

PROYECTO 2

ARQUITECTURA DE COMPUTADORES Y
ENSAMBLADORES 1

MANUAL TÉCNICO

RODRIGO ALEJANDRO HERNÁNDEZ DE LEÓN

CARNET:201900042

I. Introducción

Objetivo

Otorgar el apoyo al programador con la comprensión del programa hecho en lenguaje ensamblador con compilador MASM y DOSBox para la ejecución de pruebas para la calculadora.

II. Especificación técnica

- Computadora portátil o de escritorio.
- Sistema Operativo Windows o alguna distribución de Linux
- DOSBox
- Compilador MASM
- Visual Studio Code

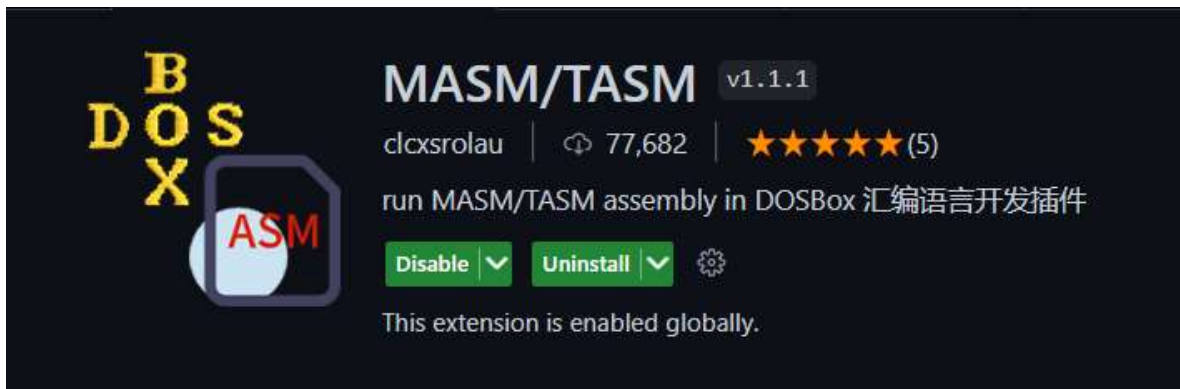
III. Lógica del Programa

En las siguientes paginas se estará explicando el código desarrollado de la calculadora:

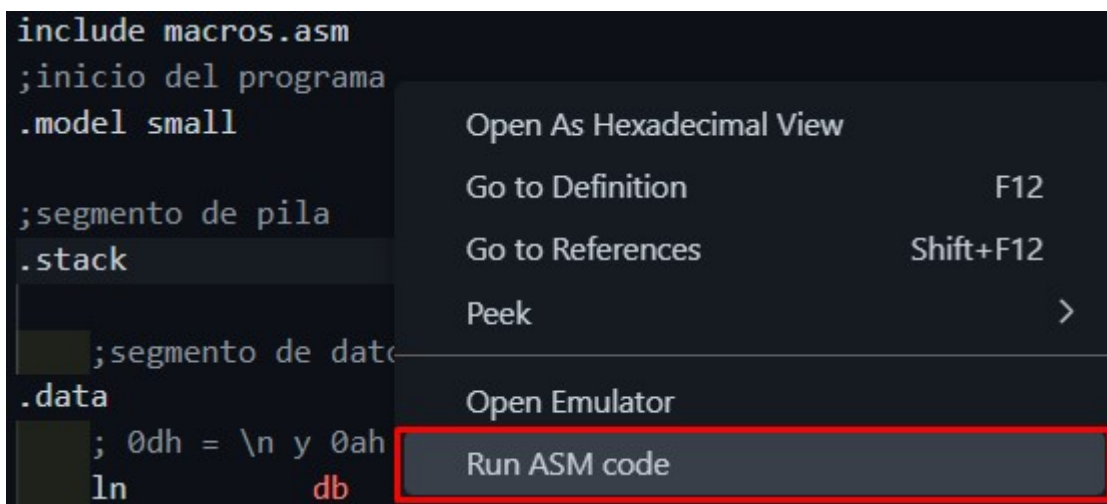
1. Ingreso a la aplicación por Visual Studio Code
2. Segmento Data
3. Segmento Code
4. Macros

