start
{
declare
a : real;
enddeclare
avg([1,4,0.4,3.2]);
}
end
	[0] rsv: avg, [1] 1, [2] 4, [3] +, [4] 0.4, [5] +, [6] 3.2, [7] +, [8] 4, [9] /,
	
	Tabla de Simbolos:
	Pos Nombre
	1 a
	2 _cte1
	3 _cte2
	4 _cte3
	5 _cte4
		.MODEL LARGE ; tipo del modelo de memoria usado.
		.386
		.STACK 200h ; bytes en el stack
		
		include fontdat.inc
		.CODE ;comienzo de la zona de codigo
		include fontaux.inc
		
		start:
		MOV EAX,@DATA ;inicializa el segmento de datos
		MOV DS,EAX
		MOV ES,EAX
		FINIT ;inicializamos la FPU
		FLD Dword Ptr ds:[_cte1]
		FLD Dword Ptr ds:[_cte2]
		
		; suma
		FADD
		FLD Dword Ptr ds:[_cte3]
		
		; suma
		FADD
		FLD Dword Ptr ds:[_cte4]
		
		; suma
		FADD
		FLD Dword Ptr ds:[_cte2]
		
		; division
		FDIV
		
		MOV EAX, 4C00h ; termina la ejecución.
		INT 21h
		END start;
		;

Ejemplo H	**(if y while)**	**Tercetos**
MAIN DEC int a,b real c string d ENDEC BEGINPROG a := 2 b := 1 while(a <= 5 AND b <= 1)	1 (_2 , _ , _) 2 (a , [1] , :=) 3 (_1 , _ , _) 4 (b , [3] , :=) 5 (_ , _ , WHILE) 6 (a , _ , _) 7 (_5 , _ , _) 8 ([6] , [7] , CMP) 9 (_ , _ , AND)	MODEL LARGE .386 .STACK 200h .DATA ;variables de la tabla de simbolos a dw ? msg_a db "a:",'$' aux_a db 10 dup(?),'$' b dw ? msg_b db "b:",'$'

do	10 (b , _ , _)	aux_b db 10 dup(?),'$'
<< 1	11 (_1 , _ , _)	c dd ?
if(b <= 2)	12 ([10] , [11] ,	msg_c db "c:",'$'
<< 1	CMP)	aux_c db 10 dup(?),'$'
b := 6	13 ([12] , [32] ,	d db 32 dup(?),'$'
<< b	BGT)	msg_d db "d:",'$'
fi	14 ([8] , [32] ,	MAXTEXTSIZE EQU 32
a := a + 1	BGT)	NEW_LINE db 0Dh,0Ah,'$'
<< a	15 (_1 , _ , <<)	_1_ dd 1; tiene que ser dd para operaciones aritmeticas
endwhile	16 (_ , _ , IF)	r0 dd ?
a:=0	17 (b , _ , _)	r1 dd ?
b:=0	18 (_2 , _ , _)	r2 dd ?
while(a <> 3 AND b<>3)	19 ([17] , [18] ,	r3 dd ?
do	CMP)	r4 dd ?
a := a + 1	20 ([19] , [25] ,	r5 dd ?
b := b + 1	BGT)	r6 dd ?
endwhile	21 (_1 , _ , <<)	r7 dd ?
<< a	22 (_6 , _ , _)	r8 dd ?
<< b	23 (b , [22] , :=)	r9 dd ?
ENDPROG	24 (b , _ , <<)	r10 dd ?
	25 (_ , _ , FI)	r11 dd ?
	26 (a , _ , _)	r12 dd ?
	27 (_1 , _ , _)	r13 dd ?
	28 ([26] , [27] ,	r14 dd ?
	+)	r15 dd ?
	29 (a , [28] , :=)	r16 dd ?
	30 (a , _ , <<)	r17 dd ?
	31 ([8] , _ , BI)	r18 dd ?
	32 (_ , _ ,	r19 dd ?
	ENDWHILE)	r20 dd ?
	33 (_0 , _ , _)	r21 dd ?
	34 (a , [33] , :=)	r22 dd ?
	35 (_0 , _ , _)	r23 dd ?
	36 (b , [35] , :=)	r24 dd ?
	37 (_ , _ ,	r25 dd ?
	WHILE)	r26 dd ?
	38 (a , _ , _)	r27 dd ?
	39 (_3 , _ , _)	r28 dd ?
	40 ([38] , [39] ,	r29 dd ?
	CMP)	r30 dd ?
	41 (_ , _ , AND)	r31 dd ?
	42 (b , _ , _)	r32 dd ?
	43 (_3 , _ , _)	r33 dd ?
	44 ([42] , [43] ,	r34 dd ?
	CMP)	r35 dd ?
	45 ([44] , [56] ,	r36 dd ?
	BEQ)	r37 dd ?
	46 ([40] , [56] ,	r38 dd ?
	BEQ)	r39 dd ?
	47 (a , _ , _)	r40 dd ?

