Universidad de San Carlos de Guatemala
Facultad de ingeniería
Escuela de Ciencias y Sistemas
Arquitectura de Computadores y Ensambladores
Segundo semestre 2022

Manual Técnico

Jhonatan Josué Tzunun Yax

Introducción

En el siguiente manual se describe a detalle los procedimientos y macros, así como el contenido de cada una y porque fue necesaria la creación de esta, para que se usó cada procedimiento, y macro, explicando de forma breve la lógica aplicada en este proyecto.

Listado de Procedimientos

- **posCursor**: Posiciona el cursor al inicio de la pantalla
- **limpiar**: Limpia la pantalla
- **integrar**: Realiza el proceso de integrar la función almacenada, divide el coeficiente con el exponente de grado.
- **Integral**: Muestra en pantalla el resultado de la integral. Comprueba que el coeficiente no esté vacío para mostrarlo en pantalla.
- **Derivar**: Realiza el proceso de derivar la función almacenada, multiplica el coeficiente con el exponente de grado.
- **Derivada**: Muestra en pantalla el resultado de la derivada. Comprueba que el coeficiente no esté vacío para mostrarlo en pantalla.
- **Mostrar_suma**: Muestra el signo de suma en pantalla.
- **printEqL5**: Muestra el grado 5 de la ecuación y comprueba si existe otro grado para llamar al procedimiento mostrar_suma.
- **printEqL4**: Muestra el grado 4 de la ecuación y comprueba si existe otro grado para llamar al procedimiento mostrar_suma.
- **printEqL3**: Muestra el grado 3 de la ecuación y comprueba si existe otro grado para llamar al procedimiento mostrar_suma.
- **printEqL2**: Muestra el grado 2 de la ecuación y comprueba si existe otro grado para llamar al procedimiento mostrar_suma.
- **printEqL1**: Muestra el grado 1 de la ecuación y comprueba si existe otro grado para llamar al procedimiento mostrar_suma y muestra la constante, si existiese.
- **resetCof**: Reinicia los coeficientes de la función, derivada e integral.
- **unirNums**: Une la decena con la unidad para formar un número, por medio de multiplicación.
- **showTwoNums**: Muestra dos números que estén en los registros bh y bl.
- **capTwoNums**: Permite capturar dos números de entrada.
- **menu_ecuacion**: Muestra el menú de elegir grado y muestra los mensajes de ingresar coeficientes.
- **pressAnyKey**: Muestra el mensaje de “presione una tecla para continuar” y espera cualquier carácter.
- **Main**: Muestra el menú principal y redirige a los demás procedimientos dependiendo de la opción elegida.

Segmento de datos:

- Msg variables: contienen el menú principal
- Salido: Muestra el mensaje de pressAnyKey
- Opc variables: Contienen el nombre de la opción elegida.
- Sub_msg variables: Muestran el submenú de elección de grados.
- Cof_msg variables: Contienen el mensaje de numero de coeficiente a ingresar.
- C variables: Guardan los coeficientes ingresados para la ecuación.
- Cd variables: Guardan los coeficientes calculados para la derivada.
- Ci variables: Guardan los coeficientes calculados para la integral.

- L variables: Contienen las literales de grado.

Macros

- dividirCof: Realiza el proceso de división de el coeficiente y el exponente de grado. Recibe tres parámetros que son: divisor, dividendo y variable donde se guardará.

FASE 2

Variables

Menús que se mostraran en pantalla, dependiendo de la elección del usuario.

```
;Mensaje de ingreso de parametros metodo de newton
   met_msg db 13, 10, 'Aproximacion inicial: $'
   met_msg1 db 13, 10, 'Numero de iteraciones maximo: $'
   met_msg2 db 13, 10, 'Coeficiente de tolerancia: $'
   met_msg3 db 13, 10, 'Grado de tolerancia: $'
   met_msg4 db 13, 10, 'Limite superior del metodo: $'
   met_msg5 db 13, 10, 'Limite inferior: $'
   met_msgError db 13, 10, 'El limite inferior debe ser menor al limite superior$'

   ;Mensaje de eleccion de grafica
   g_msg  db 13, 10, '1. Graficar funcion original'
   g_msg1 db 13, 10, '2. Graficar derivada'
   g_msg2 db 13, 10, '3. Graficar integral'
   g_msg3 db 13, 10, '4. Salir'
   g_msg4 db 13, 10, '>>> Opc: $'

   ;Mensaje de grafica elegida
   gf_msg  db 13, 10, ' ** Funcion original ** $'
   gf_msg1 db 13, 10, ' ** Derivada ** $'
   gf_msg2 db 13, 10, ' ** Integral ** $'
```

Variables para la creación de la gráfica y método de newton.

