

CALCULADORA

Introducción.	2
1º Funciones (Línea 7 a 245)	2
2º menú y llamada a funciones (Línea 255 - 355)	5
2 FUNCIONES	6
Funciones de control	6
def comprueba_negativos(decimal):	6
def valores_posibles_binario(n):	7
def binario(n):	7
Def valores_posibles_decimal(n):	8
def valores_posibles_octal(n):	9
def valores_posibles_hexadecimal(n):	9
Funciones de representación (Ca1/Ca2)	10
def invertir(n):	10
def ca1(número):	10
def ca2_num_neg(numero_de_entrada)	12
Conversiones	15
def binario_decimal(numero_de_entrada)	15
def binario_octal(result_bin_dec):	17
def binario_hexadecimal(result_bin_dec):	18
def octal_decimal(numero_de_entrada):	19
def octal_binario(result_oct_dec):	21
def octal_hexadecimal(result_oct_dec)	22
def hexadecimal_decimal(numero_de_entrada):	23
def hexadecimal_binario(result_hex_dec):	24
def hexadecimal_octal(numero_hex_dec):	25
def formateo_hexadecimal(numero):	26
def decimal_binario(numero_de_entrada):	27
def decimal_octal(numero_de_entrada):	28
def decimal hexadecimal(numero_de_entrada):	29

Introducción.

El nombre del programa es **calculadoraConFuncionesFinal.py**

El programa básicamente cuenta con dos bloques.

1º Funciones (Línea 7 a 245)

Las funciones Están **subdivididas en bloques**:

Los **tres bloques** principales son:

- **Conversiones** → Total de **13 funciones** que donde se realizan todas las conversiones entre sistemas.
- **Complementos** → Total de **3 funciones** para realizar las operaciones de representación de la información en **complementos a1 y a2**
- **Control de entradas** → Total de **5 funciones** para controlar las entradas en las entradas de las funciones

```
#funciones binario*****
def binario_decimal(numero_de_entrada): ...

def binario_octal(result_bin_dec): ...

def binario_hexadecimal(result_bin_dec): ...
def octal_decimal(numero_de_entrada): ...

def octal_binario(result_oct_dec): ...

def octal_hexadecimal(result_bin_dec): ...

#desde_hexadecimal*****
def hexadecimal_decimal(numero_de_entrada): ...

def hexadecimal_binario(result_hex_dec): ...

def hexadecimal_octal(result_hex_dec): ...

def formateo_hexadecimal(numero): ...
```

```
#funciones decimal*****###
> def decimal_binario(numero_de_entrada): ...

> def decimal_octal(numero_de_entrada): ...

> def decimal_hexadecimal(numero_de_entrada): ...

> def comprueba_negativo(decimal): ...

> def invertir(n): ...

> def ca1(numero): ...

> def ca2_num_neg(numero_de_entrada): ...

#-----FUNCIONES PARA CONTROLAR LAS ENTRADAS

> def valores_posibles_binario(n): ...

> def binario(n): ...

> def valores_posibles_decimal(n): ...

> def valores_posibles_octal(n): ...

> def valores_posibles_hexadecimal(n): ...
```

En la parte final del programa se encuentra un menú echo a partir de un **while True**, donde se introduce los datos en el formato que sea y se llama a las funciones.

De forma general en cada nivel se realizan se realizan **tres niveles de comparación.**

```
if eleccion.lower() == "binario_decimal" or eleccion.lower() == "binario":
    if binario(numero_de_entrada) == True:
        if eleccion.lower() == "binario_decimal":
            print(f"{numero_de_entrada} convertido a decimal es igual a {decimal(numero_de_entrada)}")
        elif eleccion.lower() == "binario_octal":
            print(f"{numero_de_entrada} convertido a octal es igual a {oct(numero_de_entrada)}")
        elif eleccion.lower() == "binario_hexadecimal":
            print(f"{numero_de_entrada} convertido a hexadecimal es igual a {hex(numero_de_entrada)}")
    else:
        print("Los valores introducidos no son binarios")
```

- ```
#-----MENÚ Y LLAMADA A FUNCIONES-----
print('-'*60+'\n'+'\t'*2+' CALCULADORA BINARIA\n'+ '-'*60)
while True:
 try: #excepciones
 numero_de_entrada = input("\nIntroduzca un número por teclado un número en el formato que quieras:\n>>> ")
 eleccion = str(input("Indique que función va a realizar\n\n"+ "-"*60+"\n"
 "binario_decimal\t\t"
 "binario_octal\t\t"
 "binario_hexadecimal\t\t"
 "octal_decimal\t\t"
 "octal_binario\t\t"
 "octal_hexadecimal\t\t"
 "hexadecimal_decimal\t\t"
 "hexadecimal_binario\t\t"
 "hexadecimal_octal\t\t"
 "decimal_binario\t\t"
 "decimal_octal\t\t"
 "decimal_hexadecimal\t\t"
 "ca1_desde_binario\t\t"
 "ca1_desde_decimal\t\t"
 "ca2_desde_binario\t\t"
 "ca2_desde_decimal\t\t"
 "fin\t\t"
 "\n\n"))

 # resultados binario
 if eleccion.lower() == "binario_decimal" or eleccion.lower() == "binario_octal" or eleccion.lower() == "binario_hexadecimal": ...
 elif eleccion.lower() == "octal_decimal" or eleccion.lower() == "octal_binario" or eleccion.lower() == "octal_hexadecimal": ...
 elif eleccion.lower() == "hexadecimal_decimal" or eleccion.lower() == "hexadecimal_binario" or eleccion.lower() == "hexadecimal_octal": ...
 elif eleccion.lower() == "decimal_binario" or eleccion.lower() == "decimal_octal" or eleccion.lower() == "decimal_hexadecimal": ...
 elif eleccion.lower() == "ca1_desde_binario": ...

 elif eleccion.lower() == "ca1_desde_decimal": ...

 elif eleccion.lower() == "ca2_desde_binario": ...

 elif eleccion.lower() == "ca2_desde_decimal": ...

 elif eleccion.lower() == "fin": ...

 else:
 print("La acción que ha introducido no es reconocida por el software\n"+ "-"*60)

except ValueError:
 print("Valor introducido no válido\n"+ "-"*60)

except EOFError:
 print("Valor introducido no válido\n"+ "-"*60)
```

