Compilador

TPAño revisión
2018 .1

Prof.: Mara Capuya, Hernán Villarreal

Aytes: Daniel Carrizo – Facundo Velazquez Santillán

Ejemplo A	(if con and)	Arbol
tdeclare	nodo 1:, var1,	CODE
real var1,var2;		;INICIALIZA EL SEGMENTO DE DATOS
tenddeclare	nodo 0:, _5,	
teridueciare		MOVES, @DATA
	nodo 2: 1, :=, 0	MOV DS, EAX
var1:=5	nodo 3:, var1,	;CODIGO DEL PROGRAMA EN ASM
tif var1 >4 and var1!=8	nodo 4:, _4,	;asignacion
var1:=8	nodo 5: 3, >, 4	mov eax, cte0
	nodo 6:, var1,	
inelse var1:=10	11000 0, vai 1,	mov varasm_var1, eax
	 	fld_varasm_var1
outelse	nodo 7:, _8,	fld cte1
	nodo 8: 6, !=, 7	fcomp ;compara y popea el operando sobrante
fintif	nodo 9: 5, and,	ffree St(0)
	8	fstsw ax
	nodo 11:,	sahf
	var1,	JB CONDIC_VERDADERAO Jump if Below
	nodo 10:, _8,	mov result_comp1, 0
		jmp CONDIC_FALSA1
	nodo 12: 11. :=.	CONDIC_VERDADERAO:
	10	mov result comp1, 1
		CONDIC_FALSA1:
	var1,	fld varasm_var1
	nodo 13:, _10,	fld cte2
	11000 13, _10,	
	nodo 15: 14, :=,	ffree St(0)
	13	fstsw ax
	nodo 16: 12,	sahf
	cuerpoif, 15	JNE CONDIC_VERDADERA2
	nodo 17: 9, tif,	mov result_comp2, 0
	16	jmp CONDIC_FALSA3
	nodo 18: 2,	CONDIC_VERDADERA2:
	sentencia, 17	mov result_comp2, 1
		CONDIC_FALSA3:
		mov eax, result_comp1
		AND eax, result comp2
	Raiz : nodo 18	mov result_logico3, eax
		mov eax, result_logico3
		cmp eax, 0
		JE ELSE_IF4
		;asignacion
		(mov eax, cte2)
		mov varasm_var1, eax
		JMP FIN_IF5
		ELSE_IF4:
		;asignacion
		mov eax, cte3
		mov varasm_var1, eax
	1	(1112 - 1211 - 1

