|  |  |
| --- | --- |
| Gerb-BMSTU_01 | ***«*Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_\_\_КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ОТЧЕТ**

**по лабораторной работе № 4**

# Дисциплина: Машинно-зависимые языки и основы компиляции

**Название лабораторной работы:** Программирование обработки массивов и матриц.

Студент гр. ИУ6-42Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. Д. Шатский**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ C.C.Данилюк**

(Подпись, дата) (И.О. Фамилия)

Москва, 2023

**Вариант 2.23**

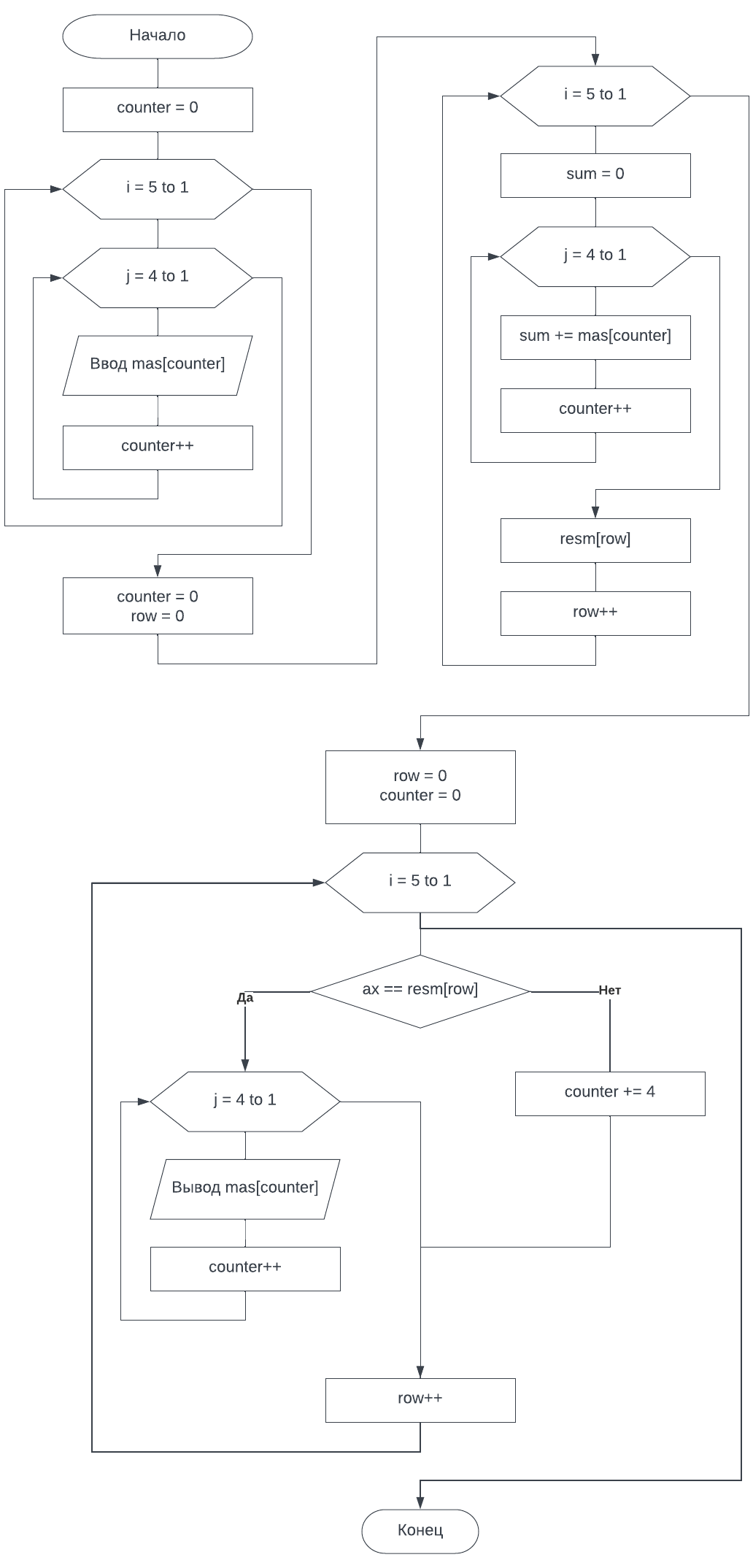
**Цель работы.** Изучение приемов моделирования обработки массивов и матриц в языке ассемблера.

**Задание.**

Дана матрица 5х4. Вычеркнуть строки с нулевой суммой элементов. Организовать ввод матрицы и вывод результатов.

**Схема.**

Схему алгоритма можно увидеть на рисунке 1:



*Рисунок 1 – Схема алгоритма*

**Программа.**

section .data

NewLine db 10

EntrMsg db "Enter number (row #):",10

lenEntr equ $-EntrMsg

ExitMsg db "Answer:",10

lenExit equ $-ExitMsg

section .bss

InBuf resb 10

lenIn equ $-InBuf

OutBuf resb 20

lenOut equ 10

MAS resw 20

RESM resw 5

section .text

global \_start

\_start:

mov EDX,0

mov EBX,0

mov ECX,5

cycle:

push RCX

push RDX

add DL,49

mov [EntrMsg+18],DL

mov ECX,4

cycle7:

mov rax, 1

mov rdi, 1

mov rsi, EntrMsg

mov rdx, lenEntr

push RCX

syscall

pop RCX

mov rax, 0

mov rdi, 0

mov rsi, InBuf

mov rdx, lenIn

push RCX

syscall

pop RCX

push RBX

mov esi,InBuf

call StrToInt64

cmp EBX, 0

jne StrToInt64.Error

pop RBX

mov [EBX\*2+MAS],ax

inc EBX

loop cycle7

pop RDX

pop RCX

INC RDX

loop cycle

mov EDX,0

mov EBX,0

mov ECX,5

cycle3:

push RCX

mov AX,0

mov ECX,4

cycle4:

add AX,[EBX\*2+MAS]

inc EBX

loop cycle4

mov [EDX\*2+RESM],AX

inc EDX

pop RCX

loop cycle3

mov rax, 1; системная функция 1 (write)

mov rdi, 1; дескриптор файла stdout=1

mov rsi, ExitMsg ; адрес выводимой строки

mov rdx, lenExit; длина строки

syscall

mov EDX,0

mov EBX,0

mov ECX,5

cycle2:

push RCX

push RDX

mov AX,0

cmp AX,[EDX\*2+RESM]

je deleted

mov ECX,4

cycle10:

mov esi, OutBuf ; загрузка адреса буфера вывода

mov ax, [EBX\*2+MAS] ; загрузка числа в регистр

cwde ; развертывание числа из ax в eax

call IntToStr64

mov rdx, rax; длина строки

mov rax, 1; системная функция 1 (write)

mov rdi, 1; дескриптор файла stdout=1

mov rsi, OutBuf ; адрес выводимой строки

push RCX

syscall

pop RCX

inc EBX

loop cycle10

mov rax, 1; системная функция 1 (write)

mov rdi, 1; дескриптор файла stdout=1

mov rsi, NewLine ; адрес выводимой строки

mov rdx, 1; длина строки

syscall

jmp skip

deleted:

add EBX,4

skip:

pop RDX

pop RCX

inc RDX

loop cycle2

; read

mov rax, 0; системная функция 0 (read)

mov rdi, 0; дескриптор файла stdin=0

mov rsi, InBuf; адрес вводимой строки

mov rdx, lenIn; длина строки

; вызов системной функции

syscall

; exit

mov rax, 60; системная функция 60 (exit)

xor rdi, rdi; return code 0

syscall

; вызов системной функции

%include "../lib64.asm"

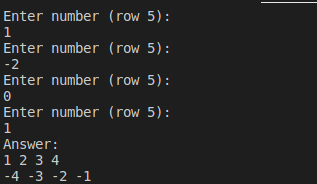
**Тесты.**

Результаты тестирования можно увидеть в таблице 1

Таблица 1 – результаты тестирования

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Полученный результат |
| 8 -8 8 -8  1 2 3 4  -4 -3 -2 -1  0 0 0 0  1 -2 0 1 | 1 2 3 4  -4 -3 -2 -1 | 1 2 3 4  -4 -3 -2 -1 |
| 1 2 3 4  1 1 1 1  2 2 2 2  3 3 3 3  -4 -4 -4 -4 | 1 2 3 4  1 1 1 1  2 2 2 2  3 3 3 3  -4 -4 -4 -4 | 1 2 3 4  1 1 1 1  2 2 2 2  3 3 3 3  -4 -4 -4 -4 |
| 1 2 3 4  0 0 0 0  1 1 -3 1  2 2 2 -6  8 8 -8 -8 | 1 2 3 4 | 1 2 3 4 |
| 0 0 0 0  0 0 0 0  0 0 0 0  0 0 0 0  0 0 0 0 | Все строки вычеркнуты | Все строки вычеркнуты |
| -7 -8 -9 -5  -8 -4 -7 -4  0 5 7 8  4 6 7 4  7 3 5 6 | -7 -8 -9 -5  -8 -4 -7 -4  0 5 7 8  4 6 7 4  7 3 5 6 | -7 -8 -9 -5  -8 -4 -7 -4  0 5 7 8  4 6 7 4  7 3 5 6 |

Результат запуска программы на выполнение можно увидеть на рисунке 2.



*Рисунок 2 – Результат выполнения программы*

**Контрольные вопросы.**

**1. Почему в ассемблере не определены понятия «массив», «матрица»?**Понятия «массив» и «матрица» в определены так как они имеют индентичное внутреннее представление, и полностью контролируются программистом.

**2. Как в ассемблере моделируются массивы?**В ассемблере массив моделируется в качестве последователности элементов в памяти. Для доступа к элементам используется смещение относительно начала массива.

**3. Поясните фрагмент последовательной адресации элементов массива?   
Почему при этом для хранения частей адреса используют регистры?**Элементы массива расположены в памяти друг за другом, Начало массива выражается непосредственным смещением. Для получения последующих элементов массива необходимо добавить к непосредственному смещению массива номер элемента-1 умноженный на длину этого элемента. Поэтому при таком способе адресации необходимо начало массива и номер элемента хранить в массивах.

**4. Как в памяти компьютера размещаются элементы матриц?**Элементы матриц размещаются в памяти либо по строкам, либо по столбцам. Для хранения их матрицу разворачивают в одномерный массив.

**5. Чем моделирование матриц отличается от моделирования массивов? В   
каких случаях при выполнении операций для адресации матриц используется один регистр, а в каких – два?**

Так как матрицы хранятся в памяти как массивы, отличие массивов от матриц заключается только в их обработке. Для адресации используется 1 регистр, если матрица расположена в памяти построчно и простмотр идёт по строкам. В других случаях нужно использовать два регистра, для хранения адреса текущей строки и смещения от начала этой строки.

**Вывод.**

Былиизучены приемы моделирования обработки массивов и матриц в языке ассемблера.