Las variables cg se utilizaron como coeficientes auxiliares para la gráfica, se reemplazaron dependiendo de la elección del usuario, con los coeficientes de la función original, integral o derivada.

Las variables res y f_x se utilizaron para realizar las operaciones en la obtención de f(x)

Las variables pt se utilizaron para guardar la entrada de petición de rango para graficar la función.

```
;Coeficientes para la grafica
   cg5 db 00
   cg4 db 00
   cg3 db 00
   cg2 db 00
   cg1 db 00
   cg0 db 00

   ;Variables newton
   n_xf db 00
   n_xi db 00

   ;Variables resultado funcion
   res_func   dw 00
   res_funcaux dw 00
   res_pot    db 00
   f_x       db 0
   f_xaux     db 0
   aux1 dw 100
   aux2 dw 1
   aux3 dw 2
   ;f_y       db 00


   ; Variables para graficar
   pt_ejeXP db 00 ; Eje x positivo
   pt_ejeXN db 00 ; Eje x negativo
   flag_vacio db 00 ;Bandera para saber si no hay una funcion
   salidoMenuG db 00 ; Bandera para salir del menu
```

Procedimientos

Muestra el menú de opciones para generar la gráfica de la función y mueve los valores de los coeficientes de la opción elegida a los coeficientes auxiliares asignados para la gráfica.

```
OpcGrafica proc

   mov salidoMenuG, 0
   ;Resetear coeficientes
   mov cg0, 00
   mov cg1, 00
   mov cg2, 00
   mov cg3, 00
   mov cg4, 00
   mov cg5, 00

   opcGf0:
      call limpiar
      call posCursor

      Print g_msg
      ;------------ Pido la opcion ------------
      mov ah,01h
      int 21h
      ;sub al,48d    ; ajusto para que aparezca el numero
      ;----------------------------------------

      ;Comparo las opciones para elejir la siguiente operacion a realizar
      cmp al, 31h
      je opcGf1
      cmp al, 32h
      je opcGf2
      cmp al, 33h
      je opcGf3
      cmp al, 34h
      je salidoOpcgAux
   jmp opcGf0

   continua...
OpcGrafica endp
```

Obtiene f(x) realizando la operación de cada coeficiente y su variable por separado, al finalizar con un coeficiente suma el resultado con el anterior, si hubiese.

Ejemplo: 3x^2 => 3*(2) ^2

```
GetY proc
   mov res_func, 0
   cof_cinco:
      cmp cg5, 00
      je cof_cuatro

      ;Reseteo las variables
      mov varCont, 1
      mov res_funcaux, 0
      mov f_xaux, 0

      ;call ResetarPotFunc
      Potencia f_x, 5, res_pot, varCont
      multiplicarFunc cg5, res_pot, res_funcaux
      sumarFunc res_func, res_funcaux

   cof_cuatro:
      cmp cg4, 00
      je cof_tres

      ;Reseteo las variables
      mov varCont, 1
      mov res_funcaux, 0
      mov f_xaux, 0

      ;call ResetarPotFunc
      Potencia f_x, 4, res_pot, varCont
      multiplicarFunc cg4, res_pot, res_funcaux
      sumarFunc res_func, res_funcaux

   continua...
GetY endp
```

Grafica la función elegida por el usuario.

Realiza un ajuste en X y Y para simular que la posición (0,0) está en (160,100).

Utiliza la variable f_x como la entrada X y la incrementa en cada operación.

```
GraficarFuncion2 proc
   cmp pt_ejeXP, 00
   je gf_loop3
   gf_loop2:
      inc f_x

      ;***** Positivo *****
      call GetY
      xor cx, cx
      xor dx, dx
      ;Ajuste en Y
      ;sumarFunc res_func, 100
      restarFunc 100, res_func
      ;Ajuste en X, si está bien
      sumar f_xaux, 160
      sumar f_xaux, f_x
      mov dx, res_func
      mov cl, f_xaux
      PrintDot2
      ;********************
      mov cl, 0
      mov cl, pt_ejeXP
      ;cmp f_x, 4
      cmp f_x, cl
      jne gf_loop2

   continua...
GraficarFuncion2 endp
```

Comprueba si los coeficientes de la función original están vacíos, si no activa la variable de error.

```
CofVacios proc
   mov flag_vacio, 0
   cofVacio5:
      cmp c5, 00
      je cofVacio4
      jne salirCofFull

   cofVacio4:
      cmp c4, 00
      je cofVacio3
      jne salirCofFull

   cofVacio3:
      cmp c3, 00
      je cofVacio2
      jne salirCofFull

   cofVacio2:
      cmp c2, 00
      je cofVacio1
      jne salirCofFull

   cofVacio1:
      cmp c1, 00
      je cofVacio0
      jne salirCofFull

   cofVacio0:
      cmp c0, 00
      je salirCofVacios
      jne salirCofFull

   salirCofVacios:
      mov flag_vacio, 0
      jmp salidoCofV
   salirCofFull:
      mov flag_vacio, 1

   salidoCofV:
      ret
CofVacios endp
```

