**ARQUITECTURA Y PROGRAMACIÓN DE COMPUTADORAS**



**PROYECTO FINAL**

Programación en Ensamblador

**PORTADA**

**PROYECTO FINAL**

**CALCULADORA BÁSICA EN LENGUAJE ENSAMBLADOR**

* Nombres de los Integrantes del Equipo
* Grupo,
* Nombre de la Materia
* Nombre del Profesor
* Fecha de entrega.

ÍNDICE.

Incluye todas las partes del trabajo. Antes de empezar a redactar, es recomendable haber esbozado un índice inicial. Sin embargo, este apartado irá variando a medida que avancemos.

INTRODUCCIÓN.

Se realiza una breve exposición del contenido del proyecto.  (Deberá de estar integrado con al menos 1 cuartilla)

**DESARROLLO.**

1. Calculadora Básica
2. Colocar todos los programas y procedimientos que componen el proyecto.

**;menu**

sel:

call menu

MOV BL, 1h

call cambia\_color

lea dx, slop

call imprime\_cadena\_color

call lee\_caracter

cmp seleccion,31h

je opcion\_1

cmp seleccion,32h

je opcion\_2

cmp seleccion,33h

je opcion\_3

cmp seleccion,34h

je opcion\_4

cmp seleccion,35h

je opcion\_5

cmp seleccion,36h

je opcion\_6

cmp seleccion,37h

je fin

**;Conversiones**

**;decimal a binario**

opcion\_1:

clean

modo

call fondo

mov seleccion,0

mov RESULTADO,0

MOV BL, 1h

call cambia\_color

lea dx,op1

call imprime\_cadena\_color

call leedec

MOV DX,RESULTADO

CALL write\_binario

MOV BL, 1h

call cambia\_color

lea dx, otra

call imprime\_cadena\_color

call lee\_caracter

cmp seleccion,53h

je sel

**;decimal a octal**

opcion\_2:

clean

modo

call fondo

mov seleccion,0

mov RESULTADO,0

MOV BL, 1h

call cambia\_color

lea dx,op2

call imprime\_cadena\_color

call leedec

MOV DX, RESULTADO

CALL write\_octal

MOV BL, 1h

call cambia\_color

lea dx, otra

call imprime\_cadena\_color

call lee\_caracter

cmp seleccion,53h

je sel

cmp seleccion,53h

je fin

**;decimal a hexadecimal**

opcion\_3:

clean

modo

call fondo

mov seleccion,0

mov RESULTADO,0

MOV BL, 1h

call cambia\_color

lea dx,op3

call imprime\_cadena\_color

call leedec

MOV DX,RESULTADO

CALL write\_hexadecimal

MOV BL, 1h

call cambia\_color

lea dx, otra

call imprime\_cadena\_color

call lee\_caracter

cmp seleccion,53h

je sel

**;binario a decimal**

opcion\_4:

clean

modo

call fondo

mov seleccion,0

mov RESULTADO,0

MOV BL, 1h

call cambia\_color

lea dx,op4

call imprime\_cadena\_color

call leeBin

MOV DX,RESULTADO

CALL write\_decimal

MOV BL, 1h

call cambia\_color

lea dx, otra

call imprime\_cadena\_color

call lee\_caracter

cmp seleccion,53h

je sel

**;octal a decimal**

opcion\_5:

clean

modo

call fondo

mov seleccion,0

mov RESULTADO,0

MOV BL, 1h

call cambia\_color

lea dx,op5

call imprime\_cadena\_color

call leeoct

MOV DX, RESULTADO

CALL write\_decimal

MOV BL, 1h

call cambia\_color

lea dx, otra

call imprime\_cadena\_color

call lee\_caracter

cmp seleccion,53h

je sel

**;hexadecimal a decimal**

opcion\_6:

clean

modo

call fondo

mov seleccion,0

mov RESULTADO,0

MOV BL, 1h

call cambia\_color

lea dx,op6

call imprime\_cadena\_color

call leehex

MOV DX, RESULTADO

CALL write\_decimal

MOV BL, 1h

call cambia\_color

lea dx, otra

call imprime\_cadena\_color

call lee\_caracter

cmp seleccion,53h

je sel

fin:

.exit

**;procedimientos**

**imprime\_cadena proc**

mov ah,9

int 21h

ret

**imprime\_cadena ENDP**

**lee\_caracter proc**

push ax

mov ah, 1

int 21h

mov seleccion, al

pop ax

ret

**lee\_caracter endp**

**leedec PROC**

push ax

push bx

push cx

push dx

; IMPRIME MENSAJE

MOV BL, 1h

call cambia\_color

LEA DX,MSG\_LD

CALL imprime\_cadena\_color

;LEE NUMERO MILES (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV MILES,AL

; LEE NUMERO CENTENAS (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV CENTENAS,AL

;LEE NUMERO DECENAS (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV DECENAS,AL

;LEE NUMERO UNIDADES (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV UNIDADES,AL

XOR AX,AX

MOV AL,MILES

MOV BX,1000

MUL BX ; AL = AL \* BL

ADD RESULTADO,AX

MOV AL,CENTENAS

MOV BL,100

MUL BL ; AL = AL \* BL

ADD RESULTADO,AX

MOV AL,DECENAS

MOV BL,10

MUL BL ; AL = AL \* BL

ADD RESULTADO,AX

MOV AL,UNIDADES

MOV BL,U\_1

MUL BL ; AL = AL \* BL

ADD RESULTADO,AX

;IMPRIME MENSAJE lo imprime en hexadecimal

MOV BL, 1h

call cambia\_color

LEA DX,MSG\_RE

CALL imprime\_cadena\_color

pop dx

pop cx

pop bx

pop ax

ret

**leedec endp**

**leebin PROC**

push ax

push bx

push cx

push dx

; IMPRIME MENSAJE

MOV BL, 1h

call cambia\_color

LEA DX,MSG\_LD

CALL imprime\_cadena\_color

;LEE NUMERO 2^3 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV OCHO,AL

; LEE NUMERO 2^2 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV CUATRO,AL