revisión 2018 .1

FIN_IF5:								
Figure F			FIN_IF5:	:				
MOV AH, 01			_					
INT 21h ;TERMINA LA EJECUCION DEL PROGRAMA TERMINAPROG:					;Espera	pulsar ι	una	tecla
CODE					MOV A	H, 01		
TERMINAPROG:					iNT 21h	1		
TERMINAPROG:			·TERMIN	ΙΔΙΔ ΕΙΙ	FCLICION	DEL PR	OGR	ΑΜΑ
INT 21h IDATA			-			DELTIN	.001	MINIC
DATA					MOV E	4X, 4C00	0h	
varasm_var1 dd ? varasm_var2 dd ? cte0 dd 5 cte0 dd 5 cte1 dd 4 cte2 dd 8 cte3 dd 10 result_comp1 dd ? result_comp2 result_comp2 dd ? result_comp2 result_co					INT 21h	1		
Cte0 dd 5			.DATA					
Cte0 dd 5				varasm	_var1		3	
Cte1					_	dd	3	
Cte2 dd 8 Cte3 dd 10 result_comp1 dd ? result_comp2 dd ? result_logico3 dd ?								
Cte3 dd 10 result_comp1 dd ? result_comp2 dd ? result_logico3 dd ?								
Tesult_comp1 dd ?								
Tesult_comp2 dd ? result_logico3 dd ? result_logico3 dd ?							_	
Tesult_logico3 dd ?				_	-			
END END				_				
Ejemplo B (inlist dentro de un if) tdeclare real var1; tenddeclare var2: tenddeclare var3: var1:=3 var1:=5 var1:=5 nodo 3:, -5, nodo 5: 4, :=, 3 nodo 6: -2, var1:=39 tif tinlist(var1;[1;4;2]) var1:=39 inelse nodo 11:, -2, var1, inelse nodo 11:, -2, var1:=6 outelse nodo 9:, -4, nodo 9:, -4, nodo 7:, 1, fintif nodo 10: -9, // Inlist es una sentencia que toma una expresión nodo 12: 11, // Inlist es una sentencia que toma una expresión nodo 12: 11, // Inlist es una sentencia que toma una expresión nodo 12: 11, // Inlist es una sentencia que toma una expresión nodo 12: 11, // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una sentencia que toma una expresión // Inlist es una expresión			END	resuit_i	logicos	uu	·	
Ideclare nodo 1:, var1, real var1; ;INICIALIZA EL SEGMENTO DE DATOS var1:=3 nodo 0:, _3, nodo 2: 1, :=, 0 nodo 4:, var1, var1:=5 MOV EAX, @DATA MOV DS, EAX var1:=5 nodo 3:, _5, nodo 5: 4, :=, 3 nodo 6: 2, sasignacion nodo 5: 4, :=, 3 nodo 6: 2, sentencia, 5 nodo 13:, var1, nodo 13:, var1, nodo 11:, _2, fld Dword Ptr ds:[varasm_var1, eax nodo varasm_var1, e			LIND					
tdeclare nodo 1:, var1, lenddeclare ;;iNICIALIZA EL SEGMENTO DE DATOS var1:=3 nodo 0:, _3, nodo 2: 1, :=, 0 MOV EAX, @DATA MOV DS, EAX var1:=3 nodo 4:, var1, nodo 3:, _5, nodo 5: 4, :=, 3 nodo 6: 2, sasignacion ;asignacion mov eax, cte0 mov varasm_var1, eax tif tinlist(var1;[1;4;2]) sentencia, 5 nodo 13:, var1, modo 11:, _2, fld Dword Ptr ds:[varasm_var1] mov varasm_var1, eax inelse nodo 11:, _2, fld Dword Ptr ds:[cte2] fcomp ;compara y popea el operando sobrante nodo 7:, _1, nodo 8: 7, lista, nodo 8: 7, lista, nodo 10: 9, fstsw ax sahf firee St(0) fstsw ax sahf fintif nodo 10: 9, lista, nodo 10: 9, lista, 8 ura sentencia que toma una expresión lista, 8 nodo 12: 11, lista, 9 jmp CONDIC_FALSA1	Ejemplo B	(inlist dentro	Arbol					
real var1; tenddeclare		de un if)						
tenddeclare nodo 0:, _3,	tdeclare	nodo 1:, var1,	.CODE					
var1:=3 nodo 2: 1, :=, 0 nodo 4:, var1, ;CODIGO DEL PROGRAMA EN ASM var1:=5 nodo 3:, _5, nodo 5: 4, :=, 3 mov eax, cte0 mov eax, cte0 mov varasm_var1, eax tif tinlist(var1;[1;4;2]) nodo 6: 2, sentencia, 5 nodo 13:, var1, mov eax, cte1 mov eax, cte1 mov varasm_var1, eax inelse nodo 11:, _2, fld Dword Ptr ds:[varasm_var1] mov varasm_var1, eax outelse nodo 9:, _4, nodo 7:, _1, ffree St(0) fcomp ;compara y popea el operando sobrante fintif nodo 8: 7, lista, nodo 10: 9, lista, 8 nodo 10: 9, lista, 8 nodo 12: 11, jmp CONDIC_FALSA1 mov result_comp2, 0 jmp CONDIC_FALSA1	real var1;				;INICIA	LIZA EL S	SEGN	MENTO DE DATOS
var1:=3	tenddeclare						ATA	
var1:=5					MOV D	S, EAX		
var1:=5 nodo 3:, _5, nodo 5: 4, :=, 3 ;asignacion mov eax, cte0 tif tinlist(var1;[1;4;2]) nodo 6: 2, sentencia, 5 nodo 13:, var1, modo 11:, _2, var1, nodo 11:, _2, var1:=6 mov varasm_var1, eax mov varasm_var1, eax fld Dword Ptr ds:[varasm_var1] inelse nodo 9:, _4, nodo 9:, _4, nodo 7:, _1, fld Dword Ptr ds:[cte2] fld Dword Ptr ds:[cte2] outelse nodo 9:, _4, nodo 7:, _1, ffree St(0) fstsw ax sahf // Inlist es una sentencia que toma una expresión lista, 8 nodo 12: 11, jmp CONDIC_FALSA1 jmp CONDIC_FALSA1	var1:=3	nodo 4:, var1,						
tif tinlist(var1;[1;4;2]) nodo 5: 4, :=, 3 nodo 6: 2, sentencia, 5 sentencia, 5 nodo 13:, var1, mov varasm_var1, eax saignacion mov eax, cte1 mov varasm_var1, eax inelse nodo 11:, _2, fld Dword Ptr ds:[varasm_var1] var1:=6 fld Dword Ptr ds:[cte2] outelse nodo 9:, _4, nodo 7:, _1, free St(0) fintif nodo 8: 7, lista, nodo 10: 9, modo 10: 9, lista, 8 nodo 10: 11. mov result_comp2, 0 nodo 12: 11, jmp CONDIC_FALSA1					-		PROG	GRAMA EN ASM
tif tinlist(var1;[1;4;2]) var1:=39 sentencia, 5 nodo 13:, var1, inelse nodo 11:, _2, var1:=6 outelse nodo 9:, _4, nodo 7:, _1, fintif nodo 8: 7, lista,	var1:=5				_			
var1:=39 sentencia, 5 nodo 13:, var1, inelse nodo 11:, _2, var1:=6 outelse nodo 9:, _4, nodo 7:, _1, fintif nodo 8: 7, lista, nodo 10: 9, // Inlist es una sentencia que toma una expresión sentencia, 5 nodo 13:, mov eax, cte1 mov eax, cte1 mov eax, cte1 mov eax, cte1 fld Dword Ptr ds:[varasm_var1] fld Dword Ptr ds:[cte2] fcomp ;compara y popea el operando sobrante ffree St(0) fstsw ax sahf mov eax, cte1 mov eax, cte1 mov eax, cte1 mov eax, cte1 fld Dword Ptr ds:[varasm_var1] fstsw ax schencia, 5 mov eax, cte1 mov eax, cte2 mov eax, cte2 mov eax, cte2 mov eax, cte2 mov e							_	
nodo 13:, var1, inelse nodo 11:, _2, fld Dword Ptr ds:[varasm_var1] var1:=6 outelse nodo 9:, _4, nodo 7:, _1, fintif nodo 8: 7, lista, fintif nodo 10: 9, // Inlist es una sentencia que toma una expresión nodo 12: 11, mov eax, cte1 mov varasm_var1, eax fld Dword Ptr ds:[varasm_var1] fld Dword Ptr ds:[cte2] fcomp ;compara y popea el operando sobrante ffree St(0) fstsw ax sahf JE CONDIC_VERDADERAO mov result_comp2, 0 jmp CONDIC_FALSA1		-					ar1,	eax
inelse nodo 11:, _2, var1:=6 fld Dword Ptr ds:[varasm_var1] outelse nodo 9:, _4, nodo 7:, _1, ffree St(0) fintif nodo 8: 7, lista, nodo 10: 9, // Inlist es una sentencia que toma una expresión nodo 12: 11, modo varasm_var1, eax fld Dword Ptr ds:[varasm_var1] fld Dword Ptr ds:[varasm_varasm	var1:=39							
inelse							ar1	024
var1:=6 outelse nodo 9:, _4, nodo 7:, _1, fintif nodo 8: 7, lista, nodo 10: 9, // Inlist es una sentencia que toma una expresión fild Dword Ptr ds:[cte2] fcomp ;compara y popea el operando sobrante ffree St(0) fstsw ax sahf JE CONDIC_VERDADERAO mov result_comp2, 0 jmp CONDIC_FALSA1	inalsa							
outelse nodo 9:, _4, nodo 7:, _1, fintif nodo 8: 7, lista, nodo 10: 9, // Inlist es una sentencia que toma una expresión nodo 12: 11,		110u0 11, _2,					-	<u> </u>
fintif nodo 7:, _1, nodo 8: 7, lista, nodo 10: 9, // Inlist es una sentencia que toma una expresión nodo 7:, _1, nodo 8: 7, lista, sahf JE CONDIC_VERDADERAO mov result_comp2, 0 jmp CONDIC_FALSA1		nodo 9· 4						
fintif nodo 8: 7, lista, sahf nodo 10: 9, // Inlist es una sentencia que toma una expresión nodo 12: 11, fstsw ax sahf JE CONDIC_VERDADERAO mov result_comp2, 0 jmp CONDIC_FALSA1	Jacobe					(0)	, c	ompara y popea el operariao sobiante
sahf JE CONDIC_VERDADERAO mov result_comp2, 0 jmp CONDIC_FALSA1	fintif	_						
nodo 10: 9, JE CONDIC_VERDADERA0 mov result_comp2, 0 jmp CONDIC_FALSA1								
// Inlist es una sentencialista, 8mov result_comp2, 0que toma una expresiónnodo 12: 11,jmp CONDIC_FALSA1						DIC VEF	RDAI	DERA0
que toma una expresión nodo 12: 11, jmp CONDIC_FALSA1	// Inlist es una sentencia	-				_		
pivot y una lista de lista, 10 CONDIC_VERDADERAO:	pivot y una lista de		CONDIC	_VERDA		_		
expresiones; y devuelve V nodo 14: 13, mov result_comp2, 1				_		sult_con	mp2,	1
o F si el resultado de la tinlist, 12 CONDIC_FALSA1:			l			_	-	
expresión pivot está o no nodo 16:, fld Dword Ptr ds:[varasm_var1]	o F si el resultado de la	tinlist, 12	CONDIC	_FALSA1	L:			