2. Destacar en la parte final del programa se ejecutan **dos excepciones** para controlar dos problemas extra que surgieron tras probar mucho el programa

## 2 FUNCIONES

Comentaré cada una de las funciones.

Para seguir el **orden lógico** en el que se hace en las llamadas empezaré con las **funciones encargadas de comprobar los valores, después con los complementos, y por último las conversiones**

### Funciones de control

*def comprueba\_negativos(decimal):*

Es la función **número 14** del programa. Y se utiliza para la **llamada de complementos desde un número decimal**, Se encarga de comprobar como argumento un número decimal de entrada. Si la primera posición es “-” devuelve valores bool True en caso contrario False

```
def comprueba_negativo(decimal):
 '''Función 14. Comprueba si el numero decimal que se está introduciendo es
 if numero_de_entrada[0] == '-':
 return True # Es negativo
 return False # Si no devuelve Falso
```

Llamada de la función:

```
elif eleccion.lower() == "ca2_desde_decimal":
 if comprueba_negativo(numero_de_entrada): # si es negativo
 numero_de_entrada = numero_de_entrada[1:]
 print(ca2_num_neg(decimal_binario(numero_de_entrada)))
 else:
 print(decimal_binario(numero_de_entrada)) # si no hace el decimal_
```

*def valores\_posibles\_binario(n):*

Es la **función 18** se **se llama dentro** de las **funciones ca1 y ca2** .El **valor de entrada es un índice que recorre el** valor de entrada de la función **ca1**. Comprueba si el valor binario de los complementos es correcto.

```
def valores_posibles_binario(n):
 '''Función 18 Comprueba si se han introducido correctamente los números b
 v = ('0','1')
 if n not in v:
 return False # no es un valor binario
 return True # es un valor binario
```

Llamada de la Función:

```
def ca1(numero):
 '''Función 16. Complemento a1 de un numero'''
 resultado = ""
 for i in numero:
 resultado += invertir(i)
 return resultado
```

**destacar** que también se utiliza en el **retorno** de la función **ca2**  
(más adelante comentaré las funciones complemento

```
 break
 return "1" + ca1(numero_de_entrada[0:len(numero_de_entrada)-c]) + resultado
```

*def binario(n):*

Función **número 19** comprueba si el **parámetro de entrada** (Como parámetro de entrada de las funciones de conversión de binario son realmente **valores binario**. Si devuelve **True** se llama a las funciones de conversión. Si devuelve **False**:

```
def binario(n):
 '''Funcion 19. Para para comprobar las
 v = ('0','1')
 for i in n:
 if i not in v:
 return False
 return True
```



Llamada:

```
if eleccion.lower() == "binario_decimal" or eleccion.lower() == "binario_octal" or eleccion.lower() == "binario_hexadecimal":
 if binario(numero_de_entrada) == True:
 if eleccion.lower() == "binario_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {binario_decimal(numero_de_entrada)}")
 elif eleccion.lower() == "binario_octal":
 print(f"{numero_de_entrada} convertido a octal es igual a {binario_octal(binario_decimal(numero_de_entrada))}")
 elif eleccion.lower() == "binario_hexadecimal":
 print(f"{numero_de_entrada} convertido a hexadecimal es igual a {binario_hexadecimal(binario_decimal(numero_de_entrada))}")
 else:
 print("Los valores introducidos no son binarios")
```

*Def valores\_posibles\_decimal(n):*

Es la función **número 20** recorre el número de entrada (**como parámetro recibe el número de entrada**) y si los valores están en la tupla **v** devuelve **True**. Funciona igual que la de binario. Si se cumple **True** se llama a las funciones de **conversiones desde decimal**

```
def valores_posibles_decimal(n):
 '''Función 20. Para comprobar las entradas en decimal'''
 v = ('0', '1', '2', '3', '4', '5', '6', '7', '8', '9')
 for i in numero_de_entrada:
 if i not in v:
 return False # no es un valor decimal
 return True # es un valor decimal
```