;LEE NUMERO 2^1 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV DOS,AL

;LEE NUMERO 2^0 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV UNO,AL

XOR AX,AX

MOV AL,OCHO

MOV BL,8

MUL BL ; AL = AL \* BL

ADD RESULTADO,AX

MOV AL,CUATRO

MOV BL,4

MUL BL ; AL = AL \* BL

ADD RESULTADO,AX

MOV AL,DOS

MOV BL,2

MUL BL ; AL = AL \* BL

ADD RESULTADO,AX

MOV AL,UNO

MOV BL,1

MUL BL ; AL = AL \* BL

ADD RESULTADO,AX

;IMPRIME MENSAJE lo imprime en hexadecimal

MOV BL, 1h

Call cambia color

LEA DX,MSG\_RE

CALL imprime\_cadena\_color

pop dx

pop cx

pop bx

pop ax

ret

**leebin endp**

**leeoct PROC**

push ax

push bx

push cx

push dx

; IMPRIME MENSAJE

MOV BL, 1h

call cambia\_color

LEA DX,MSG\_LD

CALL imprime\_cadena\_color

;LEE NUMERO 8^3 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV OCHO\_CUBO,AL

; LEE NUMERO 8^2 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV OCHO\_CUADRADO,AL

;LEE NUMERO 8^1 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV OCHO\_o,AL

;LEE NUMERO 8^0 (UN DIGITO)

MOV AH,01H

INT 21H

SUB AL,30H

MOV UNO\_o,AL

XOR AX,AX

MOV AL,OCHO\_CUBO

MOV BX,512

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

MOV AL,OCHO\_CUADRADO

MOV BX,64

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

MOV AL,OCHO\_o

MOV BX,8

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

MOV AL,UNO\_o

MOV BX,1

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

;IMPRIME MENSAJE lo imprime en hexadecimal

MOV BL, 1h

call cambia\_color

LEA DX,MSG\_RE

CALL imprime\_cadena\_color

pop dx

pop cx

pop bx

pop ax

ret

**leeoct endp**

**leehex PROC**

push ax

push bx

push cx

push dx

; IMPRIME MENSAJE

MOV BL, 1h

call cambia\_color

LEA DX,MSG\_LD

CALL imprime\_cadena\_color

;LEE NUMERO 16^3 (UN DIGITO)

MOV AH,01H

INT 21H

CALL HEX\_TO\_DEC

MOV HEX\_CUBO,AL

; LEE NUMERO 16^2 (UN DIGITO)

MOV AH,01H

INT 21H

CALL HEX\_TO\_DEC

MOV HEX\_CUADRADO,AL

;LEE NUMERO 16^1 (UN DIGITO)

MOV AH,01H

INT 21H

CALL HEX\_TO\_DEC

MOV HEX,AL

;LEE NUMERO 16^0 (UN DIGITO)

MOV AH,01H

INT 21H

CALL HEX\_TO\_DEC

MOV UNO\_h,AL

XOR AX,AX

MOV AL,HEX\_CUBO

MOV BX,4096

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

MOV AL,HEX\_CUADRADO

MOV BX,256

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

MOV AL,HEX

MOV BX,16

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

MOV AL,UNO\_h

MOV BX,1

MUL BX ; AL = AL \* BX

ADD RESULTADO,AX

;IMPRIME MENSAJE lo imprime en hexadecimal

MOV BL, 1h

call cambia\_color

LEA DX,MSG\_RE

CALL imprime\_cadena\_color

pop dx

pop cx

pop bx

pop ax

ret

**leehex endp**

**write\_binario proc**

mov si,2

call decimal

ret

**write\_binario endp**

**write\_decimal proc**

mov si,10

call decimal

ret

**write\_decimal endp**

**write\_octal proc**

mov si,8

call decimal

ret

**write\_octal endp**

**write\_hexadecimal proc**

mov si,16

call decimal

ret

**write\_hexadecimal endp**

**decimal proc**

push ax

push cx

push dx

push si

mov ax, dx

xor cx, cx

non\_zero:

xor dx, dx

div si

push dx

inc cx

or ax, ax

jne non\_zero

ciclo:

pop dx

call imprime\_digito\_hexadecimal

loop ciclo

end\_decimal:

pop si

pop dx

pop cx

pop ax

ret

**decimal endp**

**IMPRIME\_HEXADECIMAL PROC**

PUSH CX

PUSH DX

MOV DH,DL

MOV CX,4

SHR DL,CL

CALL IMPRIME\_DIGITO\_HEXADECIMAL

MOV DL,DH

AND DL,0FH

CALL IMPRIME\_DIGITO\_HEXADECIMAL

POP DX

POP CX

RET

**IMPRIME\_HEXADECIMAL ENDP**

**IMPRIME\_DIGITO\_HEXADECIMAL PROC**

PUSH DX

CMP DL,10

JAE LETRA\_HEXADECIMAL

ADD DL,"0"

JMP IMPRIME\_DIGITO

LETRA\_HEXADECIMAL:

ADD DL,"A"-10

IMPRIME\_DIGITO:

CALL IMPRIME\_CARACTER

POP DX

RET

**IMPRIME\_DIGITO\_HEXADECIMAL ENDP**

**imprime\_caracter proc**

push ax

mov ah,02

int 21h

pop ax

ret

**imprime\_caracter endp**

**HEX\_TO\_DEC PROC**

CMP AL, '0'

JB f

CMP AL, '9'

JBE dig

CMP AL, 'A'

JB f

CMP AL, 'F'

JA f

; Si es una letra (A-F), resta 'A' y suma 10

SUB AL, 'A'

ADD AL, 10

JMP f

dig:

; Si es un dígito (0-9), resta '0'

SUB AL, '0'

f:

RET

**HEX\_TO\_DEC ENDP**

**MENU PROC**

push ax

push bx

push dx

clean

modo

call fondo

MOV BL, 1h ; Establece el color del texto (1Ch = rojo brillante sobre fondo azul)

call cambia\_color

lea dx,nu

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,cu

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,barr

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,mat

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,tit

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,barr

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,me

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,op1

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op2

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op3

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op4

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op5

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op6

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,barr

call imprime\_cadena\_color

call sl

POP AX

POP BX

POP DX

ret

**MENU ENDP**

**sl proc**

push dx

mov dl,13

call imprime\_caracter

mov dl,10

call imprime\_caracter

pop dx

ret

**sl endp**

**fondo proc**

push ax

push bx

MOV AH, 0Bh ; Establece el servicio

MOV BH, 0 ; Establece el modo de paleta

MOV BL,4 ; Establece el color de fondo

INT 10h ; Llama a la interrupción

pop bx

pop ax

ret

**fondo endp**

**imprime\_cadena\_color PROC**

PUSH SI

PUSH ax

MOV SI, DX ; Obtiene la dirección de la cadena

next\_char:

MOV AH, 0Eh ; Establece el servicio

MOV AL, [SI] ; Obtiene el carácter actual

INT 10h ; Llama a la interrupción

INC SI ; Avanza al siguiente carácter

CMP BYTE PTR [SI],0 ; Comprueba si hemos llegado al final de la cadena

JNE next\_char ; Si no, repite el proceso para el siguiente carácter

POP AX

POP SI

RET

**imprime\_cadena\_color ENDP**

**cambia\_color PROC**

PUSH AX

; AH = 10h: Establece el modo de video

; AL = 0: Establece el modo de video

; BL = Color del texto

MOV AH, 10h

MOV AL, 0

INT 10h

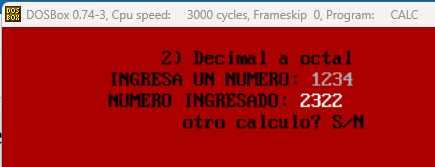
POP AX

RET

**cambia\_color ENDP**

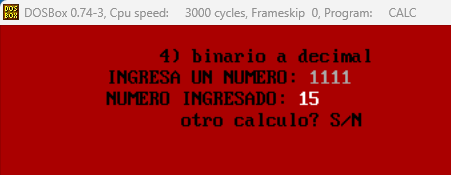
1. Realizar varios cálculos para demostrar su funcionamiento (al menos 5, para lo que deberá demostrar la captura de ventana de las corridas o ejecuciones.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

1. Descripción de Procedimientos

(Deberá de explicar brevemente los procedimientos utilizados, iniciando desde el programa principal)

Ejemplo:

* 1. Procedimiento imprime\_cadena

1. **Program**a

**imprime\_cadena proc**

mov ah,9

int 21h

ret

**imprime\_cadena ENDP**

1. **Descripción.**

El procedimiento imprime\_cadena usa la int 21h servicio 9, para imprimir una cadena previamente definida en la data segment

1. Para esto se deberá de colocar en el registro AH un 9, como se puede observas en la siguiente instrucción:

MOV AH,9

1. A partir de lo anterior, cuando se realice la INT 21H, imprimirá la cadena que se encuentre alojado en el registro DX.
2. Para poder imprimir una cadena previamente se usó la instrucción

Lea DX, cadena

* 1. Procedimiento lee\_caracter proc

1. Programa

**lee\_caracter proc**

push ax

mov ah, 1

int 21h

mov seleccion, al

pop ax

ret

**lee\_caracter endp**

1. descripción

El procedimiento usa la int 21h servicio 1, para imprimir un carácter en pantalla.

1. Para esto se deberá de colocar en el registro AH un 1, como se puede observas en la siguiente instrucción:

MOV AH,1

1. A partir de lo anterior, cuando se realice la INT 21H, imprimirá el carácter que se encuentre alojado en el registro DL.
   1. Procedimiento leedec
2. **Program**a

**leedec PROC**

**push ax**

**push bx**

**push cx**

**push dx**

**; IMPRIME MENSAJE**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_LD**

**CALL imprime\_cadena\_color**

**;LEE NUMERO MILES (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV MILES,AL**

**; LEE NUMERO CENTENAS (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV CENTENAS,AL**

**;LEE NUMERO DECENAS (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV DECENAS,AL**

**;LEE NUMERO UNIDADES (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV UNIDADES,AL**

**XOR AX,AX**

**MOV AL,MILES**

**MOV BX,1000**

**MUL BX ; AL = AL \* BL**

**ADD RESULTADO,AX**

**MOV AL,CENTENAS**

**MOV BL,100**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

**MOV AL,DECENAS**

**MOV BL,10**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

**MOV AL,UNIDADES**

**MOV BL,U\_1**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

**;IMPRIME MENSAJE**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_RE**

**CALL imprime\_cadena\_color**

**pop dx**

**pop cx**

**pop bx**

**pop ax**

**ret**

**leedec endp**

1. **Descripción.**

El procedimiento lee un numero decimal de 4 dígitos, los separa en miles, centenas, decenas y unidad, para posteriormente multiplicarlos por 1000,100,10,1 respectivamente.

1. Para leer cada digito se usa la INT 21h serv 1, se deberá de colocar en el registro AH un 1, llamar a la INT 21h, restar 30h al registro AL, y por último mover Al a la variable MILES como se puede observas en la siguiente instrucción:

**MOV AH,01H**

**INT 21H**

**SUB AL, 30H**

**MOV MILES,AL**

(esto se repite para los demás dígitos y se mueve a su variable correspondiente)

1. Para comenzar a multiplicar se limpia el registro AX usando la siguiente instrucción:

XOR AX,AX

1. Para realizar la multiplicación de cada digito se mueve la variable correspondiente a AL, después se mueve al registro BX el numero por el cual se va a multiplicar, se multiplica AL por BL y por ultimo se suma AX a la variable resultado con las siguientes instrucciones:

**MOV AL,MILES**

**MOV BX,1000**

**MUL BX**

**ADD RESULTADO,AX**

(se repite para los demás dígitos, con el valor correspondiente 100,10,1)

