Matemáticas enfocadas a la programación

Aprende y mejora unidad 1

David Santiago Calderón Barrios
Julio Eduardo Duran Coronel
Felipe Alejandro Guillen Aguilera
Jhon Sebastian Zarate Hernandez

Agenda

- Descripción del problema
- Explicación del programa
- Resultados obtenidos
- Conclusiones y recomendaciones

Descripción del problema

- Utilizar lenguaje de programación Python para encontrar el número hexadecimal de cualquier numero decimal.
- Para números decimales pares se busca también el número binario y para impares el numero octal.
- Adicionalmente se busca proveer una breve descripción de lo que se va a buscar.

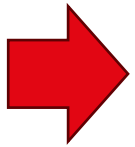
Explicación del programa

```
# Pedir número al usuario
num_decimal = int(input("¡Hola! Ingresa un número decimal: "))
```

En esta primera línea del código se le pide al usuario ingresar un numero decimal, através de la función "**input**". Dicho numero se asigna a la variable "**num_decimal**", la cual se convierte a su vez en un numero entero por medio del operador "**int**".

```
# Identificar si el número ingresado es par o no
if num_decimal % 2 == 0:
    var_par="par"
else:
    var_par="impar"
```

En estas líneas se busca encontrar si el numero ingresado es par o no se utilizando el condicional "if". Se evalúa si el residuo de la división sobre 2 del valor "num_decimal" es igual a 0 por medio de la expresión "% 2 == 0", de esta manera, cuando es verdadero determina que el número es par y se le asigna la descripción de "par" a una nueva variable "var_par. En caso contrario, el condicional "else" asigna el valor "impar" a la variable "var_par".



```
# Asignar una descripción a un  numero par, como binario y a un impar como octal

if var_par == "par":
    var_des="binario"
else:
    var_des="octal"
```

De acuerdo con la clasificación anterior, en las siguientes líneas se identifica si el resultado del código va a traer un valor binario u octal. Lo anterior, por medio nuevamente del condicional "if". En este caso si la variable creada "var_par" toma un valor "par". Le asigna a la nueva variable "var_des", la descripción de "binario", en caso contrario (else) le asigna el valor "octal". Mas adelante, esto nos va a describir si el resultado esperado es Binario u Octal, dependiendo del valor ingresado.

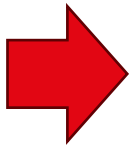
Explicación del programa

```
# Mostrar el número ingresado
print(f"El número ingresado es {num_decimal}, lo cual corresponde a un número {var_par}. \nDe acuerdo a esto, se van a encontrar los numeros hexadecimal y {var_des}.")
```

Con la función **print** se imprime un texto descriptivo que coloque de manera explicita el número seleccionado (variable **num_decimal**), el tipo de numero al que corresponde (par, impar) a través de la variable **var_par** y el resultado que se va a mostrar tras el desarrollo del código (hexadecimal y binario u octal) utilizando la variable **var_des**.

```
# Convertir a binario, octal y hexadecimal
n_binario = bin(num_decimal)[2:]
n_octal = oct(num_decimal)[2:]
n_hexadecimal = hex(num_decimal)[2:]
```

Para convertir el numero decimal a binario, octal y hexadecimal se utilizan las funciones bin, oct y hex, respectivamente. Al resultado de la función se eliminan los primeros 2 prefijos a través del comando "[2:]", con el fin de que nos traiga únicamente los resultados numéricos, omitiendo la clasificación de la función sobre el numero.



```
# Mostrar resultados
print("\nResultados")
print(f"Hexadecimal: {n_hexadecimal}")
```

En las siguientes líneas de código se le pide que nos imprima (**"print"**) la palabra "Resultados" luego de una línea de espacio insertada con **"\n"**. En la siguiente línea le pedimos que nos imprima los resultados junto a su descripción. Para incorporar línea de texto junto al valor numérico resultante se utiliza el comando **"f"** (f"Hexadecimal : {n_hexadecimal}"). En este caso los {} identifican la variable dentro de la cadena de texto resultante.

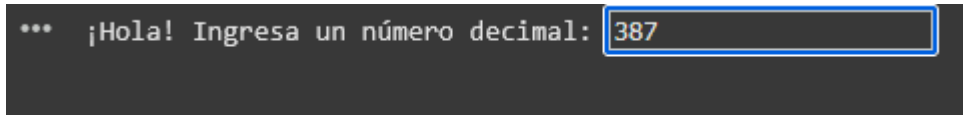
Explicación del programa

```
# Condición para mostrar binario o octal según si el número es par o impar
if num_decimal % 2 == 0:
    print(f"Binario: {n_binario}")
else:
    print(f"Octal: {n_octal}")
```