Llamada:

```
resultados decimal-----
elif eleccion.lower() == "decimal_binario" or eleccion.lower() == "decimal_octal" or eleccion.lower() == "decimal_hexadecimal":
 if valores_posibles_decimal(numero_de_entrada) == True:
 if eleccion.lower() == "decimal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {decimal_binario(numero_de_entrada)}")
 elif eleccion.lower() == "decimal_octal":
 print(f"{numero_de_entrada} convertido a octal es igual a {decimal_octal(numero_de_entrada)}")
 elif eleccion.lower() == "decimal_hexadecimal":
 print(f"{numero_de_entrada} convertido a hexadecimal es igual a {decimal_hexadecimal(numero_de_entrada)}")
 else:
 print("El numero decimal que has introducido no es decimal")
```

*def valores\_posibles\_octal(n):*

Es la **función 21**. El parámetro de entrada es el ("**numero\_de\_entrada**")  
recorre el número y si el **índice** está dentro de la tupla de valores devuelve **True**. Llama al  
resto de funciones de **de conversiones** desde **octal**.

```
def valores_posibles_octal(n):
 '''Función 21. Para controlar las entradas en octal'''
 v = ('0','1','2','3','4','5','6','7')
 for i in numero_de_entrada:
 if i not in v:
 return False # no es un octal
 return True # es un octal
```

**Llamada:**

```
elif eleccion.lower() == 'octal_decimal' or eleccion.lower() == 'octal_binario' or eleccion.lower() == 'octal_hexadecimal':
 if valores_posibles_octal(numero_de_entrada):
 if eleccion.lower() == "octal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {octal_binario(octal_decimal(numero_de_entrada))}")
 elif eleccion.lower() == "octal_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {octal_decimal(numero_de_entrada)}")
 elif eleccion.lower() == "octal_hexadecimal":
 print(f"{numero_de_entrada} convertido a hexadecimal es igual a {octal_hexadecimal(octal_decimal(numero_de_entrada))}")
 else:
 print("los valores introducidos no son en octal")
```

*def valores\_posibles\_hexadecimal(n):*

Es la **función** número 22. Recibe como entrada el **numero\_de\_entrada** para comprobar si  
es **hexadecimal** recorriendo un índice en el **numero\_de\_entrada** y comparando estos.  
Devuelve valores tipo **bool** si el resultado es **True** se realizan el resto de funciones.

```
def valores_posibles_hexadecimal(n):
 '''Función 22. Para controlar las entradas en hexadecimal'''
 v = ('0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F','a','b','c','d','e','f')
 for i in numero_de_entrada:
 if i not in v:
 return False # no es hexadecimal
 return True # es hexadecimal
```

**Llamada:**

```
elif eleccion.lower() == "hexadecimal_decimal" or eleccion.lower() == "hexadecimal_binario" or eleccion.lower() == "hexadecimal_octal":
 if valores_posibles_hexadecimal(numero_de_entrada) == True:
 if eleccion.lower() == "hexadecimal_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {hexadecimal_decimal(numero_de_entrada)}")
 elif eleccion.lower() == "hexadecimal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {hexadecimal_binario(hexadecimal_decimal(numero_de_entrada))}")
 elif eleccion.lower() == "hexadecimal_octal":
 print(f"{numero_de_entrada} convertido a octal es igual a {hexadecimal_octal(hexadecimal_decimal(numero_de_entrada))}")
 else:
 print("Los valores introducidos no son en hexadecimal")
```



## Funciones de representación (Ca1/Ca2)

*def invertir(n):*

Esta es la **función número 15**, se encarga de a partir de la entrada de de **un índice** cambiar los valores **0** por **1** y **1** por **0** devolviendo el valor cambiado en cada caso

```
def invertir(n):
 '''Función 15. Invierte los valores binarios (similar al operador and)'''
 if n == '0':
 return '1'
 return '0'
```

Se **llama** a esta función en la siguiente y participa junto a **ca1** en la función destinada a realizar el **ca2**

*def ca1(número):*

Esta es la **función 16** recibe como entrada el **numero\_de\_entrada**, y se encarga de hacer de realizar el **complemento a1** Este se ejecuta si se **comprueba previamente que el número es binario** y mediante la función **invertir()** devuelve el **complemento a1** del

```
def ca1(numero):
 '''Función 16. Complemento a1 de un numero'''
 resultado = ""
 for i in numero:
 resultado += invertir(i)
 return resultado
```

**Llamada:**

```
elif eleccion.lower() == "ca1_desde_binario":
 if binario(numero_de_entrada): # si es binario
 print(ca1(numero_de_entrada))
 else:
 print("El numero que has introducido no es binario")
```

**Ejecución:**

```

CALCULADORA BINARIA

Introduzca un número por teclado un número en el formato que quieras:
>>> 888
Indique que función va ha realizar

----- recibe como entrada el número de entrada y se encarga de hacer de realizar e
"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"ca2_desde_decimal"

 PARA SALIR INTRODUCIR "FIN"

>>> ca1_desde_binario
El número que has introducido no es binario

 elif eleccion lower() == "ca1 desde binario":
 binario(numero de entrada): # si es binario
 print(binario(numero de entrada))
 else:
 print("El número que has introducido no es binario")

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"ca2_desde_decimal"

 PARA SALIR INTRODUCIR "FIN"

>>> ca1_desde_binario
100110001
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

`def ca2_num_neg(numero_de_entrada)`