* 1. Procedimiento leebin

1. **Program**a

**Leebin proc**

**push ax**

**push bx**

**push cx**

**push dx**

**; IMPRIME MENSAJE**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_LD**

**CALL imprime\_cadena\_color**

**;LEE NUMERO 2^3 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV OCHO,AL**

**; LEE NUMERO 2^2 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV CUATRO,AL**

**;LEE NUMERO 2^1 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV DOS,AL**

**;LEE NUMERO 2^0 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV UNO,AL**

**XOR AX,AX**

**MOV AL,OCHO**

**MOV BL,8**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

**MOV AL,CUATRO**

**MOV BL,4**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

**MOV AL,DOS**

**MOV BL,2**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

**MOV AL,UNO**

**MOV BL,1**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

**;IMPRIME MENSAJE lo imprime en hexadecimal**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_RE**

**CALL imprime\_cadena\_color**

**pop dx**

**pop cx**

**pop bx**

**pop ax**

**ret**

**leebin endp**

1. **Descripción.**

El procedimiento leebin lee un numero binario de 4 dígitos, para la conversión de un numero binario a decimal este debe ser multiplicado por 2 elevado a una potencia dependiendo de cuantos dígitos sean y su posición que en este caso seria 2^3, 2^2, 2^1, 2^0, para esto se definieron 4 variables las cuales son ocho, cuatro, dos, uno.

1. Para leer cada digito se usa la INT 21h serv 1, se deberá de colocar en el registro AH un 1, llamar a la INT 21h, restar 30h al registro AL, y por último mover Al a la variable OCHO como se puede observas en la siguiente instrucción:

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV OCHO,AL**

(esto se repite para los demás dígitos y se mueve a su variable correspondiente)

1. Para comenzar a multiplicar se limpia el registro AX usando la siguiente instrucción:

XOR AX,AX

1. Para realizar la multiplicación de cada digito se mueve la variable correspondiente a AL, después se mueve al registro BX el numero por el cual se va a multiplicar, se multiplica AL por BL y por ultimo se suma AX a la variable resultado con las siguientes instrucciones:

**MOV AL,OCHO**

**MOV BL,8**

**MUL BL ; AL = AL \* BL**

**ADD RESULTADO,AX**

(se repite para los demás dígitos, con el valor correspondiente 4, 2, 1)

* 1. Procedimiento leeoct

1. **Program**a

**leeoct PROC**

**push ax**

**push bx**

**push cx**

**push dx**

**; IMPRIME MENSAJE**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_LD**

**CALL imprime\_cadena\_color**

**;LEE NUMERO 8^3 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV OCHO\_CUBO,AL**

**; LEE NUMERO 8^2 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV OCHO\_CUADRADO,AL**

**;LEE NUMERO 8^1 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV OCHO\_o,AL**

**;LEE NUMERO 8^0 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV UNO\_o,AL**

**XOR AX,AX**

**MOV AL,OCHO\_CUBO**

**MOV BX,512**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**MOV AL,OCHO\_CUADRADO**

**MOV BX,64**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**MOV AL,OCHO\_o**

**MOV BX,8**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**MOV AL,UNO\_o**

**MOV BX,1**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**;IMPRIME MENSAJE lo imprime en hexadecimal**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_RE**

**CALL imprime\_cadena\_color**

**pop dx**

**pop cx**

**pop bx**

**pop ax**

**ret**

**leeoct endp**

1. **Descripción.**

El procedimiento leeoct lee un numero octal de 4 dígitos, para la conversión de un numero octal a decimal este debe ser multiplicado por 8 elevado a una potencia dependiendo de cuantos dígitos sean y su posición que en este caso seria 8^3, 8^2, 8^1, 8^0, para esto se definieron 4 variables las cuales son ocho\_cubo, ocho\_cuadrado, ocho\_o, uno\_o.

1. Para leer cada digito se usa la INT 21h serv 1, se deberá de colocar en el registro AH un 1, llamar a la INT 21h, restar 30h al registro AL, y por último mover Al a la variable OCHO\_CUBO como se puede observas en la siguiente instrucción:

**MOV AH,01H**

**INT 21H**

**SUB AL,30H**

**MOV OCHO\_CUBO,AL**

(esto se repite para los demás dígitos y se mueve a su variable correspondiente)

1. Para comenzar a multiplicar se limpia el registro AX usando la siguiente instrucción:

XOR AX,AX

1. Para realizar la multiplicación de cada digito se mueve la variable correspondiente a AL, después se mueve al registro BX el numero por el cual se va a multiplicar, se multiplica AL por BL y por último se suma AX a la variable resultado con las siguientes instrucciones:

**MOV AL,OCHO\_CUBO**

**MOV BX,512**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

(se repite para los demás dígitos, con el valor correspondiente 64, 8, 1)

* 1. Procedimiento leehex

1. **Program**a

**leehex PROC**

**push ax**

**push bx**

**push cx**

**push dx**

**; IMPRIME MENSAJE**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_LD**

**CALL imprime\_cadena\_color**

**;LEE NUMERO 16^3 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**CALL HEX\_TO\_DEC**

**MOV HEX\_CUBO,AL**

**; LEE NUMERO 16^2 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**CALL HEX\_TO\_DEC**

**MOV HEX\_CUADRADO,AL**

**;LEE NUMERO 16^1 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**CALL HEX\_TO\_DEC**

**MOV HEX,AL**

**;LEE NUMERO 16^0 (UN DIGITO)**

**MOV AH,01H**

**INT 21H**

**CALL HEX\_TO\_DEC**

**MOV UNO\_h,AL**

**XOR AX,AX**

**MOV AL,HEX\_CUBO**

**MOV BX,4096**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**MOV AL,HEX\_CUADRADO**

**MOV BX,256**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**MOV AL,HEX**

**MOV BX,16**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**MOV AL,UNO\_h**

**MOV BX,1**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

**;IMPRIME MENSAJE lo imprime en hexadecimal**

**MOV BL, 1h**

**call cambia\_color**

**LEA DX,MSG\_RE**

**CALL imprime\_cadena\_color**

**pop dx**

**pop cx**

**pop bx**

**pop ax**

**ret**

**leehex endp**

1. **Descripción.**

El procedimiento leebin lee un numero hexadecimal de 4 dígitos, para la conversión de un numero hexadecimal a decimal este debe ser multiplicado por 16 elevado a una potencia dependiendo de cuantos dígitos sean y su posición que en este caso seria 16^3, 16^2, 16^1, 16^0, para esto se definieron 4 variables las cuales son ocho\_cubo, ocho\_cuadrado, ocho\_o, uno\_o.

