ALGORITMO promocionTorricoSmigoski **RETORNA Ø**

(*Este algoritmo realiza las conversiones mencionadas en el menú según las necesidades del usuario*)

TEXTO numero, opcion, conversación

LOGICO verificación

REPETIR

ESCRIBIR (“Ingrese un numero (ya sea binario, decimal o hexadecimal):”)

LEER (numero)

ESCRIBIR(“Elija lo opción que desea realizar:\n”,

" 1. Verificar si es un numero binario valido.\n",

" 2. Verificar si es un numero hexadecimal valido.\n",

" 3. Verificar si es un numero decimal valido.\n",

" 4. Convertir binario a hexadecimal.\n",

" 5. Convertir binario a decimal.\n",

" 6. Convertir decimal a binario.\n",

" 7. Convertir decimal a hexadecimal.\n",

" 8. Convertir hexadecimal a binario.\n",

" 9. Convertir hexadecimal a decimal.\n",

" 10. Convertir binario a su complemento a uno.\n",

" 11. Convertir binario a su complemento a dos.\n",

" 12. Terminar.”**)**

LEER (opcion)

SEGÚN (opcion) **HACER**

1: verificación←verificarBinario(numero)

SI (verificación) **ENTONCES**

ESCRIBIR (“El numero ingresado es un numero binario valido.”)

SINO

ESCRIBIR (“El numero ingresado es un numero binario no valido.”)

FIN SI

2 : verificación←verificarHexadecimal(numero)

SI (verificación) **ENTONCES**

ESCRIBIR (“El numero ingresado es un numero hexadecimal valido.”)

SINO

ESCRIBIR (“El numero ingresado es un numero hexadecimal no valido.”)

FIN SI

3 : verificación ←verificarDecimal(numero)

SI (verificacion) **ENTONCES**

ESCRIBIR (“El numero ingresado es un numero decimal valido.”)

SINO

ESCRIBIR (“El numero ingresado es un numero decimal no valido.”)

FIN SI

4 : conversión ← convertirBinarioHexadecimal(numero)

ESCRIBIR (“El numero binario expresado en hexadecimal es: “,conversión)

5 : conversión ← convertirBinarioDecimal(numero)

ESCRIBIR (“El numero binario expresado en decimal es: “,conversión)

6 : conversión ← convertirDecimalBinario(numero)

ESCRIBIR (“El numero decimal expresado en binario es: ”,conversión)

7 : conversión ← convertirDecimalHexadecimal(numero)

SI(igual(conversión,””**)ENTONCES**

ESCRIBIR (“El numero ingresado es un numero decimal no valido.”)

SINO

ESCRIBIR (“El numero decimal expresado en hexadecimal es: “, conversión)

FIN SI

8: conversión ←convertirHexadecimalBinario(numero)

SI (igual(conversión,””)) **ENTONCES**

ESCRIBIR (“El numero ingresado es un numero hexadecimal no valido”)

SINO

ESCRIBIR (“El numero hexadecimal expresado en binario es: “,conversión)

FIN SI

9: conversión ←convertirHexadecimalDecimal(numero)

SI (igual(conversión,””)) **ENTONCES**

ESCRIBIR (“El numero ingresado es un numero hexadecimal no valido”)

SINO

ESCRIBIR (“El numero hexadecimal expresado en decimal es”, conversión)

FIN SI

10: convertirBinarioComplemento1(numero

11: convertirBinarioComplemento2(numero)

12: **ESCRIBIR** (“Gracias por utilizar el Programa”)

FIN SEGÚN

HASTA (opcion=”12”)

FIN ALGORITMO

MODULO verificarBinario **(TEXTO** opcion**) RETORNA LOGICO**

(*Este módulo verifica que el número ingresado sea Binario*)

ENTERO contador, i

LOGICO verificación

contador← 0

i ← 0

MIENTRAS (i < longitud(opcion)) **HACER**

SI ((posición (opción, i) = 1) OR (posición (opción, i) = 0)) **ENTONCES**

contador ← contador+1

FIN SI

i ← i+1

FIN MIENTRAS

verificación ← contador = i

RETORNA verificación

FIN MODULO

MODULO verificarHexadecimal **(TEXTO** opcion**) RETORNA LOGICO**

(*Este módulo verifica que el número ingresado sea Hexadecimal*)