```
def ca2_num_neg(numero_de_entrada):
 '''Función 17. Hace el ca2 de un numero'''
 # Primero entra en un bucle para recorrer el contenido de la lista en sentido inverso apuntando
 # Los valores en una nueva cadena; cuando llega a 1 entra y se para
 # Mediante un contador cuento las posiciones que recorre esta dentro de la cadena
 # gracias a las posiciones contadas en el bucle anterior y mediante len cogo la rebanada que
 # ha sido tratada por el bucle y le hago el cal(cambiar valores 1 por 0 y viceversa)
 # esto se lo sumo al resultado cadena con el final y lo retorno de la función
 c = 0
 resultado = ""
 for i in reversed(numero_de_entrada):
 c += 1
 if valores_posibles_binario(i) == False:
 return 'Los valores son erroneos'
 elif i == '0':
 resultado = i + "" + resultado
 elif i == '1':
 resultado = i + "" + resultado
 break
 return "1" + cal(numero_de_entrada[0:len(numero_de_entrada)-c]) + resultado
```

Función **número 17** se encarga de **realizar el ca2** recibe el **numero\_de\_entrada**. Este lo puede recibir de dos formas distintas si **viene de de un número decimal** comprueba **si es negativo** de ser así recibirá la **conversión a binario** del número sin el "-". **si es binario** recibe este sin más. En caso de ser un número **decimal positivo** realizará la **conversión a binario natural**. En los casos anteriores entra en la **función** y retorna usando la **función c1** el número en **binario** con el complemento

**Llamada:**

```
elif eleccion.lower() == "ca2_desde_decimal":
 if comprueba_negativo(numero_de_entrada): # si es negativo
 numero_de_entrada = numero_de_entrada[1:]
 print(ca2_num_neg(decimal_binario(numero_de_entrada)))
 else:
 print(decimal_binario(numero_de_entrada)) # si no hace el decimal_biario
```

**Ejecución:**

```

CALCULADORA BINARIA
elif eleccion.lower() == "decimal hexadecimal":
 print(f"(numero_de_entrada) convertido a hexadecimal es igual a {ca2_desde_decimal}")
 print("El numero decimal que has introducido no es decimal")
Introduzca un número por teclado un número en el formato que quieras:
>>> 78rr
Indique que función va a realizar

'binario_decimal' "binario_octal"
'binario_hexadecimal' "octal_decimal"
'octal_binario' "octal_hexadecimal"
'hexadecimal_decimal' "hexadecimal_octal"
'binario_binario' "decimal_binario"
'decimal_octal' "decimal_hexadecimal"
'ca1_desde_binario' "ca2_desde_binario"
"ca2_desde_decimal"
 print("El numero que has introducido no es binario")
 PARA SALIR INTRODUCIR "FIN"
elif eleccion.lower() == "ca2_desde_decimal":
 prueba_negativo(numero_de_entrada): # si es negativo
 valor_introducido no válido de entrada = numero_de_entrada[1:]
 print(ca1_nos_neg(decimal_binario(numero_de_entrada)))
else:
 Introduzca un número por teclado un número en el formato que quieras:
 >>> -788
 Indique que función va a realizar

 elif eleccion.lower() == "fin":
 'binario_decimal' print("binario_octal")
 'binario_hexadecimal' "octal_decimal"
 'octal_binario' "octal_hexadecimal"
 'hexadecimal_decimal' "hexadecimal_octal"
 'binario_binario' print("decimal_binario")
 'decimal_octal' "decimal_hexadecimal"
 'ca1_desde_binario' "ca2_desde_binario"
 "ca2_desde_decimal"
 print("Valor introducido no válido\n"+"-"*60)
 PARA SALIR INTRODUCIR "FIN"
 print("Valor introducido no válido\n"+"-"*60)

>>> ca2_desde_decimal
10011101100

Introduzca un número por teclado un número en el formato que quieras:
>>> "binario_binario" "decimal_binario"
```



```
Introduzca un número por teclado un número en el formato que quieras:
>>> 1233
Indique que función va a realizar
número en el formato que quieras:

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"ca1_desde_decimal" "ca2_desde_decimal"
"ca1_desde_hexadecimal"
"binario" PARA SALIR INTRODUCE "FIN"

>>> ca2_desde_decimal
10011010001
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

```
>>> ca2_desde_decimal
1001101100
Introduzca un número por teclado un número en el formato que quieras:
>>> 01100111
Indique que función va a realizar
número en el formato que quieras:

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"ca1_desde_decimal" "ca2_desde_decimal"
"ca1_desde_hexadecimal"
"binario" PARA SALIR INTRODUCE "FIN"

>>> ca2_desde_binario
110011001
Introduzca un número por teclado un número en el formato que quieras:
>>>
```



```
Introduzca un número por teclado un número en el formato que quieras:
>>> 988
Indique que función va a realizar
número en el formato que quieras:

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"ca1_desde_octal" "ca2_desde_decimal"

PARA SALIR INTRODUCE "FIN"
>>> ca2_desde_binario
Los valores son erroneos
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

## Conversiones

*def binario\_decimal(numero\_de\_entrada)*

Esta es la **Función 1** recibe el **numero\_de\_entrada** después de comprobar si es binario.  
retorna la conversión de este a **decimal**

```
def binario_decimal(numero_de_entrada):
 '''Función 1 única para pasar binario a decimal'''
 contador = 0
 sumando = 2**(len(numero_de_entrada)-1)
 for a in numero_de_entrada:
 contador += sumando * int(a)
 sumando //= 2
 result_bin_dec = contador
 return result_bin_dec
```

## Llamada