1. Ingreso a la aplicación por VS Code

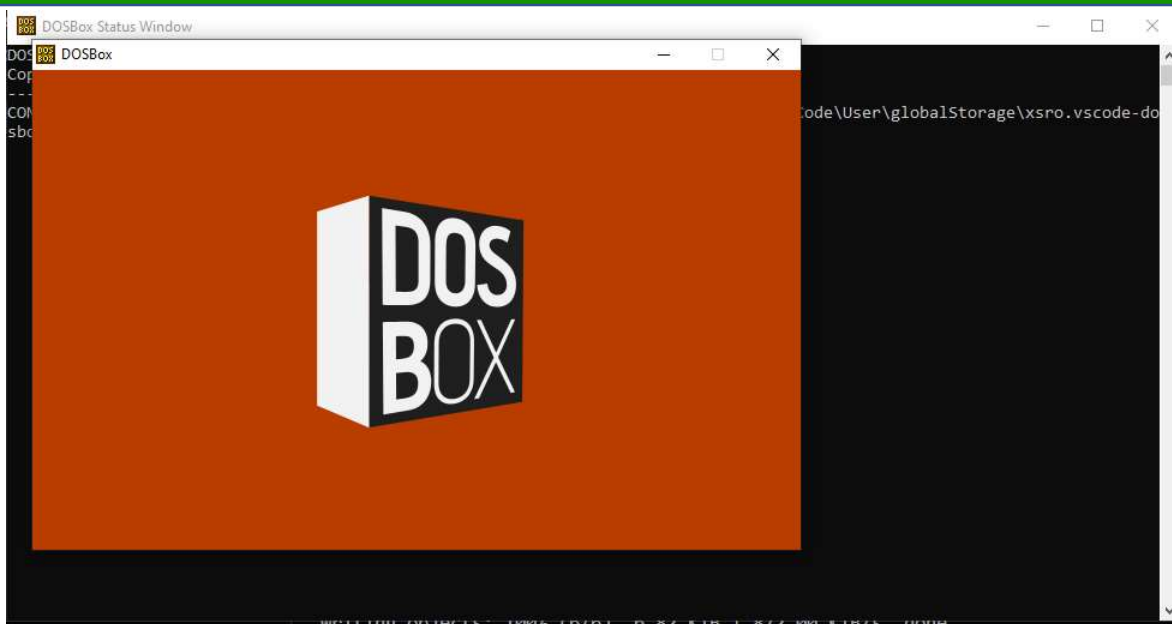
Tiene que tener instalada la siguiente extensión:



Luego hay que ubicarse en el archivo de *main.asm* y presionar click derecho, después seleccionar la opción de Run ASM:



Y posteriormente abrirá el DOSBox.



Y aparecerá el menú principal:



2. Segmento data

Se declararon las siguientes variables que se utilizaron en el programa:

```
;METODO DE NEWTON Y STEFFENSEN
msgNewton      db      '=====METODO DE NEWTON=====','$'
msgSteffensen  db      '=====METODO DE STEFFENSEN=====','$'
msgIteraciones db      'Ingrese el numero de iteraciones que desea realizar: ','$'
itn            db      0
its           db      0
msgCoefTolerancia db    'Ingrese el coeficiente de tolerancia: ','$'
toln          db      0
tols          db      0
msgGradoTolerancia db   'Ingrese el grado de tolerancia: ','$'
gradon        db      0
trunc         dw      0CF3h
grados        db      0
msgLimiteSuperior db    'Ingrese el limite superior: ','$'
limsn         db      0
limss         db      0
msgLimiteInferior db    'Ingrese el limite inferior: ','$'
limin         db      0
limis         db      0
res           db      0
tmp           db      0
dectemp       dq      0.0
contnew       db      0
msgxn         db      'Xn = ','$'
msgxn1        db      'Xn+1 = ','$'
tmp1          db      0
xnsig         dq      0.0
dos           dq      2.0
n1            dw      0
n2            dw      0
contador      db      0
xn            dq      0.0
entero        dw      0
diez          dq      10.0
cont          db      0
ciclotemp     dq      0.0
contador1     db      0
tmp2          db      0
tmp3          db      0
tmp4          db      0
tmp5          db      0
tmp6          db      0
sumatemp      dw      0
sumatemp2     dq      0.0
x0n           dq      0.0
coef_tmp      dw      0
coef_tmp2     dq      0.0
val5          dq      0.0
val4          dq      0.0
val3          dq      0.0
val2          dq      0.0
val1          dq      0.0
val0          dq      0.0
vald4         dq      0.0
vald3         dq      0.0
vald2         dq      0.0
vald1         dq      0.0
vald0         dq      0.0
fx            dq      0.0
fdx           dq      0.0
msgiteracion  db      'ITERACION #','$'
numiteracion  db      1
```

