|  |  |
| --- | --- |
| 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: Обработки массивов и матриц**

Вариант 2.21

Студент гр. ИУ6-42Б  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.С. Твердюк**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.С. Данилюк**

(Подпись, дата) (И.О. Фамилия)

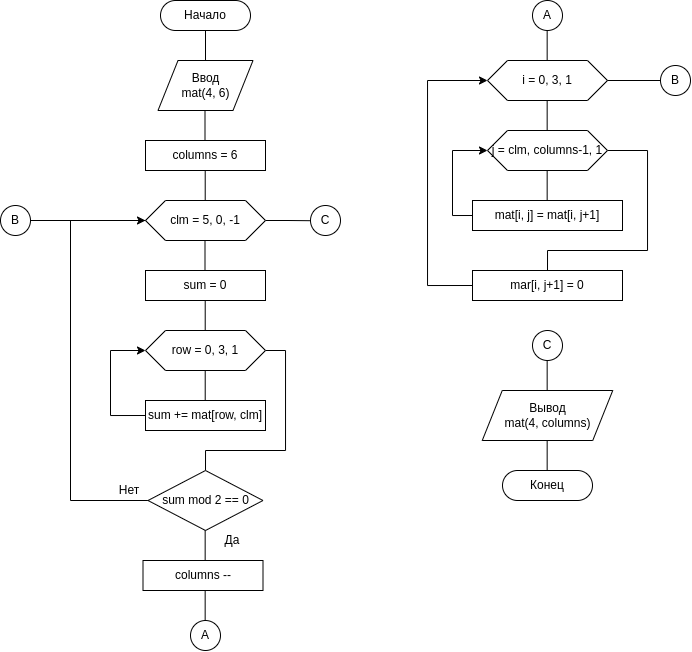
Москва, 2023

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

***Ход работы:***

***Задание.*** Дана матрица 4х6. Вычеркнуть столбцы с четной суммой элементов. Организовать ввод матрицы и вывод результатов.

Разработанная схема алгоритма приведена на рисунке 1.

Рисунок 1 - Схема алгоритма

**Программа LR4.asm**

%define ROWS 4

%define COLUMNS 6

section .data

prsNum db "Enter matrix:", 10

lenPrsNum equ $-prsNum

space db " "

endLine db 10

ExitMsg db "Result:", 10

lenExit equ $-ExitMsg

section .bss

InBuf resb 30

lenIn equ $-InBuf

OutBuf resb 10

mat resq 24

section .text

global \_start

\_start:

mov rax, 1

mov rdi, 1

mov rsi, prsNum

mov rdx, lenPrsNum

syscall

mov rcx, ROWS

xor rdi, rdi

read\_line:

push rcx

push rdi

mov rax, 0

mov rdi, 0

mov rsi, InBuf

mov rdx, lenIn

syscall

pop rdi

mov rcx, rax

xor rdx, rdx

xor r8, r8

process\_line:

cmp byte[InBuf + rdx], 10

je process\_number

cmp byte[InBuf + rdx], ' '

jne next\_sym

mov byte[InBuf + rdx], 10

jmp process\_number

process\_number:

push rdx

call StrToInt64

cmp ebx, 0

jne StrToInt64.Error

mov [mat + 8 \* rdi], rax

inc rdi

pop rdx

mov r8, rdx

inc r8

lea rsi, [InBuf + r8]

next\_sym:

inc rdx

loop process\_line

pop rcx

dec rcx

cmp rcx, 0

jne read\_line

xor rcx, rcx

mov rcx, COLUMNS

xor r9, r9

mov r9, COLUMNS

dec r9

even\_columns:

dec rcx

cmp rcx, -1

je write\_mat

push rcx

mov rdx, rcx

xor rax, rax

mov rcx, ROWS

summing\_column:

dec rcx

mov r10, 8

mov r11, r9

inc r11

imul r10, r11

imul r10, rcx

add rax, qword[mat + 8 \* rdx + r10]

cmp rcx, 0

jne summing\_column

pop rcx

mov bx, 2

cwd

idiv bx

cmp rdx, 0

jne even\_columns

push rcx

cmp rcx, r9

je data\_shift

mov rbx, rcx

columns\_shift:

mov rcx, ROWS

c\_shift:

dec rcx

mov r10, 8

mov r11, r9

inc r11

imul r10, r11

imul r10, rcx

sub r11, 8

mov rdx, qword[mat + r10 + 8 \* rbx + 8]

mov [mat + r10 + 8 \* rbx], rdx

cmp rcx, 0

jne c\_shift

inc rbx

cmp rbx, r9

jne columns\_shift

data\_shift:

mov rcx, ROWS

n\_shift:

dec rcx

mov rax, ROWS

sub rax, rcx

mov rdx, r9

inc rdx

imul rax, rdx

mov r10, 8

mov r11, r9

inc r11

imul r10, r11

imul r10, rcx

sub r10, 8

mov rdx, 0

mov qword[mat + r10 + 8 \* rax], rdx

xor rbx, rbx

n\_swap:

mov rdx, qword[mat + r10 + 8 \* rbx + 8]

mov [mat + r10 + 8 \* rbx], rdx

inc rbx

cmp rbx, rax

jne n\_swap

cmp rcx, 1

jne n\_shift

pop rcx

dec r9

jmp even\_columns

write\_mat:

mov rax, 1

mov rdi, 1

mov rsi, ExitMsg

mov rdx, lenExit

syscall

cmp r9, -1

je exit

mov rbx, -1

write\_line:

cmp rbx, 3

je exit

inc rbx

mov rcx, -1

write\_num:

inc rcx

push rcx

mov r10, 8

mov r11, r9

inc r11

imul r10, r11

imul r10, rbx

mov rsi, OutBuf

mov rax, [mat + r10 + 8 \* rcx]

cwde

call IntToStr64

mov rdx, rax

mov rax, 1

mov rdi, 1

mov rsi, OutBuf

syscall

pop rcx

cmp rcx, r9

jne print\_space

jmp end\_line

exit:

mov rax, 60

xor rdi, rdi

syscall

print\_space:

push rcx

mov rax, 1

mov rdi, 1

mov rsi, space

mov rdx, 1

syscall

pop rcx

jmp write\_num

end\_line:

mov rax, 1

mov rdi, 1

mov rsi, endLine

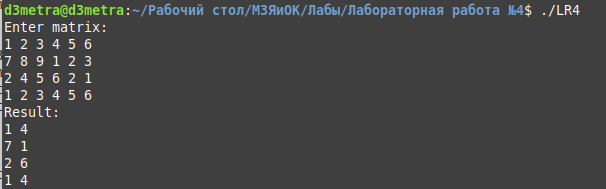
mov rdx, 1

syscall

jmp write\_line

%include "../lib642.asm"

Запустим программу. Результат выполнения программы представлен на рисунке 2.

Рисунок 2 - Выполнение программы

Проведём тестирование программы. Результаты представлены в таблице 1.

*Таблица 1 — Результаты тестирования программы*

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| 1 2 3 4 5 6  1 2 3 4 5 6  1 2 3 4 5 6  1 2 3 4 5 6 | — | — |
| -1 12 4 3 5 1  0 9 17 -13 4 8  3 -14 21 2 8 9  10 5 7 -7 -8 -9 | 4 3 5 1  17 -13 4 8  21 2 8 9  7 -7 -8 -9 | 4 3 5 1  17 -13 4 8  21 2 8 9  7 -7 -8 -9 |
| 1 2 3 4 5 6  6 5 4 3 2 1  1 2 1 2 1 2  0 0 0 0 0 0 | 2 4 6  5 3 1  2 2 2  0 0 0 | 2 4 6  5 3 1  2 2 2  0 0 0 |

***Контрольные вопросы***

1. Почему в ассемблере не определены понятия «массив», «матрица»? Понятия «массив» и «матрица» не определены, так как они имеют идентичное внутреннее представление – последовательность элементов определенного размера в памяти.

2. Как в ассемблере моделируются массивы?

В ассемблере массив моделируется в качестве последовательности 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?

mov rdx, rcx ; В rdx храним смещение для столбца

xor rax, rax

mov rcx, ROWS

summing\_column:

dec rcx

mov r10, 8 ; Вычисляем смещение для строк

mov r11, r9

inc r11

imul r10, r11

imul r10, rcx

add rax, qword[mat + 8 \* rdx + r10] ; Сложение элементов одного стобца

cmp rcx, 0

jne summing\_column

4. Как в памяти компьютера размещаются элементы матриц?

Элементы матриц размещаются в памяти по строкам или по столбцам.

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

Матрицы хранятся в памяти как массивы, отличие массивов от матриц заключается только в их обработке. Если матрица расположена построчно и просмотр выполняется по строкам, то используется один регистр. В других случаях нужно использовать два регистра - для хранения адреса текущей строки и смещения от начала этой строки.

***Вывод:*** изучены приемов моделирования обработки массивов и матриц в языке ассемблера. Разработанная программа на заданных исходных данных работает корректно.