```
resultados binario
if eleccion.lower() == "binario_decimal" or eleccion.lower() == "binario_octal" or eleccion.lower() == "binario_hexadecimal":
 if binario(numero_de_entrada) == True:
 if eleccion.lower() == "binario_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {binario_decimal(numero_de_entrada)}")
```

Ejecución:

```

 CALCULADORA BINARIA

numero en el formato que quieras:
Introduzca un número por teclado un número en el formato que quieras:
>>> 1001
Indique que función va ha realizar

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

 PARA SALIR INTRODUCE "FIN"

>>> binario_decimal
1001 convertido a decimal es igual a 9

Introduzca un número por teclado un número en el formato que quieras:
>>> 78
Indique que función va ha realizar

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

 PARA SALIR INTRODUCE "FIN"

>>> binario_decimal
Los valores introducidos no son binarios

Introduzca un número por teclado un número en el formato que quieras:
>>>
```

*def binario\_octal(result\_bin\_dec):*

Es la **Función 2** recibe **recibe el resultado de la función *binario\_decimal()*** y convierte este a **octal** retorna el valor **octal**

```
✓ def binario_octal(result_bin_dec):
 '''Función 2 llamo a la variable anterior para pasar a octal desde desde
 el resiltado en binario.'''
 numero_bin_oct = result_bin_dec
 lista = []
 ✓ while numero_bin_oct >=1:
 lista.insert(0,numero_bin_oct%8)
 numero_bin_oct //=8
 result_bin_oct = "".join(str(i) for i in lista)
 return result_bin_oct
```

**Llamada:**

```
if binario(numero_de_entrada) == True:

 if eleccion.lower() == "binario_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {binario_decimal(numero_de_entrada)}")

 elif eleccion.lower() == "binario_octal":
 print(f"{numero_de_entrada} convertido a octal es igual a {binario_octal(binario_decimal(numero_de_entrada))}")
```





### Llamada

```
print(f"{numero_de_entrada} convertido a octal es igual a {binario_octal(binario_decimal(numero_de_entrada))}")

elif eleccion.lower() == "binario_hexadecimal":
 print(f"{numero_de_entrada} convertido a hexadecimal es igual a {binario_hexadecimal(binario_decimal(numero_de_entrada))}")
else:
 print("Los valores introducidos no son binarios")
```

### Ejecución

```
Introduzca un número por teclado un número en el formato que quieras:
>>> 10011
Indique que función va a realizar
a octal es igual a {binario_octal(binario_decimal(numero_de_entrada))}")

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" igual a {bin"octal_hexadecimal" decimal(numero_de_entrada))"}
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"ca2_desde_decimal"
lon.lower() == 'octal_binario' or eleccion.lower() == 'octal_hexadecimal':
 PARA SALIR INTRODUCE "FIN"

>>> binario_hexadecimal binario(octal_decimal(numero_de_entrada))
10011 convertido a hexadecimal es igual a 13
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

*def octal\_decimal(numero\_de\_entrada):*

Función **número 4** después de comprobarse que es octal recibe como parámetro **numero\_de\_entrada** retorna el valor en **octal**

```
def octal_decimal(numero_de_entrada):
 '''Función 4. única para pasar de octal a decimal'''
 contador = 0
 sumando = 8**(len(numero_de_entrada)-1)
 for i in numero_de_entrada:
 contador += sumando *int(i)
 sumando //=8
 result_oct_dec = contador
 return result_oct_dec
```

### Llamada

```
elif eleccion.lower() == 'octal_decimal' or eleccion.lower() == 'octal_binario' or eleccion.lower() == 'octal_hexadecimal':
 if valores_posibles_octal(numero_de_entrada):
 if eleccion.lower() == "octal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {octal_binario(octal_decimal(numero_de_entrada))}")

 elif eleccion.lower() == "octal_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {octal_decimal(numero_de_entrada)}")
```



## Ejecución

```
Introduzca un número por teclado un número en el formato que quieras:
>>> 988
Indique que función va a realizar
a decimal es igual a (octal_decimal(numero_de_entrada))")

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

eleccion.lower() == "hexadecimal_binario" or eleccion.lower() == "hexadecimal_octal":
 (ada) == True:
 """
 >>> octal_decimal(hexadecimal_decimal(numero_de_entrada))
 los valores introducidos no son en octal
 """
 """
 Introduzca un número por teclado un número en el formato que quieras:
 >>> 777
 Indique que función va a realizar
 a octal es igual a (hexadecimal_octal(hexadecimal_decimal(numero_de_entrada)))")

 "binario_decimal" "binario_octal"
 "binario_hexadecimal" "octal_decimal"
 "octal_binario" "octal_hexadecimal"
 "hexadecimal_decimal" "hexadecimal_octal"
 "binario_binario" "decimal_binario"
 "decimal_octal" "decimal_hexadecimal"
 "ca1_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

 == True:
 PARA SALIR INTRODUCE "FIN"

>>> octal_decimal
777 convertido a decimal es igual a 511

Introduzca un número por teclado un número en el formato que quieras:
>>>
```