3. Segmento code

INTRO:

Este apartado se encuentra la información inicial.

A screenshot of a code editor window with a dark background and light-colored text. The window has three colored window control buttons (red, yellow, green) in the top-left corner. The code is as follows:

```
INTRO:
    cls
    print inicio
    print ln
    print inicio1
    print ln
    print inicio2
    print ln
    print inicio3
    print ln
    print inicio4
    print ln
    print inicio5
    print ln
    print inicio6
    print ln
    print inicio7
    print ln
    print inicio8
    print ln
    print inicio09
    print ln
    print inicio010
    print ln
    pausa
    jmp MENU
```

MENU:

Se encuentra para imprimir el menú principal.



MENU:

```
cls
print inicio9
print ln
print inicio10
print ln
print inicio11
print ln
print inicio12
print ln
print inicio13
print ln
print inicio14
print ln
print inicio15
print ln
print inicio16
print ln
print inicio17
print ln
pausa
cmp al, 49
je OPCION1
cmp al, 50
je OPCION2
cmp al, 51
je OPCION3
cmp al, 52
je OPCION4
cmp al, 53
je OPCION5
cmp al, 54
je OPCION6
cmp al, 55
je OPCION7
cmp al, 27
je SALIR
cmp al, 56
je SALIR
print ln
jmp OPCIONERROR
```

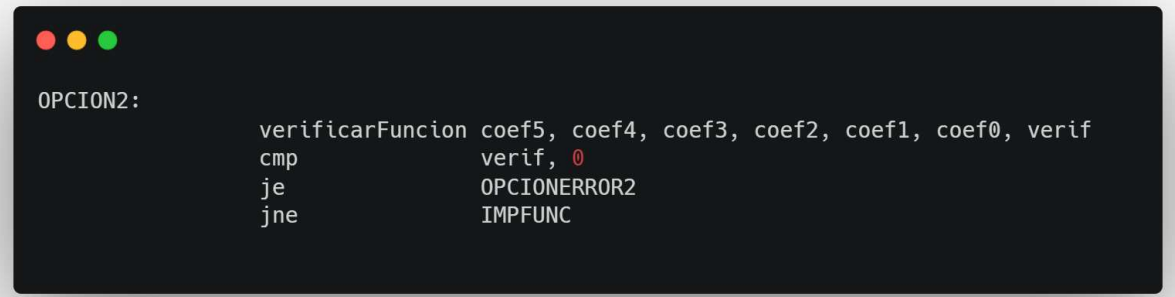
En este apartado se encuentra la lógica del ingreso de coeficientes de la función.

En este apartado se encuentra la lógica del ingreso de coeficientes de la función.