ENTERO contador, i

LOGICO verificación

contador← 0

i ← 0

MIENTRAS (contador < longitud(opcion)) **HACER**

SI ((posición (opción, contador)= 0) OR (posición (opción, contador)= 1) OR) (posición (opción, contador)= 2) OR (posición (opción, contador)= 3) OR (posición (opción, contador)= 4) OR (posición (opción, contador)= 5) OR (posición (opción, contador)= 6) OR (posición (opción, contador)= 7) OR (posición (opción, contador)= 8) OR (posición (opción, contador)= 9) OR (posición (opción, contador)= 'A') OR (posición (opción, contador)= 'B') OR (posición (opción, contador)= 'C') OR (posición (opción, contador)= 'D') OR (posición (opción, contador)= 'E') OR (posición (opción, contador)= 'F') OR (posición (opción, contador)= 'a') OR (posición (opción, contador)= 'b') OR (posición (opción, contador)= 'c') OR (posición (opción, contador)= 'd') OR (posición (opción, contador)= 'e') OR (posición (opción, contador)= 'f')) **HACER**

i ← i+1

FIN SI

contador← contador +1

FIN MIENTRAS

verificación← contador = i

RETORNA verificacion

FIN MODULO

MODULO verificarDecimal **(TEXTO** opcion**) RETORNA LOGICO**

(*Este módulo verifica que el número ingresado sea Decimal*)

ENTERO contador, i

LOGICO verificación

contador ← 0

i← 0

MIENTRAS (i < longitud(opción)) **HACER**

SI ((posición (opción, i) = 0) OR ((posición (opción, i) = 1) OR ((posición (opción, i) = 2) OR ((posición (opción, i) = 3) OR ((posición (opción, i) = 4) OR ((posición (opción, i) = 5) OR ((posición (opción, i) = 6) OR ((posición (opción, i) = 7) OR((posición (opción, i) = 8) OR ((posición (opción, i) = 9)) **ENTONCES**

contador← contador + 1

FIN SI

i ← i +1

FIN MIENTRAS

verificación← contador = i

RETORNA verificación

FIN MODULO

MODULO convertirBinarioHexadecimal **(TEXTO** opcion **) RETORNA TEXTO**

ENTERO resto, cantCuatro

LOGICO verifiacion

TEXTO hexa

resto← longitud(opcion) MOD 4

hexa←" "

verificacion← binarioHexa(cantCuatro, opcion)

SI (verificacion) **ENTONCES**

SI (resto=0) **ENTONCES**

cantCuatro←longitud(opcion)DIV 4

hexa= binarioHexa(cantCuatro, opcion) + hexa

SINO SI (resto<4) **ENTONCES**

SEGÚN (resto) **HACER**

1: opcion← “0”+”0”+”0”+ opción

2: opcion← ”0”+”0”+ opción

3: opcion← ”0”+ opción

FIN SEGÚN

cantCuatro ←longitud(opcion)DIV 4

hexa←binarioHexa(cantCuatro, opcion)+hexa

SINO

ESCRIBIR (“El numero ingresado es un numero binario no Valido”)

FIN SI

RETORNA hexa

FIN MODULO

MODULO convertirBinarioDecimal **(TEXTO** opcion**) RETORNA TEXTO**

(*Este módulo hace la conversión de un numero Binario a un Decimal *)

ENTERO i, b

TEXTO c

REAL numero

LOGICO verificacion

verificacion←verificacionBinario(opcion)

i ← longitud(opcion)

numero ← 0

b← 0

SI (verificacion) **HACER**

MIENTRAS (i > -1) **HACER**

SI (posición(opción, i)= '1') **ENTONCES**

numero ← numero + 1*(posición(2,b))

b←b + 1

SINO SI (posición(opción, i)= '0') **ENTONCES**

numero ← numero + 0*(posición(2,b))

b ← b + 1

FIN SI

i ← i - 1

SINO

ESCRIBIR (“El numero ingresado es un numero binario no VALIDO ”)

FIN SI

c ← " "+ (ENTERO)numero

RETORNA c

FIN MODULO

MODULO convertirDecimalBinario **(TEXTO** opción**) RETORNA TEXTO**

(*Este módulo hace la conversión de un numero Decimal a un Binario*)

TEXTO binario

LOGICO verificacion

ENTERO contador

REAL exponentedecimal, numero

CARÁCTER op

binario ← " "

exponentedecimal ← longitud(opcion) – 1

numero ← 0

contador ← 0

verificacion ← verificarDecimal(opcion)

SI (verificacion) **ENTONCES**

MIENTRAS (contador < longitud(opcion)) **HACER**

op ← posición(opción, contador)