1. Para leer cada digito se usa la INT 21h serv 1, se deberá de colocar en el registro AH un 1, llamar a la INT 21h, restar 30h al registro AL, y por último mover Al a la variable HEX\_CUBO como se puede observas en la siguiente instrucción:

**MOV AH,01H**

**INT 21H**

**CALL HEX\_TO\_DEC**

**MOV HEX\_CUBO,AL**

(esto se repite para los demás dígitos y se mueve a su variable correspondiente)

1. Para comenzar a multiplicar se limpia el registro AX usando la siguiente instrucción:

XOR AX,AX

1. Para realizar la multiplicación de cada digito se mueve la variable correspondiente a AL, después se mueve al registro BX el numero por el cual se va a multiplicar, se multiplica AL por BL y por último se suma AX a la variable resultado con las siguientes instrucciones:

**MOV AL,HEX\_CUBO**

**MOV BX,4096**

**MUL BX ; AL = AL \* BX**

**ADD RESULTADO,AX**

(se repite para los demás dígitos, con el valor correspondiente 256, 16, 1)

* 1. Procedimiento write\_binario/decimal/octal/hexadecimal

1. **Program**a

write\_binario proc

mov si,2

call decimal

ret

write\_binario endp

write\_decimal proc

mov si,10

call decimal

ret

write\_decimal endp

write\_octal proc

mov si,8

call decimal

ret

write\_octal endp

write\_hexadecimal proc

mov si,16

call decimal

ret

write\_hexadecimal endp

1. **Descripción.**

El procedimiento write\_binario/decimal/octal/hexadecimal son 4 procedimientos que con la ayuda de otro procedimiento llamado decimal, convierten un numero de 4 dígitos a la base deseada

1. Movemos el numero de la base deseada en este caso binario que es 2, y lo movemos al registro SI para posteriormente llamar a decimal

mov si,2

call decimal

ret

* 1. Procedimiento decimal

1. **Program**a

decimal proc

push ax

push cx

push dx

push si

mov ax,dx

xor cx,cx

non\_zero:

xor dx,dx

div si

push dx

inc cx

or ax,ax

jne non\_zero

ciclo:

pop dx

call imprime\_digito\_hexadecimal

loop ciclo

end\_decimal:

pop si

pop dx

pop cx

pop ax

ret

decimal endp

1. **Descripción.**

El procedimiento decimal ayuda a la conversión de un numero de 4 dígitos con la ayuda de write\_binario/decimal/octal/hexadecimal al brindarle la base deseada, la conversión se realiza en base a divisiones sucesivas por la base previamente determinada

1. Previo al procedimiento moveremos la variable resultado a DX con la siguiente instrucción

MOV DX,RESULTADO

1. Moveremos DX a AX y limpiaremos CX

mov ax, dx

xor cx, cx

1. Crearemos la etiqueta non\_zero, limpiamos el registro DX, se divide por SI que se definió por el procedimiento anterior, se guarda DX, se incrementa 1 a CX se hace un or entre AX y AX y hace un salto si no es igual

non\_zero:

xor dx,dx

div si

push dx

inc cx

or ax,ax

jne non\_zero

1. Creamos la etiqueta CICLO para imprimir cada digito , saca el valor de DX que guardo anteriormente y se llama al procedimiento imprime\_digito\_hexadecimal

Ciclo:

pop dx

call imprime\_digito\_hexadecimal

loop ciclo

* 1. Procedimiento IMPRIME\_HEXADECIMAL

1. **Program**a

IMPRIME\_HEXADECIMAL PROC

PUSH CX

PUSH DX

MOV DH,DL

MOV CX,4

SHR DL,CL

CALL IMPRIME\_DIGITO\_HEXADECIMAL

MOV DL,DH

AND DL,0FH

CALL IMPRIME\_DIGITO\_HEXADECIMAL

POP DX

POP CX

RET

IMPRIME\_HEXADECIMAL ENDP

1. **Descripción.**

El procedimiento imprime\_hexadecimal imprime 2 dígitos hexadecimales con la ayuda del procedimiento imprime\_digito\_hexadecimal

1. Movemos DL a DH, movemos 4 a CX, se recorre DL a la derecha por la cantidad de CX y se llama al procedimiento MPRIME\_DIGITO\_HEXADECIMAL como se observa en las siguientes instrucciones:

MOV DH, DL

MOV CX, 4

SHR DL, CL

CALL IMPRIME\_DIGITO\_HEXADECIMAL

1. Restauramos el valor d DL, realizamos un and entre DL y 0Fh para conservar los 4 bits menos significativos y volvemos a llamar a IMPRIME\_DIGITO\_HEXADECIMAL

MOV DL, DH

AND DL, 0FH

CALL IMPRIME\_DIGITO\_HEXADECIMAL

* 1. Procedimiento IMPRIME\_DIGITO\_HEXADECIMAL

1. **Program**a

IMPRIME\_DIGITO\_HEXADECIMAL PROC

PUSH DX

CMP DL,10

JAE LETRA\_HEXADECIMAL

ADD DL,"0"

JMP IMPRIME\_DIGITO

LETRA\_HEXADECIMAL:

ADD DL,"A"-10

IMPRIME\_DIGITO:

CALL IMPRIME\_CARACTER

POP DX

RET

IMPRIME\_DIGITO\_HEXADECIMAL ENDP

1. **Descripción.**

El procedimiento imprime\_digito\_hexadecimal imprime letra o numero en valor hexadecimal

1. Comparamos DL con 10, si el valor de DL es mayor o igual a 10 salta a la etiqueta letra\_hexadecimal, de no ser así, se suma el código ASCII de 0 a DL y salta a la etiqueta imprime\_digito.