Prof.: Mara Capuya, Hernán Villarreal

Ejemplos Varios

TP

Año revisión 2018 .1

1 1: 1	1 4	
en la lista	var1,	fld Dword Ptr ds:[cte2]
	nodo 15:, _39,	fcomp ;compara y popea el operando sobrante
		ffree St(0)
	nodo 17: 16, :=,	fstsw ax
	15	sahf
	nodo 19:,	JE CONDIC_VERDADERA2
	var1,	mov result_comp3, 0
	nodo 18:, _6,	jmp CONDIC_FALSA3
		CONDIC_VERDADERA2:
	nodo 20: 19, :=,	mov result_comp3, 1
	18	CONDIC_FALSA3:
	nodo 21: 17,	mov eax, result_comp3
	cuerpoif, 20	OR eax, result_comp2
	nodo 22: 14, tif,	mov result_logico1, eax
	21	fld Dword Ptr ds:[varasm_var1]
	nodo 23: 6,	fld Dword Ptr ds:[cte3]
	sentencia, 22	fcomp ;compara y popea el operando sobrante
	,	ffree St(0)
		fstsw ax
		sahf
		JE CONDIC_VERDADERA4
		mov result_comp5, 0
		jmp CONDIC_FALSA5
		CONDIC_VERDADERA4:
		mov result_comp5, 1
		CONDIC_FALSA5:
		mov eax, result_comp5
		OR eax, result_logico1
		mov result_logico4, eax
		fld Dword Ptr ds:[varasm_var1]
		fld Dword Ptr ds:[cte4]
		fcomp ;compara y popea el operando sobrante
		ffree St(0)
		fstsw ax
		sahf
		JE CONDIC_VERDADERA6
		mov result_comp7, 0
		jmp CONDIC_FALSA7
		CONDIC_VERDADERA6:
		mov result_comp7, 1
		CONDIC_FALSA7:
		mov eax, result_comp7
		OR eax, result_logico4
		mov result_logico6, eax
		mov eax, result_logico6
		cmp eax, 0
		JE ELSE_IF8
		;asignacion
		mov eax, cte5
		mov varasm_var1, eax

Prof.: Mara Capuya, Hernán Villarreal

Ejemplos Varios

TP Compilador Año revisión 2018 .1

			JMP FIN	I IF9	
		ELSE_IF8:		_	
		_	;asigna	cion	
			mov ea		
			mov va	rasm_va	ar1, eax
		FIN_IF9:			
			_		
					una tecla
			MOV A		
			INT 21h		
			;TERMI	NA LA EJ	JECUCION DEL PROGRAMA
		TERMINAPROC			
				XX, 4C00	Oh .
			INT 21h		
		.DATA			
		1	suffer db 1	4 dup (0));necesario para el uso interno de la rutina
		formatoReal	m va=1	44	2
		varasi cte0	m_var1 dd	dd 3	?
		cte0	dd	5	
		cte1	dd	1	
		cte3	dd	4	
		cte4	dd	2	
		cte5	dd	39	
		cte6	dd	6	
			_logico1	dd	?
			_ comp2	dd	?
			_ comp3	dd	?
			_logico4	dd	?
			_comp5	dd	?
			_logico6	dd	?
			_comp7	dd	?
		END			
Ejemplo C	(round y	Tercetos			
	trunc)				

