*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«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 (ИУ6)\_\_\_\_\_\_\_\_\_\_\_

**Отчет**

**по лабораторной работе № 4**

**Вариант 22**

**Дисциплина: машинно-зависимые языки и основы компиляции**

**Название лабораторной работы:** **Программирование ветвлений и циклов**

Студент гр. ИУ6-44  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шумаков А.А.**

(Подпись, дата) (И.О. Фамилия)

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

(Подпись, дата) (И.О. Фамилия)

Москва, 2018

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

**Задание**

Дана матрица 5х5. Определить сумму положительных элементов над побочной диагональю. Организовать ввод матрицы и вывод результатов.

**Выполнение**

В первую очередь приведем на рисунке 1 схему алгоритма.

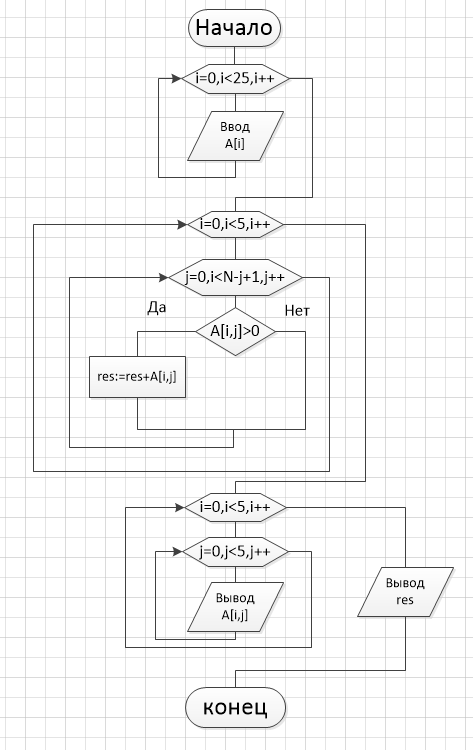


Рисунок 1: Схема алгоритма

Приведем код данной программы с комментариями:

*.586*

*.MODEL flat, stdcall*

*OPTION CASEMAP:NONE*

*Include kernel32.inc*

*Include masm32.inc*

*IncludeLib kernel32.lib*

*IncludeLib masm32.lib*

*.CONST*

*MsgExit DB 13,10,"Press Enter to Exit",0AH,0DH,0*

*.DATA*

*RequestA DB "Input matrix 5x5=25: ",0AH,0DH,0*

*; NxN*

*; i<=N-j+i*

*A sword 0,0,0,0,0*

*sword 0,0,0,0,0*

*sword 0,0,0,0,0*

*sword 0,0,0,0,0*

*sword 0,0,0,0,0*

*N sword 5 ; Количество строк и полей*

*i sword 1 ; для пробега по строкам*

*j sword 1 ; для пробега по столбцам*

*Per DB 13,10,0 ; Для перехода на следующую строку при выводе матрицы*

*Result DB 'summa = ' ; Заготовка для вывода результата*

*Pusto DB ' ' ; Пробел для разделения столбцов при выводе матрицы*

*ResStr DB 16 DUP (' '),0 ; Помещаем значение переменной для вывода результата*

*ResStrOut DB 5 DUP (' '),0 ; Помещаем значение переменной для вывода матрицы*

*OutMatrix DB 'Output the matrix',13,10,0*

*.DATA?*

*inbuf DB 100 DUP (?)*

*res sword ? ;Переменная для хранения результата*

*InOutBuf DB 100 DUP (?) ; Буфер ввода*

*current sword ? ; Текущий элемент матрицы*

*.CODE*

*Start:*

*;Ввод матрицы*

*Invoke StdOut, ADDR RequestA*

*mov EBX,0*

*mov ECX,25*

*CycleInput:*

*push ECX ; значение ECX в стек*

*Invoke StdIn,ADDR InOutBuf,LengthOf InOutBuf ;Ввод элемента матрицы*

*Invoke StripLF, ADDR InOutBuf*

*Invoke atol, ADDR InOutBuf*

*mov A[EBX],AX ; Помещаем элемент в матрицу*

*pop ECX ; достаем ECX из стека*

*add EBX,2 ; Переходим к новому элементу матрицы*

*loop CycleInput*

*; Сумма положительных элементов над побочной диагональю*

*mov EBX,0 ;в EBX лежит смещение относительно базы*

*mov ECX,5 ;ECX количество столбцов*

*cycle1:*

*push ECX ; в стек номер строки*

*mov ECX,5 ; в ECX количество столбцов*

*cycle2:*

*XOR EDX,EDX ;Очищаем EDX*

*mov DX,N ;DX:=N*

*mov current,DX ;current:=N*

*mov DX,j ;DX:=j*

*sub current,DX ;current:=N-j*

*inc current ;current:=N-j+1*

*;Выражение N-j+1 обеспечивает "проход"*

*;над побочной диагональю*

*mov DX,current ;DX:=N-j+1*

*cmp i,DX ;i<=N-j+1*

*jle metka1 ;Условие выполнено*

*jmp continue ;Условие ложно*

*metka1: cmp A[EBX\*2],0 ;Определяем положительный элемент*

*jg metka2 ;Элемент матрицы положителен*

*jmp continue ;Элемент матрицы отрицателен*

*metka2: mov AX,A[EBX\*2]*

*add res,AX ;Считаем сумму положительных элементов*

*continue:*

*inc EBX ; Переход к новому элементу*

*inc j ; следующий столбец*

*loop cycle2*

*pop ECX ; достаем из стека номер строки*

*sub j,5 ;j:=j-5=N+1-5 В j снова 1*

*inc i ; Следующая строка*

*loop cycle1*

*XOR EAX,EAX*

*;Вывод матрицы*

*Invoke StdOut, ADDR OutMatrix*

*mov EBX,0 ;в EBX лежит смещение относительно базы*

*mov ECX,5 ;ECX количество столбцов*

*cycleOutput1:*

*push ECX*

*mov ECX,5*

*cycleOutput2:*

*push ECX ;Для корректного вывода помещаем номер столбца в стек*

*Invoke dwtoa,A[EBX\*2], ADDR ResStrOut ; Элемент матрицы помещаем в ResStrOut*

*Invoke StdOut, ADDR ResStrOut ; Выводим элемент матрицы*

*Invoke StdOut, ADDR Pusto ; Ставим пробел при выводе элементов*

*pop ECX*

*inc EBX*

*loop cycleOutput2*

*pop ECX*

*push ECX ; Для корректного вывода матрицы помещаем номер строки в стек*

*invoke StdOut, ADDR Per ; Переход на новую строку при выводе*

*pop ECX*

*loop cycleOutput1*

*XOR EAX,EAX*

*invoke StdOut, ADDR Per*

*Invoke dwtoa,res, ADDR ResStr*

*Invoke StdOut,ADDR Result*

*Invoke StdOut,ADDR MsgExit*

*Invoke StdIn,ADDR inbuf,LengthOf inbuf*

*Invoke ExitProcess,0*

*End Start*

На рисунке 2 приведены результаты выполнения программы

**Тесты программы**

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Полученный результат |
| 1,2,3,4,5  6,7,8,9,10  11,12,13,14,15  16,17,18,19,20  21,22,23,24,25 | 135 | 135 |
| 9,8,7,6,5  4,3,2,1,-1  -2,-3,-4,-5,-6  -7,-8,-9,8,5  2,7,4,1,9 | 47 | 47 |
| -7,-5,-3,9,-5  1,-4,-7,5,6  -8,1,2,-4,5  8,-7,-1,0,3  -4,-8,-7,4,5 | 26 | 26 |

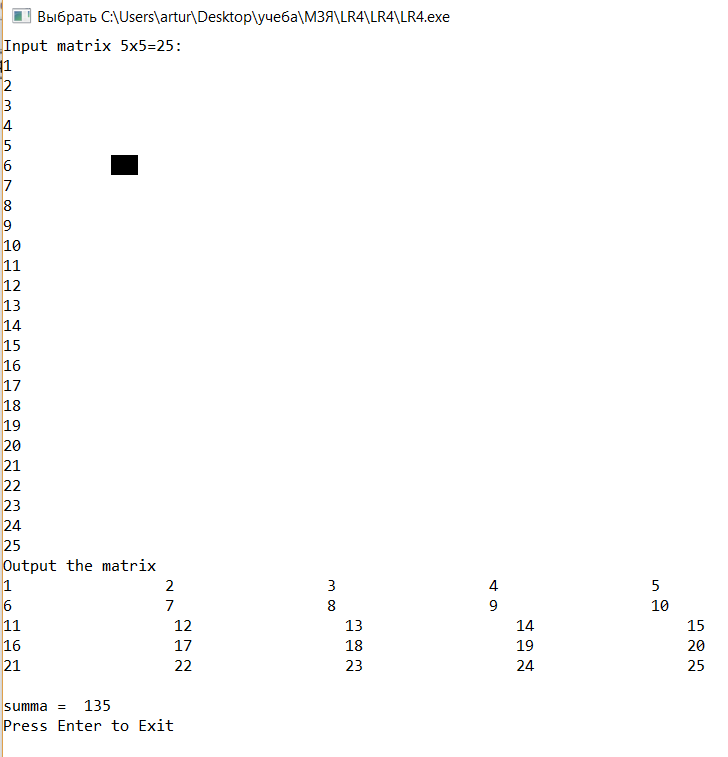
 

Рисунок 2: Результаты выполнения программы

**Вывод**

Таким образом, разработана программа, вычисляющая сумму положительных элементов, находящихся над побочной диагональю. В ходе выполнения работы были усвоены основные принципы работы с массивами и матрицами.