CMP DL,10

JAE LETRA\_HEXADECIMAL

ADD DL,"0"

JMP IMPRIME\_DIGITO

1. La etiqueta de letra\_hexadecimal suma el código ASCII de A y le resta 10 a DL

LETRA\_HEXADECIMAL:

ADD DL,"A"-10

1. La etiqueta de imprime\_digito llama al procedimiento imprime\_caracter para imprimir el numero o letra según sea el caso.

IMPRIME\_DIGITO:

CALL IMPRIME\_CARACTER

* 1. Procedimiento Imprime\_caracter

1. **Program**a

push ax

mov ah,02

int 21h

pop ax

ret

1. **Descripción.**

Usando la INT 21h serv 02, imprime un carácter en la pantalla.

1. Para esto se deberá de colocar en el registro AH un 2, como se puede observas en la siguiente instrucción:

MOV AH,02

1. A partir de lo anterior, cuando se realice la INT 21H, imprimirá el carácter que se encuentre alojado en el registro DL.
   1. Procedimiento HEX\_to\_dec
2. **Program**a

HEX\_TO\_DEC PROC

CMP AL, '0'

JB f

CMP AL, '9'

JBE dig

CMP AL, 'A'

JB f

CMP AL, 'F'

JA f

; Si es una letra (A-F), resta 'A' y suma 10

SUB AL, 'A'

ADD AL, 10

JMP f

dig:

; Si es un dígito (0-9), resta '0'

SUB AL, '0'

f:

RET

HEX\_TO\_DEC ENDP

1. **Descripción.**

Convierte un digito hexadecimal a un número decimal.

1. Compara el valor de AL con 0, salta a la etiqueta f si el valor de AL es menor a 0, compara el valor de AL con 9, salta a la etiqueta dig si el valor de AL es igual o menor a 9, compara Al con A, salta a la etiqueta f si el valor de AL es menor a A, compara el valor de AL con F, salta a la etiqueta f si el valor de AL es mayor a F

CMP AL, '0'

JB f

CMP AL, '9'

JBE dig

CMP AL, 'A'

JB f

CMP AL, 'F'

JA f

1. Resta el valor de A a AL, suma 10 a Al, salta a la etiqueta f (la etiqueta f manda al final del procedimiento)

; Si es una letra (A-F), resta 'A' y suma 10

SUB AL, 'A'

ADD AL, 10

JMP f

1. La etiqueta dig resta el valor de 0 a AL y salta a la etiqueta f

dig:

; Si es un dígito (0-9), resta '0'

SUB AL, '0'

* 1. Procedimiento menu

1. **Program**a

MENU PROC

push ax

push bx

push dx

clean

modo

call fondo

MOV BL, 1h ; Establece el color del texto (1Ch = rojo brillante sobre fondo azul)

call cambia\_color

lea dx,nu

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,cu

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,barr

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,mat

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,tit

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,barr

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,me

call imprime\_cadena\_color

call sl

MOV BL, 1h

call cambia\_color

lea dx,op1

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op2

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op3

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op4

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op5

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,op6

call imprime\_cadena\_color

MOV BL, 1h

call cambia\_color

lea dx,barr

call imprime\_cadena\_color

call sl

POP AX

POP BX

POP DX

ret

MENU ENDP

1. **Descripción.**

Imprime en pantalla una serie de cadenas que crean un menú

1. Usa los macros clean y modo, ambos usan la INT 10h para configurar las opciones de video, clean limpia la pantalla, y modo, configura la resolución, después se llama al proc fondo, que le da un color especifico al fondo usando la INT 10h

clean

modo

call fondo

1. Mueve 1 a BL que establece el color del texto, para llamar al proc cambia\_color, lee la cadena nu en DX y llama al proc imprime\_cadena\_color.

MOV BL, 1h ; Establece el color del texto

call cambia\_color

lea dx,nu

call imprime\_cadena\_color

(se repite para todas las cadenas del proc)

1. Se llama al proc sl para realizar un salto de línea

call sl

* 1. Procedimiento sl

1. **Program**a

sl proc

push dx

mov dl,13

call imprime\_caracter

mov dl,10

call imprime\_caracter

pop dx

ret

sl endp

1. **Descripción.**

Realiza un salto de línea.

1. Mueve 13 a DL, llama al proc imprime\_caracter, Mueve 10a DL, llama al proc imprime\_caracter

mov dl,13

call imprime\_caracter

mov dl,10

call imprime\_caracter

* 1. Procedimiento fondo

1. **Program**a

fondo proc

push ax

push bx

MOV AH, 0Bh ; Establece el servicio

MOV BH, 0 ; Establece el modo de paleta

MOV BL,4 ; Establece el color de fondo

INT 10h ; Llama a la interrupción

pop bx

pop ax

ret

fondo endp

1. **Descripción.**

Cambia el color del fondo usando la INT 10h serv 0Bh

1. Mueve 0B a Ah para el serv, mueve 0 a BH para la paleta de color, mueve 4 a BL para seleccionar el color y llama a la INT 10h

MOV AH, 0Bh ; Establece el servicio

MOV BH, 0 ; Establece el modo de paleta

MOV BL,4 ; Establece el color de fondo

INT 10h ; Llama a la interrupción

* 1. Procedimiento imprime\_cadena\_color

1. **Program**a

imprime\_cadena\_color PROC

PUSH SI

PUSH ax

MOV SI, DX ; Obtiene la dirección de la cadena

next\_char:

MOV AH, 0Eh ; Establece el servicio

MOV AL, [SI] ; Obtiene el carácter actual

INT 10h ; Llama a la interrupción

INC SI ; Avanza al siguiente carácter

CMP BYTE PTR [SI],0 ; Comprueba si hemos llegado al final de la cadena

JNE next\_char ; Si no, repite el proceso para el siguiente carácter

POP AX

POP SI

RET

imprime\_cadena\_color ENDP

1. **Descripción.**

Cambia el color del fondo usando la INT 10h serv 0Bh

1. Mueve DX a SI, la etiqueta next\_char mueve 0E a AH, mueve SI a AL, INT 10h, incrementa SI, compara la cadena con 0 todo esto sirve para imprimir carácter por carácter.