Prof.: Mara Capuya, Hernán Villarreal

Año revisión 2018 .1

PROGRAM	1 (:=,	.MODEL LARGE	
DECLARE	var1, _cte_1)	.386	
REAL var1, var2, var3,	2 (:=,	.STACK 200h	
var4;	var2, _cte_2)		
ENDDECLARE	3	.CODE	
	(ROUN		
var1 := .9;	D, var1,)	;CODIGO ASSEMBLER	
var2 := 6.9;	4 (:=,	;	
var3 := ROUND(var1);	var3, [3])		
var4 := TRUNC(var2);	5	mov ax,@DATA ;limpia buffe	r
	(TRUNC	mov ds,ax	
ENDPROGRAM	, var2,)	mov es,ax	
	6 (:=,		
	var4, [5])	FLD _cte_1	
		FSTP var1	
		FLD _cte_2	
		FSTP var2	
		FLD var1	
		FRNDINT	
		FSTP _aux_1	
		FLD _aux_1	
		FSTP var3	
		FLD _uno	
		FLD var2	
		FPREM	
		FSTP _resto_trunc	
		FLD var2	
		FSUB _resto_trunc	
		FSTP _aux_1	
		FLD _aux_1	
		FSTP var4	
		mov dx,OFFSET msgPRESIONE	;imprimir mensaje de espera
		mov ah,09	
		int 21h	
			ra que oprima una tecla
		int 21h	
		mov ax, 4C00h	;fin de ejecucion
		int 21h	
		.DATA	
		;DEFINICION DE VARIABLES	
		;	
		var1 dd 0	
		var2 dd 0	
		var3 dd 0	
		var4 dd 0	
		_cte_1 dd 0.900000	
		_cte_2 dd 6.900000	
		_aux_1 dd 0	
	·		

Prof.: Mara Capuya, Hernán Villarreal

TP Compilador Año revisión 2018 .1

_aux_a dw 0 _aux_b dw 0 _uno dd 1.000000 resto trunc dd 0 msgPRESIONE db 0DH,0AH,"Presione una tecla para continuar...",'\$' _NEWLINE db 0Dh,0Ah,'\$' ;FIN DEFINICION DE VARIABLES **END** Ejemplo D (STRING) **Polaca DECLARE HOLA** .MODEL LARGE .386 string pepe, pepe2; pepe **ENDDECLARE** .STACK 200h MUNDO pepe MAXTEXTSIZE equ 50 pepe = 'HOLA' **CONCATENATE** .DATA pepe concatenate(pepe, pepe2 'MUNDO') MAXTEXTSIZE dup (?),'\$' pepe_ db pepe2 db MAXTEXTSIZE dup (?),'\$' pepe2 = pepe db "HOLA ",'\$', 42 dup (?) zv2 zv3 db "MUNDO ",'\$', 41 dup (?) db aux .CODE mov AX,@DATA; inicializa el segmento de datos mov DS,AX mov es,ax; mov si,OFFSET zv2 mov di,OFFSET pepe_ CALL COPIAR; copia los strings mov si,OFFSET zv3 mov di,OFFSET pepe_ CALL CONCAT; cat los strings mov si,OFFSET pepe_

mov di,OFFSET pepe2_

CALL COPIAR; copia los strings

TP Año **revisión** 2018 .**1**

Prof.: Mara Capuya, Hernán Villarreal Aytes: Daniel Carrizo – Facundo Velazquez Santillán

		mov ax, 4C00h ; Indica que debe terminar la ejecución int 21h
Ejemplo E	(for)	Polaca
DECLARE	4	.MODEL LARGE
	Var1	.386
real var2 , var1;	=	.STACK 200h
ENIDDES! ADE	30	AAAVTEVTCITE EQ
ENDDECLARE	Var2	MAXTEXTSIZE equ 50
	=	DATA
	1	.DATA
var1 = 4	Var1	
var2 = 30	=	var2_ dd ?
	eti9:	var1 dd ?
for var1 = 1 to 3	3	zv2 dd 040800000h;valor=4
{	Var1	zv3 dd 041f00000h;valor=30
var2 = var2 - 1	<>	zv4 dd 03f800000h;valor=1
	_eti27 BF	zv5 dd 040400000h;valor=3
1		aux db ?
}	Var2	.CODE
next var1	1	mov AX,@DATA ; inicializa el segmento de datos
	Var2	mov DS,AX
	=	mov es,ax ;
	Var1	fld zv2
	1	fstp var1_
	+	fld zv3
	Var1	fstp var2_
	=	fld zv4
	eti9	fstp var1
	BI	_eti9:
	_eti27:	fld zv5

TP Compilador

Año revisión 2018 .1

Prof.: Mara Capuya, Hernán Villarreal

Aytes: Daniel Carrizo – Facundo Velazquez Santillán

Compila

```
fld var1
                                                    fxch
                                                    fcomp
                                                    fstsw ax
                                                    ffree st(0)
                                                    sahf
                                                    je _eti27
                                                    fld var2_
                                                    fld zv4
                                                    fsub
                                                    fstp var2_
                                                    fld var1_
                                                    fld zv4
                                                    fadd
                                                    fstp var1_
                                                    jmp _eti9
                                            eti27:
                                            mov ax, 4C00h; Indica que debe terminar la ejecución
                                            int 21h
Ejemplo F
                           (while con or Polaca
DECLARE
                          4
                                            .MODEL LARGE
                                            .386
        real var2, var1;
                          var1
                          =
                                            .STACK 200h
                          3
ENDDECLARE
                          Var2
                                            MAXTEXTSIZE equ 50
                                            .DATA
var1 = 4
                           _eti6:
var2 = 3
                          Var2
                                                                     ?
                          1
                                                    var2
                                                            dd
                           <>
                                                    var1_
                                                            dd
while var2 <> 1 AND var1
                           _eti34
                                                    zv2
                                                            dd
                                                                     040800000h;valor=4
> var2
                          BF
                                                            dd
                                                                     040400000h;valor=3
                                                    zv3
                          Var1
                                                            dd
                                                                     03f800000h;valor=1
                                                    zv4
        var2 = var2 - 1
                          Var2
                                                    zv5
                                                            db
                                                                     "IF2 ",'$', 43 dup (?)
                                                            db
                                                                     "POSTIF2 ",'$', 39 dup (?)
                                                    zv6
                          >
                                                            db
  if var2 == var1
                           _eti34
                                                    aux
                                                                     ?
                          BF
                                            .CODE
        WRITE('IF2')
                          Var2
  }
                          1
                                                    mov AX,@DATA; inicializa el segmento de datos
  endif
                                                    mov DS,AX
  WRITE( 'POSTIF2 ')
                          Var2
                                                    mov es,ax;
                          =
                                                    fld zv2
                          Var2
                                                    fstp var1
endwhile
                          var1
                                                    fld zv3
```