SEGÚN (op) **HACER**

0 : numero ← numero + 0 *(10 potencia(exponentedecimal))

exponentedecimal ← exponentedecimal - 1

1 : numero ← numero + 1 *(10 potencia(exponentedecimal))

exponentedecimal ← exponentedecimal - 1

2 : numero ← numero + 2 *(10 potencia(exponentedecimal))

exponentedecimal ← exponentedecimal - 1

3 : numero ← numero + 3 *(10 potencia(exponentedecimal))

exponentedecimal ← exponentedecimal - 1

4 : numero ← numero + 4 *(10 potencia(exponentedecimal))

exponentedecimal ← exponentedecimal - 1

5 : numero ← numero + 5 *(10 potencia(exponentedecimal))

exponentedecimal ← exponentedecimal - 1

6 : numero ← numero + 6 *(10 potencia(exponentedecimal))

exponentedecimal ← exponentedecimal - 1

7 : numero ← numero + 7 *(10 potencia(exponentedecimal))

```
                    exponentedecimal ← exponentedecimal - 1
                    8 : numero ← numero + 8 *( 10 potencia(exponentedecimal))
                    exponentedecimal ← exponentedecimal - 1
                    9 : numero ← numero + 9 *( 10 potencia(exponentedecimal))
                    exponentedecimal ← exponentedecimal - 1
            FIN SEGÚN
        contador ← contador + 1
        FIN MIENTRAS
        SI (numero > 0) ENTONCES
            MIENTRAS (numero > 0) HACER
                SI (((numero DIV 2 )= 0)) ENTONCES
                    binario ← "0" + binario
                SINO
                    binario← "1" + binario
            numero ← (ENTERO)numero DIV 2
        FIN MIENTRAS
        SINO SI (numero = 0)ENTONCES
            binario ← "0"
        SINO SI (numero < 0 ) ENTONCES
            MIENTRAS (numero < 0) HACER
                SI ((numero DIV 2) = 0) ENTONCES
                    binario← "0" + binario
                SINO
                    binario ← "1" + binario
            numero ← (ENTERO) numero DIV 2
            FIN MIENTRAS
        binario ← "-" + binario
        FIN SI
RETORNA binario
FIN MODULO
MODULO convertirDecimalHexadecimal (TEXTO opción) RETORNA TEXTO
(*Este módulo hace la conversión de un numero Decimal a un Hexadecimal*)
        TEXTO binario, hexa
        LOGICO verificación
        Verificación ← verificarDecimal(opcion)
```

SI (verificación) **ENTONCES**

Binario ← convertirDecimalBinario(opción)

Hexa ←convertirBinarioHexadecimal(opción)

SINO

hexa← “”

FIN SI

RETORNA hexa

FIN MODULO

MODULO convertirHexadecimalBinario **(TEXTO** opcion**) RETORNA TEXTO**

(*Este módulo hace la conversión de un numero Hexadecimal a un Binario*)

TEXTO binario

ENTERO i ←0

LOGICO verificación

verificación ←verificarHexadecimal(opción)

binario ←””

opcion ←aMayuscula(opción)

SI verificación **ENTONCES**

MIENTRAS (i < longitud(opción)) **HACER**

SEGÚN (posición(opción, i) **HACER**

‘0’: binario←binario+”0000”

‘1’: binario←binario+”0001”

‘2’: binario←binario+”0010”

‘3’: binario←binario+”0011”

‘4’: binario←binario+”0100”

‘5’: binario←binario+”0101”

‘6’: binario←binario+”0110”

‘7’: binario←binario+”0111”

‘8’: binario←binario+”1000”

‘9’: binario←binario+”1001”

‘A’: binario←binario+”1010”

‘B’: binario←binario+”1011”

‘C’: binario←binario+”1100”

‘D’: binario←binario+”1101”