[illegible]

OPCION2:

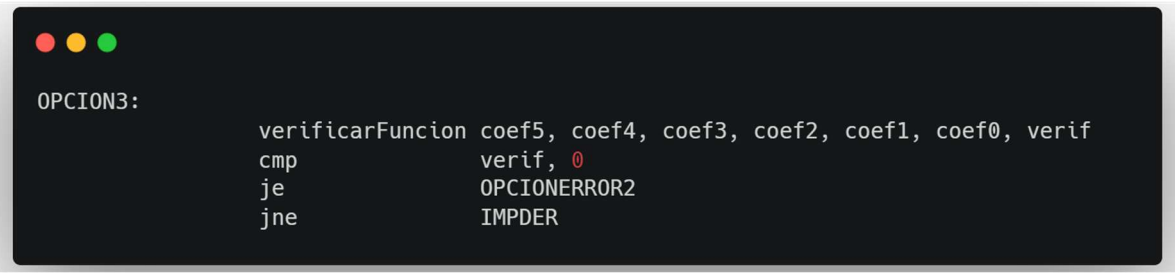
En este apartado se encuentra la lógica de imprimir la función ingresada.



```
OPCION2:
    verificarFuncion coef5, coef4, coef3, coef2, coef1, coef0, verific
    cmp             verific, 0
    je              OPCIONERROR2
    jne             IMPFUNC
```

OPCION3:

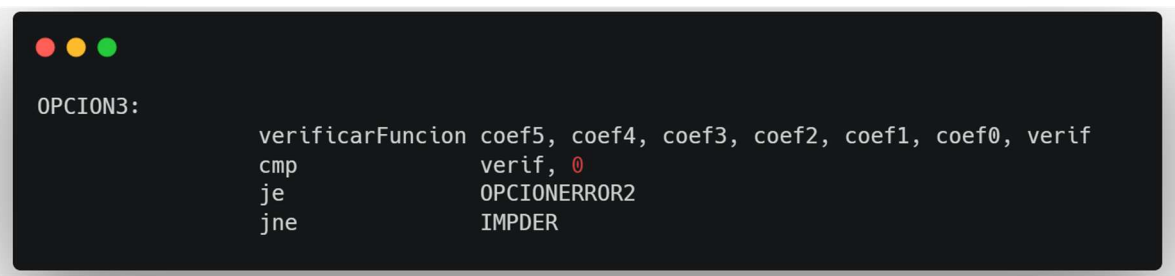
En este apartado se encuentra la lógica de imprimir la derivada de la función.



```
OPCION3:
    verificarFuncion coef5, coef4, coef3, coef2, coef1, coef0, verific
    cmp             verific, 0
    je              OPCIONERROR2
    jne             IMPDER
```

OPCION4:

En este apartado se encuentra la lógica de imprimir la integral de la función.



```
OPCION3:
    verificarFuncion coef5, coef4, coef3, coef2, coef1, coef0, verific
    cmp             verific, 0
    je              OPCIONERROR2
    jne             IMPDER
```

OPCION6:

En este apartado se encuentra la lógica del método de newton de la función.

OPCION6:

```
cls
verificarFuncion coef5, coef4, coef3, coef2, coef1, coef0, verif
cmp             verif, 0
je             OPCIONERROR2
print          msgNewton
print          ln
print          msgIteraciones
print          ln
getNumero      itn
print          ln
print          msgCoefTolerancia
print          ln
getNumero      toln
print          ln
print          msgGradoTolerancia
print          ln
getNumero      gradon
print          ln
print          msgLimiteSuperior
print          ln
getNumero      limsn
print          ln
print          msgLimiteInferior
print          ln
getNumero      limin
print          ln
mov bl, 0
mov cl, 0
mov bl, limin
mov cl, limsn
cmp bl, cl
jae ERRORNEWTON
;X0
cls
print mgsiteracion
printr numiteracion
add bl,cl
mov al,bl
mov ah,0
mov sumatemp, ax

fild sumatemp
fstp sumatemp2

fld sumatemp2
fddiv dos
fstp x0n
print ln
print msgxn
printDecimal x0n,gradon
print ln
print msgxn1
MetodoNewton x0n, xnsig
printDecimal val1, gradon
print ln
pausa
jmp             MENU
```

IMPFUNC:

En este apartado se manda a llamar la impresión de la función.