revisión 2018 .1

== fstp var2_ _eti29 _eti6: BF fld var2_ "IF2" fld zv4 fxch WRITE _eti29: fcomp "POSTIF2" fstsw ax WRITE ffree st(0) sahf eti6 ВΙ je _eti34 _eti34: fld var1_ fld var2_ fxch fcomp fstsw ax ffree st(0) sahf jbe _eti34 fld var2_ fld zv4 fsub fstp var2_ fld var2_ fld var1_ fxch fcomp fstsw ax ffree st(0) sahf jne_eti29 mov DX, OFFSET zv5 mov ah, 9; AH=9 es el servicio de impresion int 21h; imprime _eti29: mov DX, OFFSET zv6 mov ah, 9; AH=9 es el servicio de impresion int 21h; imprime jmp _eti6 _eti34: mov ax, 4C00h; Indica que debe terminar la ejecución int 21h Ejemplo G (average) **Polaca**

TP Compilador

revisión 2018 .1

Prof.: Mara Capuya, Hernán Villarreal

[0] rsv: avg, [1] .MODEL LARGE ; tipo del modelo de memoria usado. start { 1, [2] 4, [3] +, .386 .STACK 200h [4] 0.4, [5] +, [6] ; bytes en el stack 3.2, [7] +, [8] 4, declare a : real; [9] /, enddeclare include fontdat.inc de .CODE ;comienzo de la zona de codigo avg([1,4,0.4,3.2]); Tabla Simbolos: include fontaux.inc } Pos end Nombr start: MOV EAX,@DATA ;inicializa el segmento de datos e 1 MOV DS,EAX **MOV ES, EAX** 2 **FINIT** :inicializamos la FPU _cte1 FLD Dword Ptr ds:[_cte1] 3 FLD Dword Ptr ds:[_cte2] _cte2 4 _cte3 ; suma **FADD** 5 FLD Dword Ptr ds:[_cte3] cte4 ; 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; (if y while $\overline{)}$ Ejemplo H **Tercetos** MAIN MODEL LARGE 1(_2,_,_) **DEC** 2 (a,[1],:=) .386 int a,b .STACK 200h 3(_1,_,_)

real c 4 (b, [3], :=) 5 (_ , _ , WHILE .DATA string d **ENDEC** ;variables de la tabla de simbolos a dw? 6(a,_,_) **BEGINPROG** 7(_5,_,_) msg_a db "a:",'\$' a := 28 ([6],[7], aux_a db 10 dup(?),'\$' b := 1 CMP) b dw? 9(_,_,AND) msg_b db "b:",'\$' while(a <= 5 AND b <= 1)

TP

Año revisión 2018 .1

```
do
                         10(b,_,_)
                                          aux b db 10 dup(?),'$'
                         11(_1,_,_)
        << 1
                                          c dd?
       if(b \le 2)
                         12 ([10],[11],
                                          msg_c db "c:",'$'
                                          aux c db 10 dup(?),'$'
                << 1
                         CMP)
               b := 6
                         13 ([12], [32],
                                          d db 32 dup(?),'$'
               << b
                         BGT)
                                          msg d db "d:",'$'
                                          MAXTEXTSIZE EQU 32
        fi
                         14 ([8], [32],
       a := a + 1
                         BGT)
                                          NEW_LINE db 0Dh,0Ah,'$'
                         15 ( _1 , _ , << )
                                          1 dd 1; tiene que ser dd para operaciones aritmeticas
        << a
endwhile
                                          r0 dd?
                         16 ( _ , _ , IF )
a:=0
                         17 (b,_,_)
                                          r1 dd?
                         18 ( _2 , _ , _ )
b:=0
                                          r2 dd?
while( a <> 3 AND b<>3)
                         19 ([17],[18],
                                         r3 dd?
                         CMP)
                                          r4 dd?
        a := a + 1
                         20 ([19], [25],
                                         r5 dd?
        b := b + 1
                         BGT)
                                          r6 dd?
endwhile
                                          r7 dd?
                         21 ( _1 , _ , << )
<< 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?
                                          r23 dd?
                         35 (_0,_,_)
                         36 (b, [35], :=) r24 dd?
                         37 ( _ , _ ,
                                          r25 dd?
                         WHILE)
                                          r26 dd?
                                          r27 dd?
                         38 (a,_,_)
                         39 ( _3 , _ , _ )
                                          r28 dd?
                         40 ([38],[39],
                                         r29 dd?
                         CMP)
                                          r30 dd?
                         41(_,_,AND)
                                         r31 dd?
                                          r32 dd?
                         42 (b, _, _)
                         43 ( _3 , _ , _ )
                                          r33 dd?
                         44 ([42],[43],
                                          r34 dd?
                                          r35 dd?
                         CMP)
                         45 ([44], [56],
                                         r36 dd?
                         BEQ )
                                          r37 dd?
                         46 ([40], [56],
                                         r38 dd ?
                         BEQ )
                                          r39 dd?
                         47 (a,<u>,</u>)
                                          r40 dd?
```