‘E’: binario←binario+”1110”

```
                    'F': binario←binario+"1111"
                FIN SEGÚN
                i← i+1
            FIN MIENTRAS
        SINO
            Binario←""
        FIN SI
        RETORNA binario
FIN MODULO

MODULO convertirHexadecimalDecimal (TEXTO opcion) RETORNA TEXTO
(*Este módulo hace la conversión de un numero Hexadecimal a un Decimal*)
        TEXTO binario, decimal
        LOGICO verificación
        verificación←verificarHexadecimal(opción)
        SI (verificación) ENTONCES
            binario←convertirHexadecimalBinario(opción)
            decimal←convertirBinarioDecimal(binario)
        SINO
            decimal←""
        FIN SI
        RETORNA decimal
FIN MODULO

MODULO convertirBinarioComplemento1(TEXTO opción) RETORNA Ø
(*Este módulo hace la conversión de un numero Binario a Complemento a 1*)

        TEXTO complemento
        LOGICO verificación
        ENTERO i ←0
        complemento← ""
        verificación← verificarBinario(opcion)
        SI (verificación) ENTONCES
            MIENTRAS (i<longitud(opcion)) HACER
                SI (posición(opcion,i ) = '0') ENTONCES
```

```
                    complemento ←c omplemento+”1”
                SINO SI (posición(opción,i ) = ‘1’) ENTONCES
                    complemento← complemento+”1”
                FIN SI
            i← i+1
            FIN MIENTRAS
        SINO
            ESCRIBIR(“El numero ingresado es un numero binario no valido.”)
        FIN SI
FIN MODULO

MODULO convertirBinarioComplemento2 (TEXTO opcion) RETORNA Ø
(*Este módulo hace la conversión de un numero Binario a Complemento a 2*)
        TEXTO complemento
        ENTERO i , a , b
        i ←0
        a ←longitud(opción)-1
        b ← a-1
        complementob ← ””
        SI (posición(opción,0) = ‘0’) ENTONCES
            ESCRIBIR (“El binario en complemento a 2 es “,binario)
        SINO SI (posición(opción,0) = ‘1’) ENTONCES
            MIENTRAS (posición(opción,i) = ‘0’) HACER
                SI (posición(opción,i) = ‘0’) ENTONCES
                    complementob ← complementob + “1”
                SINO SI (posición(opción,i) = ‘1’) ENTONCES
                    complementob←complementob+”0”
                FIN SI
            i← +1
            FIN MIENTRAS
            SI (posición(complementob,a) = ‘0’) ENTONCES
                complementob ←subcadena(complementob, 0, a)+”1”
            SINO SI (posición(complementob,a) = ‘0’) ENTONCES
                complementob ←subcadena(complementob, 0, a)+”0”
                MIENTRAS (b>0) HACER
```

```
                    SI (posición(complementob, b)='1') ENTONCES
                        complementob← subcadena(complementob, 0,
                        b)+"0"+subcadena(complementob, a)
                    SINO SI (posición(complementob, b)='0') ENTONCES
                        complementob← subcadena(complementob, 0,
                        b)+"1"+subcadena(complementob, a)
                        B← -1
                    FIN SI
                a← -1
                b← -1
                FIN MIENTRAS
            FIN SI
        FIN SI
FIN MODULO

MODULO binarioHexa (ENTERO cantCuatro, TEXTO opción) RETORNA TEXTO
(*Este módulo hace la conversión de un numero Binario a Hexadecimal*)
        ENTERO c, d
        TEXTO hexa, subs
        hexa←""
        opcion ←opcion + "0"
        c←longitud(opción)
        d←c-5
        MIENTRAS (d>=0) HACER
            subs = subcadena(opción, d, c-1)
            SEGÚN (subs) HACER
                "0000": hexa←"0"+hexa;
                    c←c-4
                    d←d-4
                "0001": hexa←"1"+hexa;
                    c←c-4
                    d←d-4
                "0010": hexa←"2"+hexa;
                    c←c-4
                    d←d-4
                "0011": hexa←"3"+hexa;
```

c←c-4

d←d-4

“0100”: hexa←”4”+hexa;

c←c-4

d←d-4

“0101”: hexa←”5”+hexa;

c←c-4

d←d-4

“0110”: hexa←”6”+hexa;

c←c-4

d←d-4

“0111”: hexa←”7”+hexa;

c←c-4

d←d-4

“1000”: hexa←”8”+hexa;

c←c-4

d←d-4

“1001”: hexa←”9”+hexa;

c←c-4

d←d-4

“1010”: hexa←”A”+hexa;

c←c-4

d←d-4

“1011”: hexa←”B”+hexa;

c←c-4

d←d-4

“1100”: hexa←”C”+hexa;

c←c-4

d←d-4

“1101”: hexa←”D”+hexa;

c←c-4

d←d-4

“1110”: hexa←”E”+hexa;

c←c-4

d←d-4

```
                "1111": hexa←"F"+hexa;
                        c←c-4
                        d←d-4
            FIN SEGÚN
        FIN MIENTRAS
        RETORNA hexa
FIN MODULO
```