`def octal_binario(result_oct_dec):`

Función **número 5** después de comprobar **recibe el resultado de la función `octal_decimal(numero_de_entrada)`** convierte el valor decimal a binario y retorna el valor en binario

```
def octal_binario(result_oct_dec):
 '''Función 5. Llama a la función anterior para pasa desde el
 resultado decimal anterior a binario para así
 hacer la conversión octal a binario'''
 num_oct_bin = result_oct_dec
 lista = []
 while num_oct_bin >= 1:
 lista.insert(0, num_oct_bin % 2)
 num_oct_bin //= 2
 result_oct_bin = "".join(str(i) for i in lista)
 return result_oct_bin
```

### Llamada

```
if valores_posibles_octal(numero_de_entrada):
 if eleccion.lower() == "octal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {octal_binario(octal_decimal(numero_de_entrada))}")
```

### Ejecución

```

a binario es igual a CALCULADORA BINARIA

Introduzca un número por teclado un número en el formato que quieras:
>>> 274 es igual a {octal_decimal(numero_de_entrada)}
Indique que función va a realizar
1:
a hexadecimal es igual a {octal_hexadecimal(decimal(numero_de_entrada))}
"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" hexadecimal ca2_desde_binario.lower() == "hexadecimal_octal":
ada) == True: "ca2_desde_decimal"
1:
PARA SALIR INTRODUCE "FIN"
a decimal es igual a {hexadecimal_decimal(numero_de_entrada)}

>>> octal_binario
274 convertido a binario es igual a 10111100 decimal(numero_de_entrada)}
Introduzca un número por teclado un número en el formato que quieras:
>>> octal es igual a {hexadecimal_octal(hexadecimal_decimal(numero_de_entrada))}
hexadecimal")
```

`def octal_hexadecimal(result_oct_dec)`

Función **número 6** recibe el retorno de la función `octal_decimal(numero_de_entrada)` esta la convierte a decimal y retorna el valor en hexadecimal usado la función `formateo_hexadecimal(número)`

```
def octal_hexadecimal(result_bin_dec):
 '''Función 6. Llama a resultado decimal anterior a hexadecimal para
 así hacer la conversión octal a hexadecimal'''
 num_oct_hex = result_bin_dec
 lista = []
 while num_oct_hex >= 1:
 lista.insert(0, num_oct_hex % 16)
 num_oct_hex //= 16
 result_oct_hex = formateo_hexadecimal(lista)
 return result_oct_hex
```

### Llamada

```
print(f"{numero_de_entrada} convertido a binario es igual a {hexadecimal_binario(hexadecimal_decimal(numero_de_entrada))}")

elif eleccion.lower() == "hexadecimal_octal":
 print(f"{numero_de_entrada} convertido a octal es igual a {hexadecimal_octal(hexadecimal_decimal(numero_de_entrada))}")
se:
```

### Ejecución

```
Introduzca un número por teclado un número en el formato que quieras:
>>> 6777
Indique que función va a realizar
a binario es igual a (hexadecimal_binario(hexadecimal_decimal(numero_de_entrada)))

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

cion.lower() == "decimal_hexadecimal":
== True:
>>> octal_hexadecimal
6777 convertido a hexadecimal es igual a DFF
Introduzca un número por teclado un número en el formato que quieras:
>>>
```



*def hexadecimal\_decimal(numero\_de\_entrada):*

Función **número 7** una vez comprobado recibe como entrada **numero\_de\_entrada** y realiza la conversión a **hexadecimal**

```
def hexadecimal_decimal(numero_de_entrada):
 '''Función 7. Única conversión hexadecimal a decimal'''
 result_hex_dec = int(numero_de_entrada, 16)
 return result_hex_dec
```

**Llamada**

```
elif eleccion.lower() == "hexadecimal_decimal" or eleccion.lower() == "hexadecimal_binario" or eleccion.lower() == "hexadecimal_octal":
 if valores_posibles_hexadecimal(numero_de_entrada) == True:
 if eleccion.lower() == "hexadecimal_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {hexadecimal_decimal(numero_de_entrada)}")
```

**Ejecución**

```
Introduzca un número por teclado un número en el formato que quieras:
>>> 98op
Indique que función va a realizar (numero_de_entrada))")

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
eleccion.lower() == "ca2_desde_decimal" or eleccion.lower() == "hexadecimal_octal":
ada) == True:
 PARA SALIR INTRODUCE "FIN"
!":
a decimal es igual a {hexadecimal_decimal(numero_de_entrada)}")

>>> hexadecimal_decimal
Los valores introducidos no son en hexadecimal
a binario es igual a {hexadecimal_binario(hexadecimal_decimal(numero_de_entrada))}")
Introduzca un número por teclado un número en el formato que quieras:
>>> 98ef
Indique que función va a realizar (numero_de_entrada))")

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario" == "decimal_hexadecimal":
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"ca2_desde_decimal"

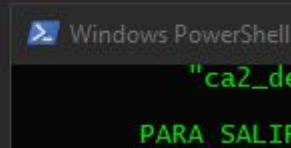
 PARA SALIR INTRODUCE "FIN"

>>> hexadecimal_decimal
98ef convertido a decimal es igual a 39151
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

*def hexadecimal\_binario(result\_hex\_dec):*

Función **número 8**. después de comprobar que **numero\_de\_entrada** es hexadecimal recibe el retorno de la función **hexadecimal\_decimal(numero\_de\_entrada)** y la realiza la conversión a **binario**.