Año revisión 2018 .1

## (147) [48] r42 dd ? r42 dd ? r43 dd ? r43 dd ? r43 dd ? r43 dd ? r45 dd ? r45 dd ? r46 dd ? r46 dd ? r46 dd ? r47 dd ? r47 dd ? r48 dd ? r49		
**) \$0 (a , [49], :=) r44 dd ? \$1 (b ,) \$2 (_ 1 ,) \$3 ([51], [52], :=) r45 dd ? \$4 (b , [53], :=) r49 dd ? \$5 ([40], Bl) \$6 (, < _) \$7 (a , _ , < <) \$8 (b , _ , < <) \$9 dd ? raux dd ? op3 dd ?raux 3 dd ? op5 dd ?raux 3 dd ? op5 dd ?raux 3 dd ? op6 dd ?raux 3 dd ? op7 dd ?raux 3 dd ? op8 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op6 dd ?raux 3 dd ? op7 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op1 dd ?raux 4 dd ? op1 dd ?raux 5 dd ? op1 dd ?raux 6 d ? op1 dd ?raux 7 dd ? op1 dd ?	48 (_1 , _ , _)	r41 dd ?
**) \$0 (a , [49], :=) r44 dd ? \$1 (b ,) \$2 (_ 1 ,) \$3 ([51], [52], :=) r45 dd ? \$4 (b , [53], :=) r49 dd ? \$5 ([40], Bl) \$6 (, < _) \$7 (a , _ , < <) \$8 (b , _ , < <) \$9 dd ? raux dd ? op3 dd ?raux 3 dd ? op5 dd ?raux 3 dd ? op5 dd ?raux 3 dd ? op6 dd ?raux 3 dd ? op7 dd ?raux 3 dd ? op8 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op6 dd ?raux 3 dd ? op7 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op9 dd ?raux 3 dd ? op1 dd ?raux 4 dd ? op1 dd ?raux 5 dd ? op1 dd ?raux 6 d ? op1 dd ?raux 7 dd ? op1 dd ?	49 ([47],[48],	r42 dd ?
S1 (b, _ , _) F45 dd ? F46 dd ? F47 dd ? F48		
S1 (b, _ , _) F45 dd ? F46 dd ? F47 dd ? F48	50 (a, [49], :=)	r44 dd ?
\$2(_1,,) \$3([5],[52], ref dd? +1) \$4(b,[53],=) \$5([40],_,8] \$6(,,<) \$7(a,_,<<) \$8(b,_,<<) \$8(b,_,<<) \$1000000000000000000000000000000000000		r45 dd ?
## 133 ([51], [52], ## 148 dd ? ## 36 d		r46 dd ?
+) 54 (b, [53], :=) r48 dd? 55 ([40], _, [8]) auxEstado dw? 56 (, _, <) 58 (b, _, <<) 59 (d)? _raux2 dd? op2 dd? _raux3 dd? op3 dd? _raux4 dd? op4 dd? _raux5 dd? op5 dd? _raux6 dd? op6 dd? _raux7 dd? op7 dd? _raux8 dd? op8 dd? _raux9 dd? op9 dd? _raux10 dd? _raux11 dd? op10 dd? _raux12 dd? op12 dd? _raux13 dd? op13 dd? _raux15 dd? op15 dd? _raux15 dd? op15 dd? _raux16 dd? op16 dd? _raux16 dd? op16 dd? _raux16 dd? op16 dd? _raux16 dd? op16 dd? _raux18 dd? op18 dd? _raux18 dd? op18 dd? _raux18 dd? op18 dd? _raux19 dd?		r47 dd ?
S4 (b, [53], :=) 55 ([40], _, Bl) 56 (_, _, _, color of the color o		
S5 ([40], _, BI) S6 (_ , _ , _ ENDWHILE) S7 (a, _, <<) S8 (b, _, <<) S9 (b) S9		r49 dd ?
56 (, rawO dd? op0 dd? 57 (a , _ , <) 58 (b , _ , <) op1 dd? _ rawX dd? op2 dd? _ raw3 dd? op2 dd? _ raw3 dd? op3 dd? _ raw5 dd? op4 dd? _ raw5 dd? op5 dd? _ raw6 dd? op6 dd? _ raw8 dd? op8 dd? _ raw8 dd? op8 dd? _ raw10 dd? _ op9 dd? _ raw11 dd? op10 dd? _ raw11 dd? op11 dd? _ raw12 dd? op12 dd? _ raw13 dd? op13 dd? _ raw15 dd? _ op14 dd? _ raw15 dd? _ op15 dd? _ raw16 dd? _ op16 dd? _ raw17 dd? _ op16 dd? _ raw17 dd? _ op16 dd? _ raw16 dd? _ op16 dd? _ raw17 dd? _ op17 dd? _ raw18 dd? _ op17 dd? _ raw18 dd? _ op18 dd? _ raw18 dd? _ op18 dd? _ raw18 dd? _ op18 dd? _ raw19 dd?		
ENDWHILE) 57 (a , _ , <<) 58 (b , _ , <<) 59 (d ? 50		
57(a,_,<<) 58(b,_,<<) onumber of the state o		
58 (b , _ , <<) op1 dd ?		
_raux2 dd ?		op1 dd ?
op2 dd ? _raux3 dd ? op3 dd ? _raux4 dd ? op4 dd ? _raux5 dd ? op5 dd ? _raux6 dd ? op6 dd ? _raux7 dd ? op7 dd ? _raux8 dd ? op8 dd ? _raux9 dd ? op8 dd ? _raux10 dd ? op1 dd ? _raux11 dd ? op11 dd ? _raux12 dd ? op13 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op13 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op15 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op17 dd ? _raux18 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?	, , _ ,	
_raux3 dd ?		
op3 dd ?		
op4 dd ? _raux5 dd ? op5 dd ? _raux6 dd ? op6 dd ? _raux7 dd ? op7 dd ? _raux8 dd ? op8 dd ? _raux9 dd ? op9 dd ? _raux10 dd ? op10 dd ? _raux11 dd ? op12 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op15 dd ? _raux16 dd ? op15 dd ? _raux16 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux18 dd ? op18 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		
op4 dd ? _raux5 dd ? op5 dd ? _raux6 dd ? op6 dd ? _raux7 dd ? op7 dd ? _raux8 dd ? op8 dd ? _raux9 dd ? op9 dd ? _raux10 dd ? op10 dd ? _raux11 dd ? op12 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op15 dd ? _raux16 dd ? op15 dd ? _raux16 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux18 dd ? op18 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		raux4 dd ?
_raux5 dd ?		op4 dd ?
op5 dd ? _raux6 dd ? op6 dd ? _raux7 dd ? op7 dd ? _raux8 dd ? op8 dd ? _raux9 dd ? op9 dd ? _raux10 dd ? op10 dd ? _raux11 dd ? op11 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux16 dd ? op15 dd ? _raux17 dd ? op15 dd ? _raux18 dd ? op16 dd ? _raux18 dd ? op17 dd ? _raux18 dd ? op17 dd ? _raux18 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		_raux5 dd ?
op6 dd ? _raux7 dd ? op7 dd ? _raux8 dd ? op8 dd ? _raux9 dd ? op9 dd ? _raux10 dd ? op10 dd ? _raux11 dd ? op11 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op14 dd ? _raux16 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		
_raux7 dd ?		_raux6 dd ?
op7 dd ? _raux8 dd ? op8 dd ? _raux9 dd ? op9 dd ? _raux10 dd ? op10 dd ? _raux11 dd ? op11 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op16 dd ? _raux18 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op6 dd?
_raux8 dd ?		_raux7 dd ?
op8 dd ? _raux9 dd ? op9 dd ? _raux10 dd ? op10 dd ? _raux11 dd ? op11 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op7 dd ?
_raux9 dd ?		_raux8 dd ?
op9 dd ? _raux10 dd ? op10 dd ? _raux11 dd ? op11 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op8 dd ?
_raux10 dd ?		_raux9 dd ?
op10 dd ? _raux11 dd ? op11 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op9 dd ?
_raux11 dd ?		_raux10 dd ?
op11 dd ? _raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op10 dd ?
_raux12 dd ? op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		_raux11 dd ?
op12 dd ? _raux13 dd ? op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op11 dd ?
_raux13 dd ?		_raux12 dd ?
op13 dd ? _raux14 dd ? op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op12 dd?
_raux14 dd ?		_raux13 dd ?
op14 dd ? _raux15 dd ? op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op13 dd ?
_raux15 dd ?		_raux14 dd ?
op15 dd ? _raux16 dd ? op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op14 dd?
_raux16 dd ?		_raux15 dd ?
op16 dd ? _raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		op15 dd ?
_raux17 dd ? op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		_raux16 dd ?
op17 dd ? _raux18 dd ? op18 dd ? _raux19 dd ?		
_raux18 dd ? op18 dd ? _raux19 dd ?		
op18 dd ? _raux19 dd ?		op17 dd ?
_raux19 dd ?		
op19 dd ?		_raux19 dd ?
		op19 dd ?