A screenshot of a terminal window with a dark background and light gray text. The window has three colored window control buttons (red, yellow, green) in the top-left corner. The text inside the terminal is as follows:

```
IMPFUNC:      cls
               printFunc      coef5, coef4, coef3, coef2, coef1, coef0
               pausa
               jmp             MENU
```

IMPDER:

En este apartado se manda a llamar la impresión de la derivada.

```
IMPDER:
        cls
        printDerivada    der5, der4, der3, der2, der1
        pausa
        jmp              MENU
```

IMPINT:

En este apartado se manda a llamar la impresión de la integral.

```
IMPINT:
        cls
        printIntegral    int5, int4, int3, int2, int1, int0
        pausa
        jmp              MENU
```

4. Macros

Print

En esta macro se encarga de imprimir cadenas.

```
print macro txt
    ; carga en memoria las variables del segmento de datos
    mov ax, @data
    mov ds, ax

    ; impresion por pantalla
    mov ah, 09h
    lea dx, txt
    int 21h
endm
```

Pausa

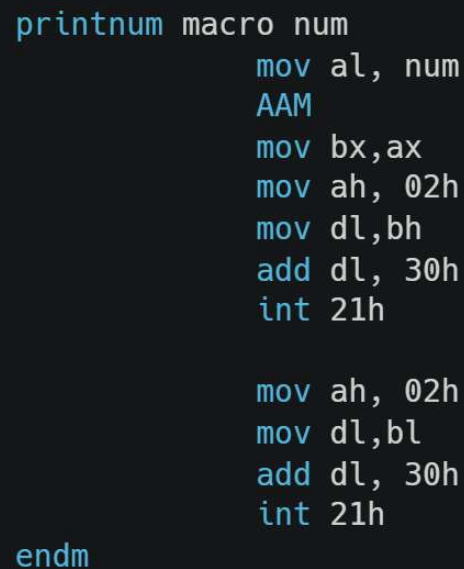
Esta macro se encarga de dar una pausa para poder leer funciones y el menú principal.



```
pausa macro
    mov ah, 01h
    int 21h
endm
```

Printnum

En esta macro se encarga de imprimir los números.



```
printnum macro num
    mov al, num
    AAM
    mov bx, ax
    mov ah, 02h
    mov dl, bh
    add dl, 30h
    int 21h

    mov ah, 02h
    mov dl, bl
    add dl, 30h
    int 21h
endm
```

getNumero

Esta macro se encarga de obtener los números ingresados

```

getNumero macro var
    LOCAL          n1,n2,n3,n2n,negativo,negativo1,salir
    limpiarNumero numtextaux
    mov             ah, 0ah
    lea             dx, textaux
    int             21h
    cmp             longtextaux,1
    je              n1
    cmp             longtextaux,2
    je              n2
    ; VERIFICA QUE EL NUMERO INGRESADO SEA DE 1 DIGITO (POSITIVO)
n1:
    mov             al,numtextaux
    sub             al, 30h                      ;48
    mov             var, al
    jmp             salir
    ; VERIFICA QUE EL NUMERO INGRESADO SEA DE 2 DIGITOS (POSITIVO) O 1 DIGITO (NEGATIVO)
n2:
    mov             unidades,0
    mov             decenas,0

    mov             al, numtextaux[0]
    sub             al, 30h
    mov             decenas, al

    mov             al, numtextaux[1]
    sub             al, 30h
    mov             unidades, al

    mov             al, decenas
    mov             bl, 10
    mul             bl
    add             al, unidades
    mov             var, al

    salir:
endm

```

limpiarNumero

Esta macro se encarga de limpiar la entrada de numeros.

```
limpiarNumero macro text
    LOCAL repetir
    xor     bx, bx
    mov     cx, lengthof text
repetir:
    mov     text[bx], '$'
    inc     bx
    loop    repetir
endm
```

Multiplicar

Esta macro se encarga de multiplicar 2 números positivos.

```
multiplicar macro coefi, expo, total
    mov     al, coefi
    mov     bl, expo
    mul     bl
    mov     total, al
    mov     ax, 0000h
    mov     al, bl
endm
```

