ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

"ВЫСШАЯ ШКОЛА ЭКОНОМИКИ"

Факультет компьютерных наук

Департамент программной инженерии

Микропроект №1

Пояснительная записка

Исполнитель

Студент группы БПИ194

_____/ В.Д. Хворостяной /

"___"_____2020 г.

Москва 2020

СОДЕРЖАНИЕ

1. УСЛОВИЕ ЗАДАНИЯ

1.1. Текст задания

Разработать программу умножения матриц порядка N=4 при условии размещения элементов матриц в линейном массиве по строкам.

2. МЕТОДЫ И АЛГОРИТМЫ ПРОГРАММЫ

2.1. Методы работы с данными

Для ввода и обработки данных в программе используются циклы на основе условных и безусловных переходов, заменяющие собой циклы while – jmp+jle/jl/jge, а также условную инструкцию – je/jl/jge. Также в программе используются section data с данными использующими в программе.

Основные структуры - массивы, строки, числа.

2.2. Алгоритм нахождения произведения матриц

Для нахождения произведения двух матриц применяется прямое определение: A*B=C, $C_{ij} = \sum^{n}_{k=1} Aik * B_{kj}$. Для вычислений значений используются циклы.

3. ОПИСАНИЕ ДАННЫХ В ПРОГРАММЕ

3.1. Входные данные

Фактически входные данные ограничены 10 байтами для каждого введенного целого числа. В противном случае происходит переполнение. Значения будут некорректными.

В программе описаны ограничения для 10000 по модулю. В таких пределах данные всегда будут корректные. Фактическое ограничение несколько больше, но смысловой нагрузки незначительное повышение ограничений не имеет, поскольку нет необходимости вычислять матрицы с большими числовыми значениями.

При вводе нечисловых символов программа завершит работу заполним все оставшиеся входные значения 0.

3.2. Выходные и промежуточные данные

С учетом ограничений по вводу чисел в 10000 промежуточные значения как и выходные не превышают 4 * 10^8 по модулю. При вводе больших чисел, чем описано в программе приведет к увеличению промежуточных и выходных данных, а также может привести их к переполнению, и как следствие некорректным данным.

4. ТЕСТИРОВАНИЕ ПРОГРАММЫ

4.1. Положительные тесты

```
This is program to multiplicate two matrix: A * B. Both matrix 4*4
Program will work only if user input correct data. In other situation program will finish
Program will output correct result when absolute integer numbers not upper than 10^4(10000)
A_1_1 = 1
A_1_2 = 2
A_1_3 = 3
A_1_4 = 4
A_2_1 = 2
A_2_2 = 3
A_2_3 = 4
A_2_4 = 5
A_3_1 = 3
A_3_2 = 4
A_3_3 = 5
A_3_4 = 6
A_4_1 = 0
A_4_2 = 1
A_4_3 = 2
A_4_4 = 3
Matrix A:
1 2 3 4
2 3 4 5
3 4 5 6
0 1 2 3
B_1_1 = 1
B_1_2 = 2
B_1_3 = 3
B_1_4 = 4
B_2_1 = 4
B_2_2 = 3
B_2_3 = 2
B_2_4 = 1
B_3_1 = 0
B_3_2 = 1
B_3_3 = 2
B_3_4 = 3
B_4_1 = 3
B_4_2 = 2
B_4_3 = 1
B_4_4 = 0
Matrix B:
1 2 3 4
4 3 2 1
0 1 2 3
3 2 1 0
Result Matrix(A*B):
21 19 17 15
29 27 25 23
37 35 33 31
13 11 9 7
```