Prof.: Mara Capuya, Hernán Villarreal

Ejemplos Varios

TP Año revisión 2018 .**1**

_raux20 dd ?
op20 dd ?
_raux21 dd ?
op21 dd?
_raux22 dd ?
op22 dd ?
_raux23 dd ?
op23 dd ?
raux24 dd ?
op24 dd ?
_raux25 dd ?
op25 dd ?
_raux26 dd ?
op26 dd ?
_raux27 dd ?
op27 dd ?
_raux28 dd ?
op28 dd ?
_raux29 dd ?
op29 dd ?
_raux30 dd ?
op30 dd ?
_raux31 dd ?
op31 dd ?
_raux32 dd ?
op32 dd ?
_raux33 dd ?
op33 dd ?
_raux34 dd ?
op34 dd ?
_raux35 dd ?
op35 dd?
_raux36 dd ?
op36 dd?
_raux37 dd ?
op37 dd ?
_raux38 dd ?
op38 dd ?
_raux39 dd ?
op39 dd ?
_raux40 dd ?
op40 dd ?
_raux41 dd ?
op41 dd ?
_raux42 dd ?
op42 dd ?
_raux43 dd ?
op43 dd ?
_raux44 dd ?
op44 dd ?

Ejemplos Varios

TP Compilador

revisión 2018 .1

raux45 dd? op45 dd? _raux46 dd? op46 dd? raux47 dd? op47 dd? raux48 dd? op48 dd? raux49 dd? op49 dd? _raux_cad1_ db 32 dup(?),'\$' _raux_cad2_ db 32 dup(?),'\$' _raux_cad3_ db 32 dup(?),'\$' aux db 32 dup(?),'\$' rS0 db 32 dup(?),'\$' rS1 db 32 dup(?),'\$' rS2 db 32 dup(?),'\$' rS3 db 32 dup(?),'\$' rS4 db 32 dup(?),'\$' rS5 db 32 dup(?),'\$' rS6 db 32 dup(?),'\$' rS7 db 32 dup(?),'\$' rS8 db 32 dup(?),'\$' rS9 db 32 dup(?),'\$' rS10 db 32 dup(?),'\$' rS11 db 32 dup(?),'\$' rS12 db 32 dup(?),'\$' rS13 db 32 dup(?),'\$' rS14 db 32 dup(?),'\$' rS15 db 32 dup(?),'\$' rS16 db 32 dup(?),'\$' rS17 db 32 dup(?),'\$' rS18 db 32 dup(?),'\$' rS19 db 32 dup(?),'\$' rS20 db 32 dup(?),'\$' rS21 db 32 dup(?),'\$' rS22 db 32 dup(?),'\$' rS23 db 32 dup(?),'\$' rS24 db 32 dup(?),'\$' rS25 db 32 dup(?),'\$' rS26 db 32 dup(?),'\$' rS27 db 32 dup(?),'\$' rS28 db 32 dup(?),'\$' rS29 db 32 dup(?),'\$' rS30 db 32 dup(?),'\$' rS31 db 32 dup(?),'\$' rS32 db 32 dup(?),'\$' rS33 db 32 dup(?),'\$'

rS34 db 32 dup(?),'\$' rS35 db 32 dup(?),'\$'

