование числа*

52 endOfConvert:

53 pop dx *; Данные из стека*

54 pop cx

55 pop bx

56 endm

57

58 *; Макрос вывода 10-чного числа из регистра AX*

59 mWriteAX macro

60 local convert, write

61 push ax *; Данные в стек*

62 push bx

63 push cx

64 push dx

65 push di

66

67 mov cx, 10 *; cx - основание системы счисления*

68 xor di, di *; di - количество цифр в числе*

69 or ax, ax *; Проверка числа на ноль*

70 jns convert *; Переход, если число положительное*

71 push ax *; Регистр ax в стек*

72 mov dx, '-' *; Поместить в регистр dx символ '-'*

73 mov ah, 02h *; Вывод символа на экран*

74 int 21h *; Прерывание DOS*

75 pop ax *; Регистр ax из стека*

76 neg ax *; Инвертирование отрицательного числа*

77 convert:

78 xor dx, dx *; Очистка регистра dx*

79 div cx *; После деления dl = остатку от деления ax на cx*

80 add dl, '0' *; Перевод в символьный формат*

81 inc di *; Увеличение количества цифр в числе на 1*

82 push dx *; Регистр dx в стек*

83 or ax, ax *; Проверка числа на ноль*

84 jnz convert *; Переход, если число не равно нулю*

85 write:

86 pop dx *; dl = очередной символ*

87 mov ah, 02h *; Вывод символа на экран*

88 int 21h *; Прерывание DOS*

89 dec di *; Повторение, пока di != 0*

90 jnz write

91

92 pop di *; Данные из стека*

93 pop dx

94 pop cx

95 pop bx

96 pop ax

97 endm

98

99 *; Макрос вывода строки*

100 mWriteStr macro string

101 push ax

102 push dx

103

104 mov ah, 09h

105 mov dx, offset string

106 int 21h

107

108 pop dx

109 pop ax

110 endm

111

112 mClear macro *;Макрос очистки экрана*

113 push ax

114 push bx

115 push cx

116 push dx

117

118 mov ax,0600h *; Подготавливает код для очистки экрана (функция 0).*

119 mov bh, 4Ch *; Устанавливает цвет фона и шрифта*

120 mov cx, 0000 *; Указывает количество строк для очистки (все).*

121 mov dx, 184FH *; Указывает адрес экрана (184FH — адрес видеопамяти).*

122 int 10h *; Вызывает прерывание BIOS для выполнения очистки экрана.*

123

124 pop dx

125 pop cx

126 pop bx

127 pop ax

128 ENDM

129

130 mSetCursor MACRO row, col *;Макрос установки курсора*

131 push ax

132 push bx

133 push cx

134 push dx

135

136 mov ah, 02 *; Установка курсора*

137 mov dh, row *; номер строки в DH*

138 mov dl, col *; номер столбца в DL*

139 mov bh, 0 *; Указывает страницу экрана (0).*

140 int 10h

141

142 pop dx

143 pop cx

144 pop bx

145 pop ax

146 ENDM

**Листинг программы:**

1 include macroses.asm *; Подключение файла с макросами*

2

3 .model small

4 .stack 100h

5 .data

6 mes\_n db 'Enter the size of the array N (max 10): $'

7 mes\_c db 'Enter the value of C: $'

8 mes\_d db 'Enter the value of D: $'

9 mes\_num db 'Enter array element: $'

10 mes\_pos db 'Number of positive elements: $'

11 mes\_neg db 'Number of negative elements: $'

12 mes\_zero db 'Number of zero elements: $'

13 new\_line db 13, 10, '$' *; Перевод строки*

14 buf db 7 dup(0) *; Буфер для чтения ввода*

15 n dw ? *; Переменная для хранения размера массива*

16 c dw ? *; Переменная для хранения значения C*

17 d dw ? *; Переменная для хранения значения D*

18 arr dw 10 dup(?) *; Массив из 10 элементов (слова)*

19 count\_pos dw 0

20 count\_neg dw 0

21 count\_zero dw 0

22

23 .code

24 start:

25 mov ax, @data

26 mov ds, ax

27

28 *; Ввод размера массива N*

29 mWriteStr mes\_n

30 mReadAX buf, 3 *; Чтение размера массива в регистр AX*

31 cmp ax, 10 *; Проверка, не превышает ли размер массива 10*

32 jbe Valid\_N

33 mov ax, 10 *; Ограничение размера массива значением 10, если введено больше*

34 Valid\_N:

35 mov n, ax *; Сохранение размера массива*

36

37 *; Ввод значений C и D*

38 mWriteStr new\_line

39 mWriteStr mes\_c

40 mReadAX buf, 7 *; Ввод значения C в регистр AX*

41 mov c, ax *; Сохранение C*

42

43 mWriteStr new\_line

44 mWriteStr mes\_d

45 mReadAX buf, 7 *; Ввод значения D в регистр AX*

46 mov d, ax *; Сохранение D*

47

48 *; Ввод массива*

49 xor si, si *; Обнуление индекса массива*

50 xor cx, cx

51 mov cx, n *; Установка счетчика цикла на количество элементов массива*

52 InputArray:

53 mWriteStr new\_line

54 mWriteStr mes\_num

55 mReadAX buf, 7 *; Ввод элемента массива*

56 mov arr[si], ax *; Сохранение элемента в массив*

57 add si, 2 *; Переход к следующему элементу массива*

58 loop InputArray *; Повторение цикла, пока CX != 0*

59

60 *; Обработка массива*

61 xor si, si *; Сброс индекса массива*

62 xor cx, cx

63 mov cx, n *; Установка счетчика на количество введенных элементов*

64

65 ProcessArray:

66 mov ax, arr[si] *; Загрузка текущего элемента массива*

67 cmp ax, c

68 jl NextElement *; Переход к следующему элементу, если меньше С*

69 cmp ax, d

70 jg NextElement *; Переход к следующему элементу, если больше D*

71

72 *; Условие выполнения: C <= arr[i] <= D*

73 cmp ax, 0

74 jg IncrementPositive *; > 0*

75 jl IncrementNegative *; < 0*

76 inc count\_zero *; = 0*

77 jmp NextElement

78

79 IncrementPositive:

80 inc count\_pos *; Увеличение счетчика положительных элементов*

81 jmp NextElement

82 IncrementNegative:

83 inc count\_neg *; Увеличение счетчика отрицательных элементов*

84 jmp NextElement

85

86 NextElement:

87 add si, 2 *; Переход к следующему элементу массива*

88 loop ProcessArray *; Повторение цикла обработки массива*

89

90 *; Вывод результатов*

91 mWriteStr new\_line

92 mWriteStr mes\_pos

93 mov ax, count\_pos

94 mWriteAX *; Вывод количества положительных элементов*

95

96 mWriteStr new\_line

97 mWriteStr mes\_neg

98 mov ax, count\_neg

99 mWriteAX *; Вывод количества отрицательных элементов*

100

101 mWriteStr new\_line

102 mWriteStr mes\_zero

103 mov ax, count\_zero

104 mWriteAX *; Вывод количества нулевых элементов*

105

106 *; Завершение программы*

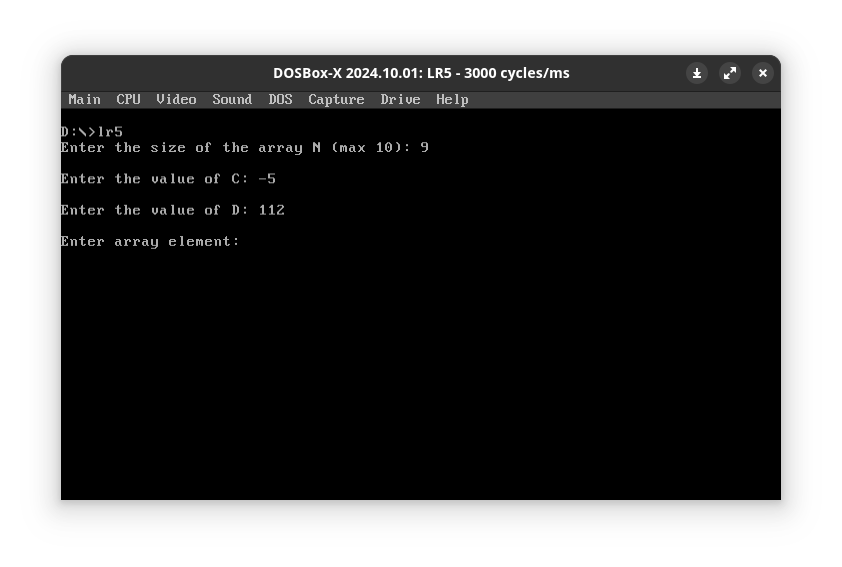
107 mov ax, 4C00h

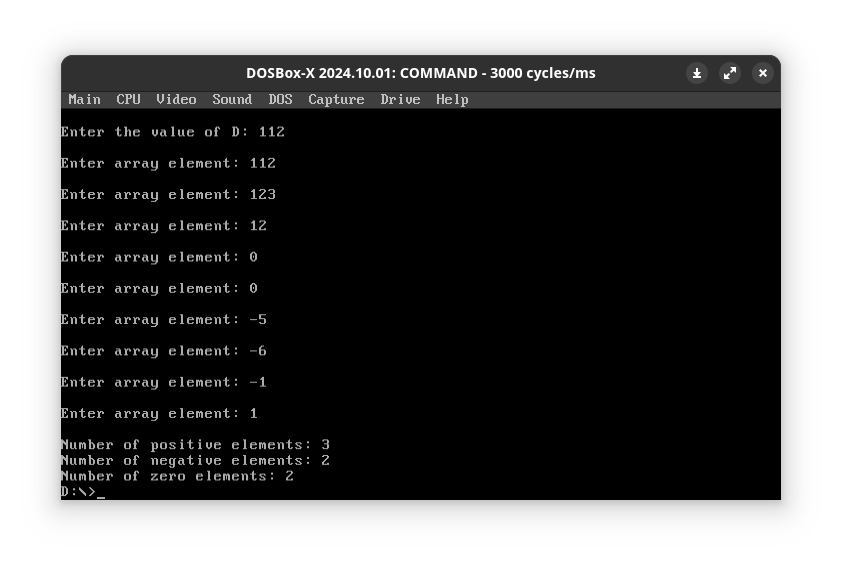
108 int 21h

109

110 end start

**Результат выполнения программы:**

****

****

**Вывод:** в ходе выполнения работы были сформированы практические навыки разработки программного кода на языке Ассемблера; изучены приёмы разработки макроопределений.