Проверка

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{B} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 0 & 1 & 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \\ 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 21 & 19 & 17 & 15 \\ 29 & 27 & 25 & 23 \\ 37 & 35 & 33 & 31 \\ 13 & 11 & 9 & 7 \end{pmatrix}$$

```
This is program to multiplicate two matrix: A * B. Both matrix 4*4
Program will work only if user input correct data. In other situation program will finish
Program will output correct result when absolute integer numbers not upper than 10^4(10000)
A_1_1 = 10
A_1_2 = 30
A_1_3 = 23
A_1_4 = 29
A_2_1 = 17
A_2_2 = 29
A_2_3 = 50
A_2_4 = 34
A_3_1 = 59
A_3_2 = 59
A_3_3 = 34
A_3_4 = 57
A_4_1 = 23
A_4_2 = 54
A_4_3 = 21
A_4_4 = 34
Matrix A:
10 30 23 29
17 29 50 34
59 59 34 57
23 54 21 34
B_1_1 = 89
B_1_2 = 34
B_1_3 = 55
B_1_4 = 34
B_2_1 = 87
B_2_2 = 67
B_2_3 = 54
B_2_4 = 32
B_3_1 = 91
B_3_2 = 81
B_3_3 = 84
B_3_4 = 73
B_4_1 = 69
B_4_2 = 50
B_4_3 = 34
B_4_4 = 18
Matrix B:
89 34 55 34
87 67 54 32
91 81 84 73
69 50 34 18
Result Matrix(A*B):
7594 5663 5088 3501
10932 8271 7857 5768
17411 11563 11225 7402
11002 7801 7101 4655
```

Проверка

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{B} = \begin{pmatrix} 10 & 30 & 23 & 29 \\ 17 & 29 & 50 & 34 \\ 59 & 59 & 34 & 57 \\ 23 & 54 & 21 & 34 \end{pmatrix} \cdot \begin{pmatrix} 89 & 34 & 55 & 34 \\ 87 & 67 & 54 & 32 \\ 91 & 81 & 84 & 73 \\ 69 & 50 & 34 & 18 \end{pmatrix} = \begin{pmatrix} 7594 & 5663 & 5088 & 3501 \\ 10932 & 8271 & 7857 & 5768 \\ 17411 & 11563 & 11225 & 7402 \\ 11002 & 7801 & 7101 & 4655 \end{pmatrix}$$

4.2. Отрицательные тесты

```
This is program to multiplicate two matrix: A * B. Both matrix 4*4
Program will work only if user input correct data. In other situation program will finish
Program will output correct result when absolute integer numbers not upper than 10^4(10000)
A_1_1 = -523
A_1_2 = 335
A_1_3 = 342
A_1_4 = 353
A_2_1 = -352
A_2_2 = 35
A_2_3 = -553
A_2_4 = -535
A_3_1 = -134
A_3_2 = 343
A_3_3 = 553
A_3_4 = 556
A_4_1 = 876
A_4_2 = -656
A_4_3 = -685
A_4_4 = -976
Matrix A:
-523 335 342 353
-352 35 -553 -535
-134 343 553 556
876 -656 -685 -976
B_1_1 = -913
B_1_2 = -353
B_1_3 = -363
B_1_4 = -644
B_2_1 = -405
B_2_2 = 353
B_2_3 = 565
B_2_4 = 434
B_3_1 = 696
B_3_2 = 466
B_3_3 = 454
B_3_4 = 648
B_4_1 = -546
B_4_2 = -456
B_4_3 = -283
B_4_4 = 532
Matrix B:
-913 -353 -363 -644
-405 353 565 434
696 466 454 648
-546 -456 -283 532
Result Matrix(A*B):
387118 301278 434493 891614
214423 122873 47894 -401086
64739 172543 336151 889294
-477972 -414950 -723410 -1811960
```

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{B} = \begin{pmatrix} -523 & 335 & 342 & 353 \\ -352 & 35 & -553 & -535 \\ -134 & 343 & 553 & 556 \\ 876 & -656 & -685 & -976 \end{pmatrix} \cdot \begin{pmatrix} -913 & -353 & -363 & -644 \\ -405 & 353 & 565 & 434 \\ 696 & 466 & 454 & 648 \\ -546 & -456 & -283 & 532 \end{pmatrix} =$$

$$= \begin{pmatrix} 387118 & 301278 & 434493 & 891614 \\ 214423 & 122873 & 47894 & -401086 \\ 64739 & 172543 & 336151 & 889294 \\ -477972 & -414950 & -723410 & -1811960 \end{pmatrix}$$