```
def hexadecimal_binario(result_hex_dec):
 '''Función 8. pasa desde el resultado decimal anterior a hexadecimal para
 así hacer la conversión octal a hexadecimal'''
 num_hex_bin = result_hex_dec
 lista = []
 while num_hex_bin >= 1:
 lista.insert(0, num_hex_bin % 2)
 num_hex_bin //= 2
 result_hex_dec = "".join(str(i) for i in lista)
 return result_hex_dec
```



### Llamada

```
if valores_posibles_hexadecimal(numero_de_entrada) == True:
 if eleccion.lower() == "hexadecimal_decimal":
 print(f"{numero_de_entrada} convertido a decimal es igual a {hexadecimal_decimal(numero_de_entrada)}")

 elif eleccion.lower() == "hexadecimal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {hexadecimal_binario(hexadecimal_decimal(numero_de_entrada))}")
```

### Ejecución

```
introduzca un número por teclado un número en el formato que quieras:
>> 98ef
indique que función va a realizar
n.lower() == 'octal_binario' or eleccion.lower() == 'octal_hexadecimal':

binario_decimal" "binario_octal"
binario_hexadecimal" "octal_decimal"
octal_binario" a (octal_b" "octal_hexadecimal"(numero_de_entrada))")
hexadecimal_decimal" "hexadecimal_octal"
binario_binario" "decimal_binario"
decimal_octal" a (octal_d" "decimal_hexadecimal")
ca1_desde_binario" "ca2_desde_binario"
":
"ca2_desde_decimal"
a hexadecimal es igual a (octal_hexadecimal(octal_decimal(numero_de_entrada)))")
PARA SALIR INTRODUCE "FIN"

octal')
>> hexadecimal_binario
8ef convertido a binario es igual a 1001100011101111

introduzca un número por teclado un número en el formato que quieras:
>> 11111
indique que función va a realizar
```



`def hexadecimal_octal(numero_hex_dec):`

Función **número 9** utiliza el retorno de la función `hexadecimal_decimal(numero_de_entrada)` y realiza la conversión a octal

```
def hexadecimal_octal(result_hex_dec):
 '''Función 9. Pasa desde el resultado decimal anterior a hexadecimal para
 así hacer la conversión octal a hexadecimal'''
 num_hex_octal = result_hex_dec
 lista = []
 while num_hex_octal >= 1:
 lista.insert(0, num_hex_octal % 8)
 num_hex_octal //= 8
 result_hex_dec = "".join(str(i) for i in lista)
 return result_hex_dec
```

### Llamada

```
elif eleccion.lower() == "hexadecimal_octal":
 print(f"{numero_de_entrada} convertido a octal es igual a {hexadecimal_octal(hexadecimal_decimal(numero_de_entrada))}")
else:
 print("Los valores introducidos no son en hexadecimal")
```

### Ejecución

```

a hexadecimal es igual a {decimal_hexadecimal(numero_de_entrada)}")
CALCULADORA BINARIA

no no es decimal')
Introduzca un número por teclado un número en el formato que quieras:
>>> 677f
Indique que función va ha realizar
(Comprobación se hace en la función)

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"cal_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

PARA SALIR INTRODUCE "FIN"

>>> hexadecimal_octal
677f convertido a octal es igual a 63577
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

*def formateo\_hexadecimal(numero):*

Función **número 11** recibe como entrada de argumento una lista con los resultados de convertir distintos sistemas a **hexadecimal** para **retornar** como **string** **formateados mediante bucle for y comparaciones**

```
def formateo_hexadecimal(numero):
 '''Función 10. formatea los numeos en letras para las conversiones en las que participe hexadecimal'''
 resultado = ""
 for i in numero:
 if i == 10:
 resultado += 'A'
 elif i == 11:
 resultado += 'B'
 elif i == 12:
 resultado += 'C'
 elif i == 13:
 resultado += 'D'
 elif i == 14:
 resultado += 'E'
 elif i == 15:
 resultado += 'F'
 else:
 resultado += str(i)
 return resultado
```

**Llamada** (Se utiliza en varias funciones, pondré como **ejemplo binario\_hexadecimal**)

```
def binario_hexadecimal(result_bin_dec):
 '''Función 3 llamo a la variable anterior para pasar a hexa desde desde
 el resultado en decimal.'''
 numero_bin_hex = result_bin_dec
 lista = []
 while numero_bin_hex >= 1:
 lista.insert(0,numero_bin_hex%16)
 numero_bin_hex //=16
 result_bin_hex = formateo_hexadecimal(lista)
 return result_bin_hex
```

## Ejecución

```
Introduzca un número por teclado un número en el formato que quieras:
>>> 1111
Indique que función va ha realizar

'binario_decimal' "binario_octal"
'binario_hexadecimal' "octal_decimal"
'octal_binario' "octal_hexadecimal"
'hexadecimal_decimal" "hexadecimal_octal"
'binario_binario' "decimal_binario"
'decimal_octal" "decimal_hexadecimal"
'cal_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

 PARA SALIR INTRODUCE "FIN"

>>> binario_hexadecimal
1111 convertido a hexadecimal es igual a F
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

*def decimal\_binario(numero\_de\_entrada):*

Función **número 11** después de comprobar que es un número decimal recibe como entrada **numero\_de\_entrada** realiza la conversión y devuelve este en **binario**