Dividir

Macro para dividir 2 números

```
dividir macro coefi, expo, total
    mov     al, coefi
    mov     bl, expo
    div     bl
    mov     total, al
    mov     ax, 0000h
    mov     al, bl
endm
```

printFunc

Macro para imprimir la función ingresada.

```
printFunc macro c5, c4, c3, c2, c1, c0
    LOCAL C0F5,C0F4,C0F3,C0F2,C0F1,C0F0,PR5,PR4,PR3,PR2,PR1,PR0,salir
    print msg
    cmp c5, 0
    je C0F4
    jne C0F5
C0F5:
    cmp c5, 0
    je C0F4
    jne PR5
PR5:
    printn c5
    print x5
    print mas
C0F4:
    cmp c4, 0
    je C0F3
    jne PR4
PR4:
    printn c4
    print x4
    print mas
C0F3:
    cmp c3, 0
    je C0F2
    jne PR3
PR3:
    printn c3
    print x3
    print mas
C0F2:
    cmp c2, 0
    je C0F1
    jne PR2
PR2:
    printn c2
    print x2
    print mas
C0F1:
    cmp c1, 0
    je C0F0
    jne PR1
PR1:
    printn c1
    print x1
    print mas
C0F0:
    printn c0
salir:
    print ln
    print inicio09
    print ln
endm
```


printDerivada

Macro para imprimir la derivada.

```
printDerivada macro c4, c3, c2, c1, c0
    LOCAL C0F4,C0F3,C0F2,C0F1,C0F0,PR4,PR3,PR2,PR1,PR0,salir
    print msgDerivada
    print ln
    cmp c4, 0
    je C0F3
    jne C0F4
C0F4:
    cmp c4, 0
    je C0F3
    jne PR4
PR4:
    printn c4
    print x4
    print mas
C0F3:
    cmp c3, 0
    je C0F2
    jne PR3
PR3:
    printn c3
    print x3
    print mas
C0F2:
    cmp c2, 0
    je C0F1
    jne PR2
PR2:
    printn c2
    print x2
    print mas
C0F1:
    cmp c1, 0
    je C0F0
    jne PR1
PR1:
    printn c1
    print x1
    print mas
C0F0:
    printn c0
salir:
    print ln
    print inicio09
    print ln
endm
```

printIntegral

Este macro se encarga de imprimir la integral.

```

printIntegral macro c5, c4, c3, c2, c1, c0
    LOCAL
    COF5,COF4,COF3,COF2,COF1,COF0,PR5,PR4,PR3,PR2,PR1,PR0,salir
    print msgIntegral
    print ln
    cmp c5, 0
    je COF4
    jne COF5
    COF5:
        cmp c5, 0
        je COF4
        jne PR5
    PR5:
        printn c5
        print x6
        print mas
    COF4:
        cmp c4, 0
        je COF3
        jne PR4
    PR4:
        printn c4
        print x5
        print mas
    COF3:
        cmp c3, 0
        je COF2
        jne PR3
    PR3:
        printn c3
        print x4
        print mas
    COF2:
        cmp c2, 0
        je COF1
        jne PR2
    PR2:
        printn c2
        print x3
        print mas
    COF1:
        cmp c1, 0
        je COF0
        jne PR1
    PR1:
        printn c1
        print x2
        print mas
    COF0:
        printn c0
        print x1
        print mas
    salir:
        print constant
        print ln
        print inicio09
        print ln
endm

```

VerificarFuncion

Este macro verifica si se ingreso datos a la función.

```
verificarFuncion macro c5,c4,c3,c2,c1,c0, verificador
    LOCAL
    C0F5,C0F4,C0F3,C0F2,C0F1,C0F0,PR5,PR4,PR3,PR2,PR1,PR0,salir
    mov     verificador, 0
    cmp     c5 , 0
    je      C0F4
    jne     C0F5
C0F5:
    cmp     c5 , 0
    je      C0F4
    jne     PR5
PR5:
    mov     verificador, 1
    jmp     salir
C0F4:
    cmp     c4, 0
    je      C0F3
    jne     PR4
PR4:
    mov     verificador, 1
    jmp     salir
C0F3:
    cmp     c3, 0
    je      C0F2
    jne     PR3
PR3:
    mov     verificador, 1
    jmp     salir
C0F2:
    cmp     c2, 0
    je      C0F1
    jne     PR2
PR2:
    mov     verificador, 1
    jmp     salir
C0F1:
    cmp     c1, 0
    je      C0F0
    jne     PR1
PR1:
    mov     verificador, 1
    jmp     salir
C0F0:
    cmp     c0, 0
    je      salir
    jne     PR0
PR0:
    mov     verificador, 1
salir:
    mov     verificador, 1
endm
```