Prof.: Mara Capuya, Hernán Villarreal

TP Compilador Año revisión 2018 .1

```
rS36 db 32 dup(?),'$'
rS37 db 32 dup(?),'$'
rS38 db 32 dup(?),'$'
rS39 db 32 dup(?),'$'
rS40 db 32 dup(?),'$'
rS41 db 32 dup(?),'$'
rS42 db 32 dup(?),'$'
rS43 db 32 dup(?),'$'
rS44 db 32 dup(?),'$'
rS45 db 32 dup(?),'$'
rS46 db 32 dup(?),'$'
rS47 db 32 dup(?),'$'
rS48 db 32 dup(?),'$'
rS49 db 32 dup(?),'$'
.CODE
JMP MAIN
; devuelve en BX la cantidad de caracteres que tiene un string
; DS:SI apunta al string.
STRLEN PROC
mov bx,0
STRL01:
cmp BYTE PTR [SI+BX],'$'
je STREND
inc BX
jmp STRL01
STREND:
STRLEN ENDP
; copia DS:SI a ES:DI; busca la cantidad de caracteres
COPIAR PROC
call STRLEN; busco la cantidad de caracteres
cmp bx,MAXTEXTSIZE
jle COPIARSIZEOK
mov bx,MAXTEXTSIZE
COPIARSIZEOK:
mov cx,bx ; la copia se hace de X
cld; cld es para que la copia se realice
; hacia adelante
rep movsb; copia la cadea
mov al,'$'; car@ter terminador
mov BYTE PTR [DI], al; el registro DI quedo apuntando al
; final
ret
COPIAR ENDP
```

TP Compilador Año revisión 2018 .1

Prof.: Mara Capuya, Hernán Villarreal

```
***************
; concatena DS:SI al final de ES:DI.
; busco el size del primer string
; sumo el size del segundo string
; si la suma excede MAXTEXTSIZE, copio solamente MAXTEXTSIZE caracteres
; si la suma NO excede MAXTEXTSIZE, copio el total de caracteres que tiene el
segundo string
CONCAT PROC
push ds
push si
call STRLEN; busco la cantidad de caracteres del
;2do string
mov dx,bx; guardo en DX la cantidad de caracteres
;en el origen.
mov si,di
push es
pop ds
call STRLEN; tama#o del 1er string
add di,bx; DI ya queda apuntando al final del
;primer string
add bx,dx; tama#o total
cmp bx,MAXTEXTSIZE; excede el tama#o maximo?
jg CONCATSIZEMAL
CONCATSIZEOK: ; La suma no excede el maximo, copio
; todos
mov cx,dx; los caracteres del segundo string.
imp CONCATSIGO
CONCATSIZEMAL: ; La suma de caracteres de los 2 strings
; exceden el maximo
sub bx,MAXTEXTSIZE
sub dx,bx
mov cx,dx; copio lo maximo permitido el resto
;se pierde.
CONCATSIGO:
push ds
pop es
pop si
pop ds
cld; cld es para que la copia se realice
;hacia adelante
rep movsb; copia la cadea
mov al,'$'; car@ter terminador
mov BYTE PTR [DI], al; el registro DI quedo apuntando al
;final
ret
CONCAT ENDP
```

;******FIN****FUNCIONES*********

TP Compilador Año revisión 2018 .1

Prof.: Mara Capuya, Hernán Villarreal

Aytes: Daniel Carrizo – Facundo Velazquez Santillán

Co

```
.********************
CARGAR PROC
GOBAK1:
MOV AH,01H
INT 21H
CMP AL, ODH
JE BAK1
MOV [SI],AL
INC SI
JMP GOBAK1
BAK1:
MOV AL,'$'
MOV [SI],AL
ret
CARGAR ENDP
; Parametros
; ax: valor
; bx: donde guardar la cadena final
; Retorna
; cadena
itoa proc near
xor cx, cx ; CX = 0
itoa 1:
cmp ax,0 ; El ciclo itoa 1 extrae los digitos del
je itoa_2; menos al mas significativo de AX y los
; guarda en el stack. Al finalizar el
xor dx,dx ; ciclo el digito mas significativo esta
push bx ; arriba del stack.
mov bx,10; CX contiene el numero de digitos
div bx
pop bx
push dx
inc cx
jmp itoa_1
itoa_2:
cmp cx,0 ; Esta seccion maneja el caso cuando
ja itoa_3 ; el numero a convertir (AX) es 0.
mov ax,'0'; En este caso, el ciclo anterior
mov [bx],ax; no guarda valores en el stack y
inc bx ; CX tiene el valor 0
jmp itoa 4
itoa_3:
pop ax ; Extraemos los numero del stack
add ax,30h; lo pasamos a su valor ascii
```

TP Compilador Año revisión 2018 .1

Prof.: Mara Capuya, Hernán Villarreal

```
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
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.
         ;byte a word
 cbw
 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
imp atoi 1 ;seguir mientras SI apunte a un numero ascii
 noascii:
        ;BX tiene el valor final
 ret
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
```

Prof.: Mara Capuya, Hernán Villarreal

TP

revisión 2018 .1

Compilador

MOV r0,1 FILD r0 FIST b FFREE st(0) ;FIN ASIGNACION WHILEO: ;INICIO WHILEO 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 WHILEO MOV ax,auxEstado **FWAIT** SAHF JB FIN_WHILEO ;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

Prof.: Mara Capuya, Hernán Villarreal

TP Compilador Año revisión 2018 .1

FILD r4 FILD r5 **FCOMP** FSTSW ax **FWAIT** SAHF FFREE st(0) JB FIN_IFO ;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 FIO: 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

Prof.: Mara Capuya, Hernán Villarreal

TP

revisión 2018 .1

Compilador

;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 WHILEO FIN_WHILEO: ;FIN_WHILEO 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, aux Estado

FWAIT

TP Compilador Año revisión 2018 .1

Prof.: Mara Capuya, Hernán Villarreal

Aytes: Daniel Carrizo – Facundo Velazquez Santillán

```
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

LENGUAJES Y COMPILADORES - UNLAM

Prof.: Mara Capuya, Hernán Villarreal

Aytes: Daniel Carrizo – Facundo Velazquez Santillán

Ejemplos Varios

TP Compilador Año revisión 2018 .1

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.