4.3. Тесты с переполнением

```
This is program to multiplicate two matrix: A * B. Both matrix 4*4
Program will work only if user input correct data. In other situation program will finish
Program will output correct result when absolute integer numbers not upper than 10^4(10000)
A_1_1 = 10000
A_1_2 = 30000
A_1_3 = 10000
A_1_4 = 40000
A_2_1 = 20000
A_2_2 = 30000
A_2_3 = 40000
A_2_4 = 20000
A_3_1 = 10000
A_3_2 = 10000
A_3_3 = 30000
A_3_4 = 20000
A_4_1 = 40000
A_4_2 = 20000
A_4_3 = 30000
A_4_4 = 10000
Matrix A:
10000 30000 10000 40000
20000 30000 40000 20000
10000 10000 30000 20000
40000 20000 30000 10000
B_1_1 = 10000
B_1_2 = 40000
B_1_3 = 30000
B_1_4 = 20000
B_2_1 = 30000
B_2_2 = 20000
B_2_3 = 10000
B_2_4 = 20000
B_3_1 = 40000
B_3_2 = 40000
B_3_3 = 30000
B_3_4 = 20000
B_4_1 = 40000
B_4_2 = 30000
B_4_3 = 30000
B_4_4 = 10000
Matrix B:
10000 40000 30000 20000
30000 20000 10000 20000
40000 40000 30000 20000
40000 30000 30000 10000
Result Matrix(A*B):
-1294967296 -1694967296 2100000000 1400000000
-794967296 -694967296 -1594967296 2000000000
-1894967296 -1894967296 1900000000 1200000000
-1694967296 -794967296 -1694967296 1900000000
```

4.4. Тесты с прерыванием программы

```
This is program to multiplicate two matrix: A * B. Both matrix 4*4
Program will work only if user input correct data. In other situation program will finish
Program will output correct result when absolute integer numbers not upper than 10^4(10000)
A_1_1 = 1
A_1_2 = 1.0
A_1_3 = A_1_4 = A_2_1 = A_2_2 = A_2_3 = A_2_4 = A_3_1 = A_3_2 = A_3_3 = A_3_4 = A_4_1 = A_4_2 = A_4_3 = A_4_4 = Matrix A:
1 1 0 0
0 0 0 0
0 0 0 0
0 0 0 0
B_1_1 = B_1_2 = B_1_3 = B_1_4 = B_2_1 = B_2_2 = B_2_3 = B_2_4 = B_3_1 = B_3_2 = B_3_3 = B_3_4 = B_4_1 = B_4_2 = B_4_3 = B_4_4 = Matrix B:
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
Result Matrix(A*B):
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
```

```
This is program to multiplicate two matrix: A * B. Both matrix 4*4
Program will work only if user input correct data. In other situation program will finish
Program will output correct result when absolute integer numbers not upper than 10^4(10000)
A_1_1 = 1
A_1_2 = 1
A_1_3 = 1
A_1_4 = 1
A_2_1 = 1
A_2_2 = 1
A_2_3 = 1
A_2_4 = A_3_1 = A_3_2 = A_3_3 = A_3_4 = A_4_1 = A_4_2 = A_4_3 = A_4_4 = Matrix A:
1 1 1 1
1 1 0 0
0 0 0 0
0 0 0 0
B_1_1 = B_1_2 = B_1_3 = B_1_4 = B_2_1 = B_2_2 = B_2_3 = B_2_4 = B_3_1 = B_3_2 = B_3_3 = B_3_4 = B_4_1 = B_4_2 = B_4_3 = B_4_4 = Matrix B:
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
Result Matrix(A*B):
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
```

ПРИЛОЖЕНИЕ 1

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1) ГОСТ 19.404-79 Пояснительная записка. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

2) http://softcraft.ru/edu/comparch/practice/asm86/03-subprog/sum1-32/sum.asm - Пример программы на языке ассемблера.

3) http://osinavi.ru/index.php?param1=4&param2=18&hidden=0&supp=1 - Теоритический материал по условным и безусловным переходам.