printNumero2

Esta macro imprime números de 16 bits

```
printnumero2 macro numeroprint
    LOCAL label1, print1, exit
    MOV AX,@DATA
    MOV DS,AX
    mov ax, numeroprint
    mov cx,0
    mov dx,0
    label1:
    cmp ax,0
    je print1
    mov bx,10
    div bx
    push dx
    inc cx
    xor dx,dx
    jmp label1
    print1:
        cmp cx,0
        je exit
        pop dx
        add dx,48
        mov ah,02h
        int 21h
        dec cx
        jmp print1
    exit:
endm
```

metodoFx

Macro para calcular $f(x)$

```
metodoFx macro num1,num2
;X5
mov bx,0
mov bl, coef5
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul num1
fmul num1
fmul num1
fmul num1
fmul coef_tmp2
fstp val5
;X4
mov bx,0
mov bl, coef4
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul num1
fmul num1
fmul num1
fmul coef_tmp2
fstp val4
;X3
mov bx,0
mov bl, coef3
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul num1
fmul num1
fmul coef_tmp2
fstp val3
;X2
mov bx,0
mov bl, coef2
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul num1
fmul coef_tmp2
fstp val2
;X1
mov bx,0
mov bl, coef2
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul coef_tmp2
fstp val1
;X0
mov bx,0
mov bl, coef0
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp val0
;SUMA DE TODO
fld val5
fadd val4
fadd val3
fadd val2
fadd val1
fadd val0
fstp num2
endm
```

metodoDfxMacro para calcular $f'(x)$

```
metodoDfx macro num1,num2
;X4
mov bx,0
mov bl, der5
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul num1
fmul num1
fmul num1
fmul coef_tmp2
fstp vald4
;X3
mov bx,0
mov bl, der4
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul num1
fmul num1
fmul coef_tmp2
fstp vald3
;X2
mov bx,0
mov bl, der3
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul num1
fmul coef_tmp2
fstp vald2
;X1
mov bx,0
mov bl, der2
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp coef_tmp2
fld num1
fmul coef_tmp2
fstp vald1
;X0
mov bx,0
mov bl, der1
mov bh, 0
mov coef_tmp, bx
fild coef_tmp
fstp vald0
;SUMA DE TODO
fld vald4
fadd vald3
fadd vald2
fadd vald1
fadd vald0
fstp num2
endm
```

MetodoNewton

Macro para hacer el método de newton

```
MetodoNewton macro inicio, siguiente
    metodoFx inicio, fx
    metodoDFx inicio, fdx
    fld fx
    fdiv fdx
    fsub inicio
    fstp siguiente
endm
```

printDecimal

Macro para imprimir los decimales

```
printDecimal macro numerodec, gradodec
    LOCAL ciclo, salir
    fld numerodec
    frndint
    fistp entero
    printnumero2 entero
    print punto
    mov cont, 0
    fld numerodec
    fstp dectemp
    jmp ciclo
ciclo:
    mov al, cont
    cmp al, gradodec
    je salir
    ;RESTA EL ENTERO CON LOS DECIMALES
    fld dectemp
    fisub entero
    fstp dectemp
    ;MULTIPLICA CON 10
    fld dectemp
    fmul diez
    fst dectemp
    ;truca decimales
    frndint
    fistp entero
    ;imprime el decimal
    printnumero2 entero
    inc cont
    jmp ciclo
salir:
endm
```