Muestra el menú para las opciones 6 y 7, además comprueba si el límite inferior es menor al superior

```
menu_steff_new proc
   ;Pregunta
   mostrarMsgMet:
   Print met_msg
   ;Captura dos numeros de entrada
   call capTwoNums
   ;----- Unir los numeros --------
   call unirNums
   mov p_init, al  ; muevo el resultado a p_init
   ;------------------------------
   ;Pregunta
   Print met_msg4
   ;Captura dos numeros de entrada
   call capTwoNums
   ;----- Unir los numeros --------
   call unirNums
   mov p_lsup, al  ; muevo el resultado a p_lsup
   ;------------------------------
   ;Pregunta
   Print met_msg5
   ;Captura dos numeros de entrada
   call capTwoNums
   ;----- Unir los numeros --------
   call unirNums
   mov p_linf, al  ; muevo el resultado a p_linf
   ;------------------------------

   mov al, p_linf
   cmp p_lsup, al
   jle mostrarMsgError
   jg salidoMenSN

   mostrarMsgError:
      Print met_msgError
      call pressAnyKey
      call limpiar
      call posCursor
      jmp mostrarMsgMet

   salidoMenSN:
   ret

menu_steff_new endp
```

Macros

Realiza la multiplicacion de dos variables y guarda el resultado en otra.

```
multiplicarFunc macro coƒ, var, res
   mov al, coƒ
   mov bl, var
   mul bl
   mov cl, al ; movemos el resultado a cx
   mov res, cx
endm
```

Realiza la suma de dos variables y lo guarda en la primera variable ingresada.

```
sumarFunc macro N1, N2

   mov ax, N1          ; MOV VARIABLE N1 A al
   adc ax, N2          ; SUMAR N1 Y N2
   ;add al, 30h          ; SUMAR RESULTADO CON 30H/48D
   mov N1, ax          ; MOVER RESULTADO A N3

endm
```

Realiza la resta de dos variables y lo guarda en la segunda variable ingresada.

```
restarFunc macro N1, N2
   mov ax, N1
   sbb ax, N2
   mov N2, ax
endm
```

Realiza la potencia de los números ingresados, lo guarda en una variable diferente y también recibe un contador que le ayuda a realizar la potencia.

```
Potencia macro base, exp, pot, cont
   local product
   push dx
   push ax

   ;xor cx, cx
   mov cl, exp ;Se cambia el registro cx con el numero de iteraciones
   ;mov varAux, cl
   mov al, base
   mov pot, al
   product:
   mov al, base ; Se mueve el valor de la base al registro al
```

```
   mov bl, pot ; Se mueve a bl el valor de variable pot
   mul bl ; Se multiplica a bl
   mov pot, al ; Se pone el resultado en la variable pot
   inc cont


   cmp cont, cl
   jne product

   pop ax
   pop dx
endm
```

Imprime la cadena que se le pase.

```
;Imprime una cadena
Print macro cadena
   push dx
   push ax

   mov ah, 09h
   mov dx, offset cadena
   int 21h

   pop ax
   pop dx
endm
```

Activa el modo video y guarda el modo texto.

```
ModoVideo macro video
   mov ah,0Fh  ; Petición de obtención de modo de vídeo
   int 10h     ; Llamada al BIOS
   mov video,al

   mov ah,00h  ; Función para establecer modo de video
   mov al,13h  ; Modo gráfico resolución 640x480
   int 10h
endm
```

Activa el modo texto con la ayuda de una variable que guarda el modo video.

```
ModoTexto macro video
   mov ah,00h    ; Función para re-establecer modo de texto
   mov al,video
   int 10h       ; Llamada al BIOS
endm
```

Imprime el eje X y el eje Y en el modo video.

```
PrintEjes macro
   local ejeX, ejeY
   xor cx, cx
   ejeX:
      mov ah, 0Ch
      mov al, 06; Color rojo
      mov bh, 0h
      mov dx, 100
      int 10h
      inc cx
      cmp cx, 320
      jne ejeX

   xor dx, dx
   ejeY:
      mov ah, 0Ch
      mov al, 06; Color rojo
      mov bh, 0h
      mov cx, 160
      int 10h
      inc dx
      cmp dx, 200
      jne ejeY

endm
```

Imprime un punto, anterior mente se deben manipular las variables cx y dx.

```
PrintDot2 macro
   ;mov cx, pmx
   ;mov dx, pmy
   mov ah, 0Ch
   mov al, 10
   mov bh, 0h
   int 10h

endm
```

Link github: https://github.com/TheJhonXD/-ACE-ProyectoUnico_201900831.git