Borre varias hojas de funciones en el medio

```
mov [bx],ax ; lo guardamos en la cadena final
inc bx
loop itoa_3

itoa_4:
mov ax,'$'  ; terminar cadena con '$' para
mov [bx],ax ; imprimirla con la INT21h/AH=9
ret
itoa endp
;*******************************************************
            ; ========= Convertir cadena a numero ===============ATOI======
; Parametros
; si: offset inicial de la cadena con respecto a DS
; Retorna
; bx: valor
atoi proc
 xor bx,bx   ;BX = 0

 atoi_1:
 lodsb      ;carga byte apuntado por SI en AL
           ;e incrementa si
 cmp al,'0'  ;es numero ascii? [0-9]
 jb noascii  ;no, salir
 cmp al,'9'
 ja noascii  ;no, salir

 sub al,30h  ;ascii '0'=30h, ascii '1'=31h...etc.
 cbw        ;byte a word
 push ax
 mov ax,bx   ;BX tendra el valor final
 mov cx,10
 mul cx     ;AX=AX*10
 mov bx,ax
 pop ax
 add bx,ax
 jmp atoi_1  ;seguir mientras SI apunte a un numero ascii
 noascii:
 ret        ;BX tiene el valor final
atoi endp
;FIN funciones

MAIN:
MOV AX,@DATA
MOV DS,AX
FINIT; Inicializa el coprocesador

MOV r0,2
FILD r0
FIST a
FFREE st(0)          ;FIN ASIGNACION
```

```
MOV r0,1
FILD r0
FIST b
FFREE st(0)          ;FIN ASIGNACION

WHILE0:              ;INICIO WHILE0
FILD a
FIST r0
FFREE st(0)
MOV r1,5
FILD r0
FILD r1
FCOMP
FSTSW ax
FWAIT
SAHF
FFREE st(0)
MOV auxEstado, ax
FILD b
FIST r2
FFREE st(0)
MOV r3,1
FILD r2
FILD r3
FCOMP
FSTSW ax
FWAIT
SAHF
FFREE st(0)
JB FIN_WHILE0
MOV ax,auxEstado
FWAIT
SAHF
JB FIN_WHILE0

;imprimiendo entero
mov ax,1
mov bx,offset _raux_cad1_
  call itoa
  MOV DX,OFFSET _raux_cad1_
  MOV Ah,9
  INT 21h
  MOV DX,OFFSET NEW_LINE
  MOV Ah,9
  INT 21h
FILD b
FIST r4
FFREE st(0)
MOV r5,2
```

```
FILD r4
FILD r5
FCOMP
FSTSW ax
FWAIT
SAHF
FFREE st(0)
JB FIN_IF0

;imprimiendo entero
mov ax,1
mov bx,offset _raux_cad1_
  call itoa
  MOV DX,OFFSET _raux_cad1_
  MOV Ah,9
  INT 21h
  MOV DX,OFFSET NEW_LINE
  MOV Ah,9
  INT 21h
MOV r6,6
FILD r6
FIST b
FFREE st(0)          ;FIN ASIGNACION


;imprimiendo entero
mov ax,b
mov bx,offset _raux_cad1_
  call itoa
  MOV DX,OFFSET _raux_cad1_
  MOV Ah,9
  INT 21h
  MOV DX,OFFSET NEW_LINE
  MOV Ah,9
  INT 21h
FI0:
FIN_IF0:              ;si no hay else
FILD a
FIST r6
FFREE st(0)
MOV r7,1
FILD r7
FILD r6
FADD
FIST r6
FFREE st(0)
FILD r6
FIST a
FFREE st(0)          ;FIN ASIGNACION
```

```
;imprimiendo entero
mov ax,a
mov bx,offset _raux_cad1_
  call itoa
  MOV DX,OFFSET _raux_cad1_
  MOV Ah,9
  INT 21h
  MOV DX,OFFSET NEW_LINE
  MOV Ah,9
  INT 21h
JMP WHILE0
FIN_WHILE0:      ;FIN_WHILE0
MOV r6,0
FILD r6
FIST a
FFREE st(0)      ;FIN ASIGNACION

MOV r6,0
FILD r6
FIST b
FFREE st(0)      ;FIN ASIGNACION

WHILE1:          ;INICIO WHILE1
FILD a
FIST r6
FFREE st(0)
MOV r7,3
FILD r6
FILD r7
FCOMP
FSTSW ax
FWAIT
SAHF
FFREE st(0)
MOV auxEstado, ax
FILD b
FIST r8
FFREE st(0)
MOV r9,3
FILD r8
FILD r9
FCOMP
FSTSW ax
FWAIT
SAHF
FFREE st(0)
JE FIN_WHILE1
MOV ax,auxEstado
FWAIT
```

```
SAHF
JE FIN_WHILE1
FILD a
FIST r10
FFREE st(0)
MOV r11,1
FILD r11
FILD r10
FADD
FIST r10
FFREE st(0)
FILD r10
FIST a
FFREE st(0)          ;FIN ASIGNACION

FILD b
FIST r10
FFREE st(0)
MOV r11,1
FILD r11
FILD r10
FADD
FIST r10
FFREE st(0)
FILD r10
FIST b
FFREE st(0)          ;FIN ASIGNACION

JMP WHILE1
FIN_WHILE1:       ;FIN_WHILE1

;imprimiendo entero
mov ax,a
mov bx,offset _raux_cad1_
  call itoa
  MOV DX,OFFSET _raux_cad1_
  MOV Ah,9
  INT 21h
  MOV DX,OFFSET NEW_LINE
  MOV Ah,9
  INT 21h

;imprimiendo entero
mov ax,b
mov bx,offset _raux_cad1_
  call itoa
  MOV DX,OFFSET _raux_cad1_
  MOV Ah,9
  INT 21h
  MOV DX,OFFSET NEW_LINE
```

		MOV Ah,9
		INT 21h
		
		FINAL:
		mov ah, 1 ; pausa, espera que oprima una tecla
		int 21h ; AH=1 es el servicio de lectura
		MOV AX, 4C00h ; Sale del Dos
		INT 21h ; Enviamos la interripcion 21h
		END ; final del archivo.