```
def decimal_binario(numero_de_entrada):
 '''Función 11. Única para convertit decimal a binario
 También se utiliza para hacer el complemento a2 de los numeros añadiendole
 0 en la función de abajo'''
 numero_decimal = int(numero_de_entrada)
 lista = []
 while numero_decimal >= 1:
 lista.insert(0, numero_decimal % 2)
 numero_decimal //= 2
 result_dec_bin = "".join(str(i) for i in lista)
 return result_dec_bin
```

Llamada

```
elif eleccion.lower() == "decimal_binario" or eleccion.lower() == "decimal_octal" or eleccion.lower() == "decimal_hexadecimal":
 if valores_posibles_decimal(numero_de_entrada) == True:
 if eleccion.lower() == "decimal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {decimal_binario(numero_de_entrada)}")
```

Ejecución

```
Introduzca un número por teclado un número en el formato que quieras:
>>> 1024
Indique que función va a realizar

'binario_decimal' 'binario_octal'
'binario_hexadecimal' 'octal_decimal'
'octal_binario' 'octal_hexadecimal'
'hexadecimal_decimal' 'hexadecimal_octal'
'binario_binario' 'decimal_binario'
'decimal_octal' 'decimal_hexadecimal'
'ca1_desde_binario' 'ca2_desde_binario'
'ca2_desde_binario' 'ca2_desde_decimal'
o negativo:
PARA SALIR INTRODUCE "FIN"

>>> decimal_binario
1024 convertido a binario es igual a 100000000000
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

*def decimal\_octal(numero\_de\_entrada):*

Función **número 12** después de comprobar que es un número decimal recibe como entrada **numero\_de\_entrada** realiza la conversión y devuelve este en **octal**

```
def decimal_octal(numero_de_entrada):
 '''Función 12. Única para convertir de decimal a octal'''
 numero_decimal = int(numero_de_entrada)
 lista = []
 while numero_decimal >= 1:
 lista.insert(0, numero_decimal % 8)
 numero_decimal //= 8
 result_dec_oct = "".join(str(i) for i in lista)
 return result_dec_oct
```

### Llamada

```
elif eleccion.lower() == "decimal_binario" or eleccion.lower() == "decimal_octal" or eleccion.lower() == "decimal_hexadecimal":

 if valores_posibles_decimal(numero_de_entrada) == True:

 if eleccion.lower() == "decimal_binario":
 print(f"{numero_de_entrada} convertido a binario es igual a {decimal_binario(numero_de_entrada)}")

 elif eleccion.lower() == "decimal_octal":
 print(f"{numero_de_entrada} convertido a octal es igual a {decimal_octal(numero_de_entrada)}")
```

### Ejecución



```
>>> 28d
Indique que función va a realizar
a binario es igual a {hexadecimal_binario(hexadecimal_decimal(numero_de_entrada))}

"binario_decimal" "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" {hexadecimal_binario(hexadecimal_decimal(numero_de_entrada))}
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

comprobación se hace en la función
PARA SALIR INTRODUCE "FIN"

== True:
>>> decimal_octal
El número decimal que has introducido no es decimal
a binario es igual a {decimal_binario(numero_de_entrada)}
Introduzca un número por teclado un número en el formato que quieras:
>>> 1024
Indique que función va a realizar
a octal es igual a {decimal_octal(numero_de_entrada)}

"binario_decimal" "binario_octal"
"binario_hexadecimal" {decimal_binario(numero_de_entrada)}
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
 "ca2_desde_decimal"

comprobación se hace en la función
PARA SALIR INTRODUCE "FIN"

>>> decimal_octal
1024 convertido a octal es igual a 2000
Introduzca un número por teclado un número en el formato que quieras:
>>>
```

*def decimal\_hexadecimal(numero\_de\_entrada):*

Función **número 13** después de comprobar que es un número decimal, recibe como argumento **numero\_de\_entrada** convierte en hexadecimal y usando **formateo\_hexadecimal** retorna el valor en hexadecimal

```
def decimal_hexadecimal(numero_de_entrada):
 '''Función 13. Única para convertir de decimal a hexadecimal'''
 numero_decimal = int(numero_de_entrada)
 lista = []
 while numero_decimal >= 1:
 lista.insert(0, numero_decimal % 16)
 numero_decimal //= 16
 result_dec_hex = formateo_hexadecimal(lista)
 return result_dec_hex
```



## Llamada

```
elif eleccion.lower() == "decimal_hexadecimal":
 print(f"{numero_de_entrada} convertido a hexadecimal es igual a {decimal_hexadecimal(numero_de_entrada)}")
else:
 print("El numero decimal que has introducido no es decimal")
```

## Ejecución

```
Introduzca un número por teclado un número en el formato que quieras:
>>> 99999
Indique que función va a realizar

"binario_decimal" en "binario_octal"
"binario_hexadecimal" "octal_decimal"
"octal_binario" "octal_hexadecimal"
"hexadecimal_decimal" "hexadecimal_octal"
"binario_binario" "decimal_binario"
"decimal_octal" "decimal_hexadecimal"
"ca1_desde_binario" "ca2_desde_binario"
"binario") "ca2_desde_decimal"

PARA SALIR INTRODUCE "FIN"

>>> decimal_hexadecimal
99999 convertido a hexadecimal es igual a 1869F
Introduzca un número por teclado un número en el formato que quieras:
>>>
```