ТЕКСТ ПРОГРАММЫ

```
format PE console
entry start

include 'C:\Fasm\INCLUDE\win32a.inc'

section '.data' data readable writable

        strVecElemA       db 'A_%d_%d = ', 0                    ;для информации о вводе элемента в матрице A
        strVecElemB       db 'B_%d_%d = ', 0                    ;для информации о вводе элемента в матрице B
        strScanInt        db '%d', 0                            ;\
        strVecElemOutEnter db '%d ', 0                          ;/для ввода чисел
        strStartInfo      db 'This is program to multiplicate two matrix: A * B', 10, 0      ;информация о программа
        strMatrixSize     db "Both matrix 4*4", 10, 0                                        ;информация о размере матриц
        strInstract       db 'Program will work only if user input correct data. In other situation program will finish', 10, 0
;информация касаемо ввода чисел в программе
        strNumbers        db 'Program will output correct result when absolute numbers not upper than 10^4(10000) ', 10, 0
;информация для пользовател об ограничениях
        strMatrixA        db 'Matrix A:', 10, 0                                              ;информация о матрице A
        strMatrixB        db 'Matrix B:', 10, 0                                              ;информация о матрице B
        strMatrixRes      db 'Result Matrix(A*B):', 10, 0                                    ;информация о результирующе матрице
        newLine           db '', 10, 0                                                       ;перенос строки

        vec_size     dd 16          ;постоянное значение длины массивов
        sum          dd 0           ;переменная для суммы
        i            dd ?           ;индекс первого уровня
        tmp          dd ?           ;дополнительная переменная для хранения
        vec          rd 16          ;массив A
        vecB         rd 16          ;массив B
        vecRes       rd 16          ;итоговый массив
        j            dd ?           ;индекс 2ого уровня
        cicleUp      dd ?           ;счетчик внешнего цикла
        cicleDown    dd ?           ;счетчик внутреннего цикла

section '.code' code readable executable

start:
        push strStartInfo           ;\
        call [printf]               ; \
        push strMatrixSize          ;  \
        call [printf]               ;   \
                                    ;    basic information to user
        push strInstract            ;   /
        call [printf]               ;  /
        push strNumbers             ; /
        call [printf]               ;/

        mov eax, [vec_size]
        cmp eax, 0
        jg  getVectorA              ;start work with matrix

;input matrix A
getVectorA:
        mov eax, 1
        mov ebx, vec                ; ebx = &vec

getVecLoopA:
        mov ecx, 1

getOneElemA:
        mov [tmp], ebx
        mov [i], ecx
        push ecx
```

```
        mov [j], eax
        push eax
        push strVecElemA
        call [printf]               ;print elem info

        push ebx
        push strScanInt
        call [scanf]                ;scan to get elem

        mov ecx, [i]
        mov eax, [j]
        inc ecx
        mov ebx, [tmp]
        add ebx, 4
        cmp ecx, 5
        jl getOneElemA              ;update cicle data

        inc eax
        cmp eax, 5
        jl getVecLoopA
        cmp eax, 5
        jge endInputMatrixA         ;continue or finish input

endInputMatrixA:
        push strMatrixA
        call [printf]               ;matrix A info

        mov eax, 0
        mov ebx, vec

;print matrix A
printLineA:
        mov ecx, 0

printElemA:
        mov [tmp], ebx
        cmp eax, 16
        jge getVectorB              ;finish print A
        mov [i], ecx
        mov [j], eax

        push dword [ebx]
        push strVecElemOutEnter
        call [printf]               ;print elem

        mov ecx, [i]
        mov eax, [j]
        inc ecx
        inc eax
        mov ebx, [tmp]
        add ebx, 4
        cmp ecx, 4
        jl printElemA               ;cicle to print elem in line

        mov [i], ecx
        mov [j], eax

        push newLine