MOV SI, DX ; Obtiene la dirección de la cadena

next\_char:

MOV AH, 0Eh ; Establece el servicio

MOV AL, [SI] ; Obtiene el carácter actual

INT 10h ; Llama a la interrupción

INC SI ; Avanza al siguiente carácter

CMP BYTE PTR [SI],0 ; Comprueba si hemos llegado al final de la

1. Hace un salto a la etiqueta next\_char y realiza un salto al principio del loop

cadena

JNE next\_char ; Si no, repite el proceso para el siguiente carácter

* 1. Procedimiento cambia\_color

1. **Program**a

cambia\_color PROC

PUSH AX

; AH = 10h: Establece el modo de video

; AL = 0: Establece el modo de video

; BL = Color del texto

MOV AH, 10h

MOV AL, 0

INT 10h

POP AX

RET

cambia\_color ENDP

1. **Descripción.**

Usa la int 10h serv 10h para cambiar el color del texto

1. Mueve 10 a AH, mueve 0 a AL , INT 10h

1. **Introducción**

**Sistemas numéricos y conversiones**

1. **De decimal a binario**

Este método consiste en ir dividiendo la cantidad decimal por 2, apuntando los residuos, hasta obtener un cociente cero. El último residuo obtenido es el bit más significativo (MSB) y el primero es el bit menos significativo (LSB).

|  |  |
| --- | --- |
| Procedimiento   * Dividir entre 2 sucesivamente * Apuntar el resultado y el resto de cada operación * Apuntar a lista de ceros y unos de abajo a arriba |  |

1. **De decimal a octal**

En la conversión de una magnitud decimal a octal se realizan divisiones sucesivas por 8 hasta obtener la parte entera del cociente igual a cero.

Los residuos forman el número octal equivalente, siendo el último residuo el dígito más significativo y el primero el menos significativo

|  |  |
| --- | --- |
| Procedimiento   * Dividir entre 8 sucesivamente * Apuntar el resultado y el resto de cada operación * Apuntar a lista de RESIDUOS de abajo a arriba |  |

1. **De decimal a hexadecimal**

En la conversión de una magnitud decimal a hexadecimal se realizan divisiones sucesivas por 16 hasta obtener un cociente de cero.

Los residuos forman el número hexadecimal equivalente, siendo el último residuo el dígito más significativo y el primero el menos significativo.

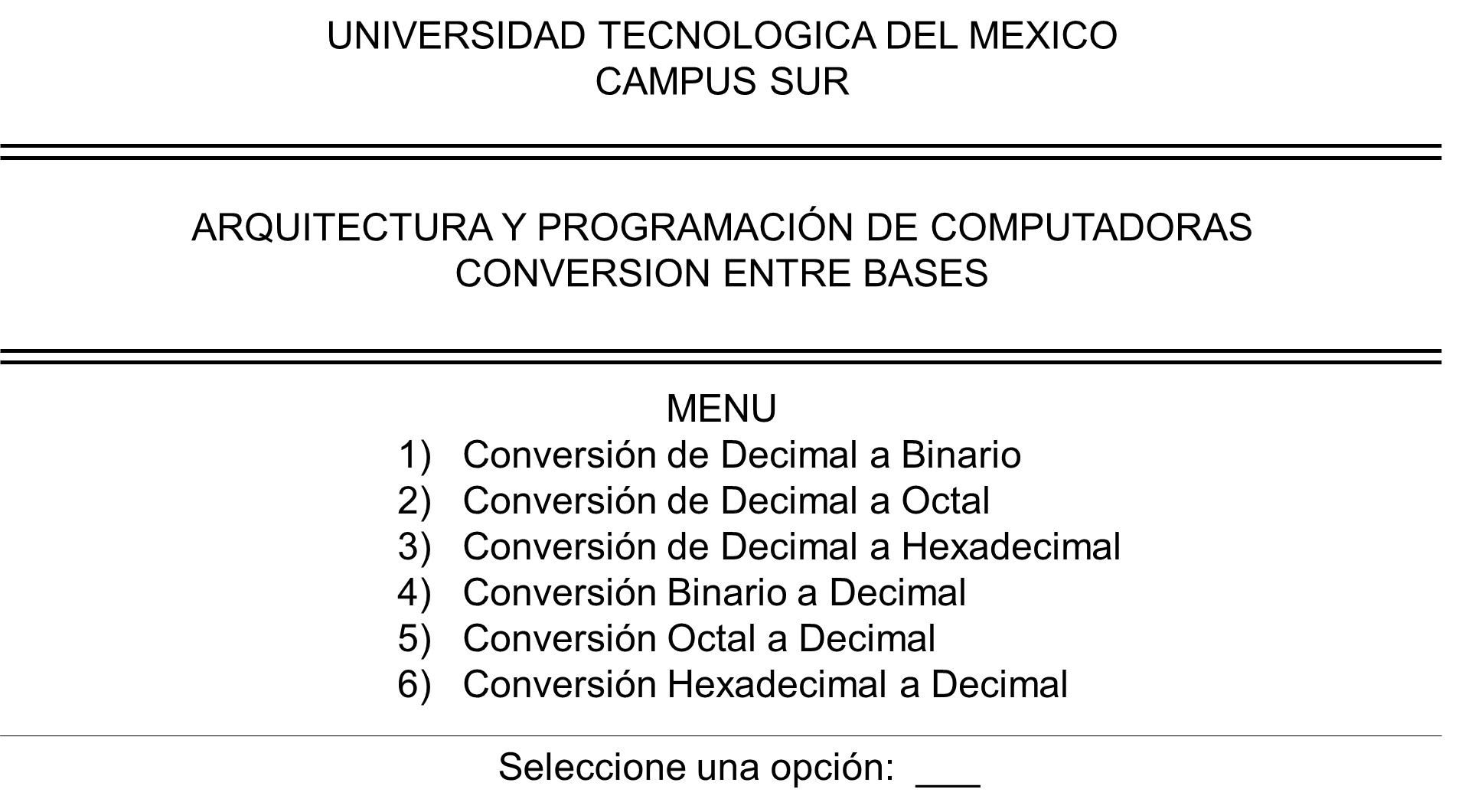
|  |  |
| --- | --- |
| Procedimiento   * Dividir entre 16 sucesivamente * Apuntar el resultado y el resto de cada operación * Apuntar a lista de RESIDUOS de abajo a arriba |  |

