

.386

.model flat, stdcall

option casemap: none

; --- Подключение файлов с кодом, макросами, константами, прототипами функций и т.д.

include c:\masm32\include\windows.inc

include c:\masm32\include\kernel32.inc

include c:\masm32\include\user32.inc

include c:\masm32\include\msvcrt.inc

; --- Подключаемые библиотеки ---

includelib c:\masm32\lib\user32.lib

includelib c:\masm32\lib\kernel32.lib

includelib c:\masm32\lib\msvcrt.lib

.data

fmatr dd ?

smatr dd ?

k dd ?

q\_num dq 0.0

in\_fr db "%i", 0

in\_fr2 db "%lf", 0

out\_fr db "%lf ", 0

out\_fr2 db 13, 10, 0

.code

;allc\_matr(double\*\* matr, int m, int n)

allc\_matr proc

PUSH EAX

PUSH EBX

PUSH ECX

PUSH EDX

MOV EDX, 0

MOV EAX, [ESP + 24]

MOV EBX, [ESP + 28]

MOV ECX, 8

MUL EBX ;Вычисляем размер памяти

MUL ECX

MOV EBX, [ESP + 20] ;ebx = \*\*matr

call GetProcessHeap ;Получаем указатель на кучу

PUSH EAX

PUSH 0

PUSH EAX

call HeapAlloc

MOV [EBX], EAX

POP EDX

POP ECX

POP EBX

POP EAX

RET 12

allc\_matr endp

;free\_matr(double\*\* matr, int m, int n)

free\_matr proc

free\_matr endp

;read\_matr(double\* matr, int m, int n)

read\_matr proc

PUSH ECX

PUSH EBX

MOV EBX, [ESP + 12] ;указатель на матрицу

MOV ECX, [ESP + 16]

rm\_cycl1:

PUSH ECX

MOV ECX, [ESP + 24]

rm\_cycl2:

PUSH ECX

PUSH EBX

PUSH offset in\_fr2

call crt\_scanf

ADD ESP, 8

POP ECX

ADD EBX, 8

LOOP rm\_cycl2

POP ECX

LOOP rm\_cycl1

POP EBX

POP ECX

RET 12

read\_matr endp

;write\_matr(double\* matr, int m, int n)

write\_matr proc

PUSH ECX

PUSH EBX

MOV EBX, [ESP + 12] ;указатель на матрицу

MOV ECX, [ESP + 16]

wr\_cycl1:

PUSH ECX

MOV ECX, [ESP + 24]

wr\_cycl2:

PUSH ECX

PUSH DWORD PTR [EBX + 4]

PUSH DWORD PTR [EBX]

PUSH offset out\_fr

call crt\_printf

ADD ESP, 12

POP ECX

ADD EBX, 8

LOOP wr\_cycl2

PUSH offset out\_fr2

call crt\_printf

ADD ESP, 4

POP ECX

LOOP wr\_cycl1

POP EBX

POP ECX

RET 12

write\_matr endp

;solve(double\* fmatr, double\* smatr, int k)

solve proc

PUSH ESI

PUSH EDI

PUSH ECX

MOV ESI, [ESP + 16]

MOV EDI, [ESP + 20]

MOV ECX, [ESP + 24]

MOV EDX, dword ptr [ESP + 28]

FINIT

sl\_c1:

PUSH ECX

MOV ECX, [ESP + 28]

sl\_c2:

SUB EDX, ECX;

CMP EDX, [ESP]

JE step\_skip

FLD QWORD PTR [ESI]

FSTP QWORD PTR [EDI]

ADD EDI, 8

step\_skip:

ADD ESI, 8

LOOP sl\_c2

POP ECX

LOOP sl\_c1

POP ECX

POP EDI

POP ESI

RET 12

solve endp

start:

PUSH offset k

PUSH offset in\_fr

call crt\_scanf

ADD ESP, 8

PUSH DWORD PTR k

PUSH DWORD PTR k

PUSH offset fmatr

call allc\_matr

MOV EBX, k

SUB EBX, 1

PUSH EBX

PUSH DWORD PTR k

PUSH offset smatr

call allc\_matr

PUSH DWORD PTR k

PUSH DWORD PTR k

PUSH DWORD PTR fmatr

call read\_matr

PUSH DWORD PTR k

PUSH DWORD PTR smatr

PUSH DWORD PTR fmatr

call solve

PUSH EBX

PUSH DWORD PTR k

PUSH DWORD PTR smatr

call write\_matr

call crt\_\_getch

push 0

call ExitProcess

end start