        call [printf]               ;\n

        mov ecx, [i]
        mov eax, [j]
        cmp ecx, 4
        je printLineA               ;cicle to print - new line

;input Matrix B
getVectorB:
        mov eax, 1
        mov ebx, vecB               ; ebx = &vec
```

```
getVecLoopB:
	mov ecx, 1

getOneElemB:
	mov [tmp], ebx
	mov [i], ecx
	push ecx
	mov [j], eax
	push eax
	push strVecElemB
	call [printf]		;print info about new elem

	push ebx
	push strScanInt		;scan to get elem
	call [scanf]

	mov ecx, [i]
	mov eax, [j]
	inc ecx
	mov ebx, [tmp]
	add ebx, 4
	cmp ecx, 5
	jl getOneElemB		;cicle to get elems

	inc eax
	cmp eax, 5
	jl getVecLoopB
	cmp eax, 5
	jge endInputMatrixB	;continue cicle or finish

endInputMatrixB:
	push strMatrixB		;base data to printB
	call [printf]

	mov eax, 0
	mov ebx, vecB

;printB
printLineB:
	mov ecx, 0

printElemB:
	mov [tmp], ebx
	cmp eax, 16
	jge MulMatrix		;end print
	mov [i], ecx
	mov [j], eax

	push dword [ebx]
	push strVecElemOutEnter	;print elem
	call [printf]

	mov ecx, [i]
	mov eax, [j]
	inc ecx
	inc eax
	mov ebx, [tmp]
	add ebx, 4
	cmp ecx, 4
	jl printElemB		;cicle to print elem in line

	mov [i], ecx
	mov [j], eax

	push newLine
	call [printf]		;\n

	mov ecx, [i]
```

```
        mov eax, [j]
        cmp ecx, 4
        je printLineB               ;cicle to print - new line


MulMatrix:                          ;base data od cicles
        mov edx, 0
        mov esi, 0
        cmp esi, 64
        jge endMul

MulLine:                            ;base data of cicleUp
        mov [cicleUp], 0
        mov edi, 0
        cmp esi, 64
        jge endMul

MulElem:                            ;base data of cicleDown
        cmp esi, 64
        jge endMul
        mov [cicleDown], 0
        mov [sum], 0

        cmp [cicleUp], 4
        jge endLineMul

Multi:
        cmp [cicleDown], 4
        jge endElemMul
        mov ecx, [vec+edx]
        imul ecx, [vecB+edi]
        add [sum], ecx              ;sum elements

        add edi, 16
        add edx, 4
        inc [cicleDown]
        jmp Multi                   ;update cicleUp parametrs

endElemMul:
        mov ecx, [vecRes+esi]
        mov ecx, [sum]
        mov [vecRes+esi], ecx       ;update res matrix

        add esi, 4
        sub edx, 16
        sub edi, 60
        inc [cicleUp]
        add ebx, 4
        jmp MulElem                 ;update cicleUp parametrs

endLineMul:
        add edx, 16                 ;end line mul
        jmp MulLine

endMul:                             ;start to print result
        push strMatrixRes
        call [printf]

        mov eax, 0
        mov ebx, vecRes


printLineRes:                       ;point to refresh ecx
        mov ecx, 0

printElemRes:                       ;print result matrix
        mov [tmp], ebx
        cmp eax, 16
        jge endOutputRes            ;to out from program
```

```
        mov [i], ecx
        mov [j], eax

        push dword [ebx]
        push strVecElemOutEnter          ;print element
        call [printf]

        mov ecx, [i]
        mov eax, [j]
        inc ecx
        inc eax
        mov ebx, [tmp]
        add ebx, 4
        cmp ecx, 4
        jl printElemRes                  ;return if cicle is end

        mov [i], ecx
        mov [j], eax

        push newLine                     ;\n
        call [printf]

        mov ecx, [i]
        mov eax, [j]
        cmp ecx, 4
        je printLineRes                  ;return if cicle is end

endOutputRes:

finish:
                call [getch]

                push 0
                call [ExitProcess]

;-------------------------------third act - including HeapApi--------------------------

section '.idata' import data readable
    library kernel, 'kernel32.dll',\
            msvcrt, 'msvcrt.dll',\
            user32,'USER32.DLL'

include 'C:\Fasm\INCLUDE\api\user32.inc'
include 'C:\Fasm\INCLUDE\api\kernel32.inc'
    import kernel,\
           ExitProcess, 'ExitProcess',\
           HeapCreate,'HeapCreate',\
           HeapAlloc,'HeapAlloc'
  include 'C:\Fasm\INCLUDE\api\kernel32.inc'
    import msvcrt,\
           printf, 'printf',\
           scanf, 'scanf',\
           getch, '_getch'
```