1. **Descripción del proyecto.**

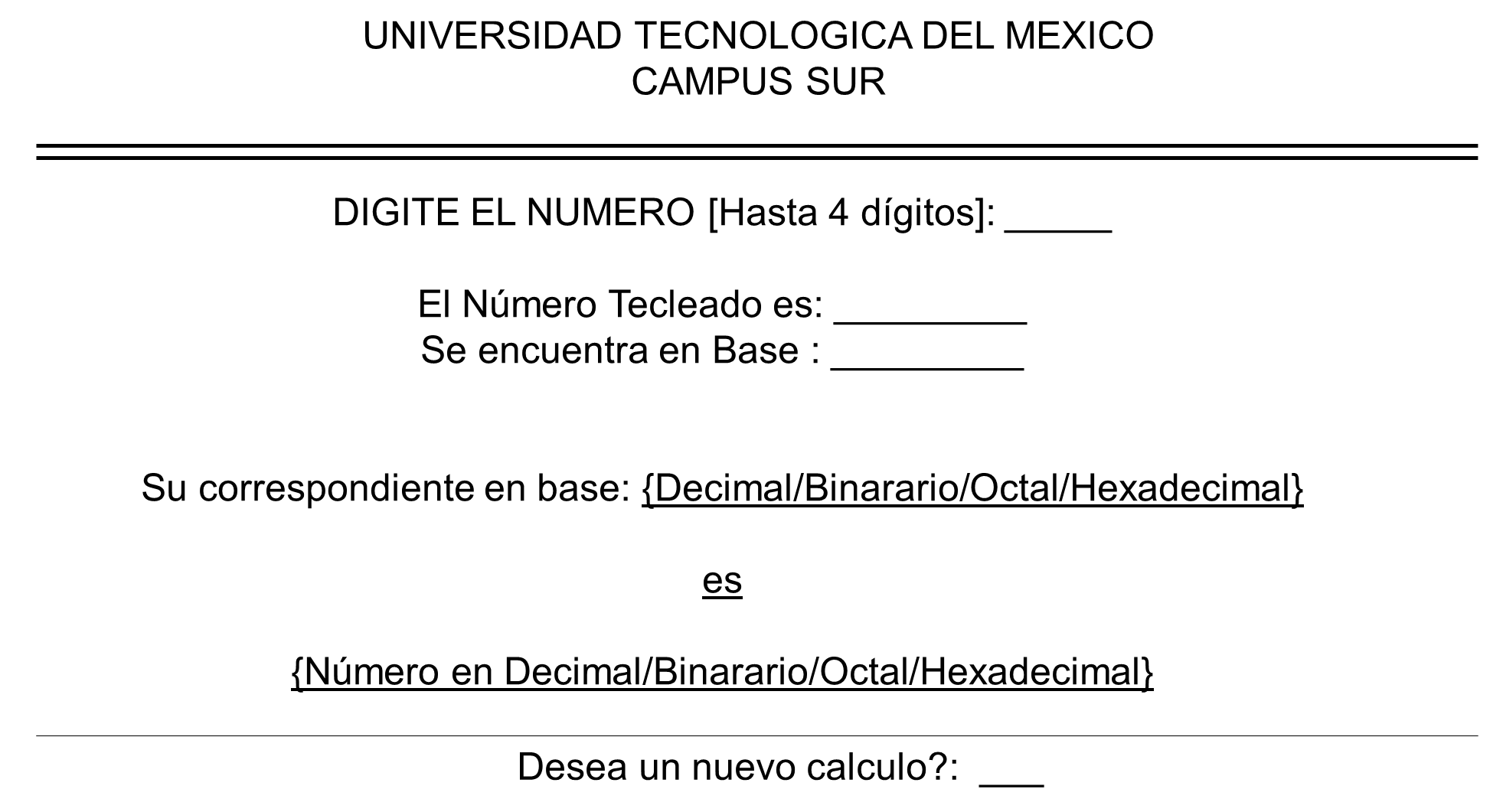
Considerando los procedimientos para la conversión entre bases mencionados en los **Sistemas numéricos y conversiones,** el proyecto consiste en realizar una tabla de conversiones en donde a partir de ingresar un numero decimal de hasta cuatro dígitos sea representado en sus correspondientes en base BINARIA, OCTAL y HEXADECOIMAL.

Para lo anterior, deberá considerar una vista que al menos cumpla con los que se muestra a continuación:

1. Página No. 1



1. Página No. 2



1. **Contenido del proyecto.**
2. Representar mediante un Diagrama de Flujo, el proceso que se está programando, utilizando Seudocódigo.
3. Enlistar todos los procedimientos utilizados los cuales deberán de estar estructurados de la siguiente forma:

* Archivo que contiene el Programa Principal
* Archivo que contiene todos los procedimientos utilizados (Librería)
* Cada procedimiento deberá de tener los siguientes apartados:
  + Codificación
  + Prueba de Escritorio
  + Su trazabilidad demostrada con el programa DEBUG.

Ejemplo:

|  |  |
| --- | --- |
| **SUBRUTINA** | |
|  | |
| **Prueba de Escritorio** | **Programa DEBUG** |
|  |  |
| **Descripción de la Subrutina.** | |
|  | |

1. Deberá anexar archivo en formato ZIP, el cual contenga todas las rutinas descritas y el programa principal para su evaluación.