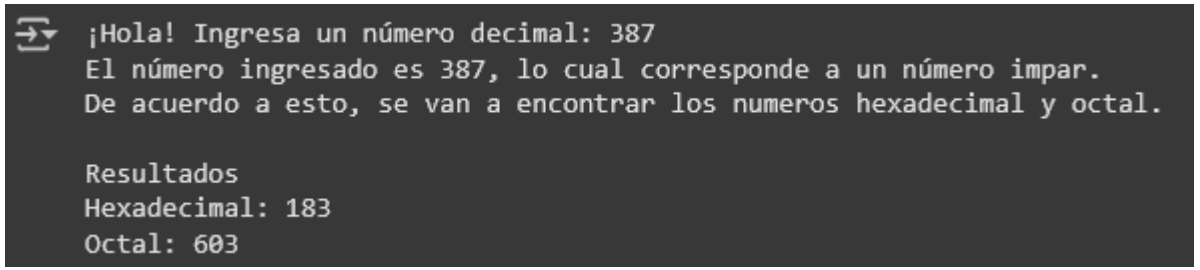
Por ultimo usamos nuevamente el condicional **"if"** para evaluar si el numero ingresado es par, mediante la expresión "**num_decimal % 2 == 0"**, donde evalúa si el residuo de la operación es igual a cero. En caso que el numero evaluado sea par, la expresión es verdadera, en cuyo caso el condicional imprimiría (**print**) el resultado binario **(f"binario: {n_binario}"),** de lo contrario **"else"** imprimiría el resultado octal **(f"octal: {n_octal}")**, para números es impares.

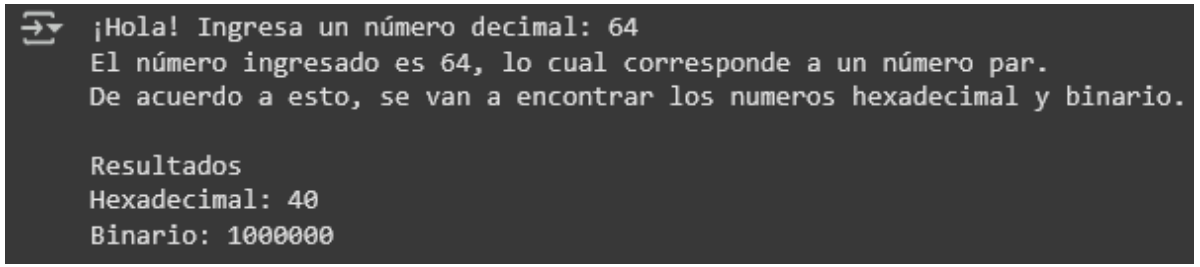
Resultados obtenidos

Al principio del código se solicita al usuario ingresar un numero decimal en el espacio otorgado.



El número ingresado corresponde al que se utilizara para la evaluaciones de las demás condiciones del código.





Al finalizar, nos muestra las los resultados junto a una breve explicación de lo que se muestra. Arrojando los resultados numéricos en las ultimas 3 líneas. Los resultados incorporan opciones distintas para números pares e impares.

Conclusiones y recomendaciones

Conclusiones

1.0 Comprensión de los sistemas numéricos

Durante el desarrollo del ejercicio, aprendimos a convertir números del sistema decimal a binario y octal usando Python. Esto me permitió entender mejor cómo las computadoras representan los datos y cómo funcionan internamente estos sistemas numéricos.

1.1 Uso de funciones en Python

Descubrimos que Python tiene funciones nativas como **bin()** y **oct()** que facilitan estas conversiones. Sin embargo, también entendimos la importancia de manipular los resultados para eliminar los prefijos 0b y 0o, haciendo que la salida sea más clara para el estudiante.

1.2 Interacción con el usuario

Aprendimos a solicitar datos de entrada con **input()** y a formatear los resultados para que la información sea comprensible. Esta parte fue clave, ya que un programa eficiente no solo debe dar respuestas correctas, sino que también debe ser fácil de usar.

Dificultades Encontradas

2.1 Comprender la estructura del código

Al inicio, nos costó entender cómo Python manejaba los sistemas numéricos y cómo funcionan las funciones **bin()** y **oct()**. Tuvimos que investigar y probar diferentes maneras de mostrar los resultados sin los prefijos.

2.2 Errores en la conversión de datos

En algunos intentos, olvidamos que **input()** devuelve valores como cadenas de texto, por lo que tuvimos que convertir la entrada en un número entero con **int()**. Sin esta conversión, el código no funcionaba correctamente.

2.3 Formateo de los resultados

Al principio, el programa imprimía los valores con prefijos **(0b y 0o)**, lo cual podría confundir a alguien que no está familiarizado con estos sistemas. Encontramos la solución utilizando el método de segmentación de cadenas **([2:])**, lo que nos permitió eliminar los caracteres adicionales y mostrar únicamente el valor numérico en el formato deseado.

Oportunidades de mejora del programa en el futuro

3.1 Mejorar la interacción con el estudiante

Se podría agregar un menú interactivo donde el estudiante elija a qué sistema numérico desea convertir su número. También sería útil incluir mensajes de error si el usuario ingresa datos no válidos.

3.2 Implementar una interfaz gráfica

En vez de usar solo texto en una consola, podríamos crear una ventana con botones, cuadros de texto y etiquetas para que la interacción sea más visual y amigable utilizando bibliotecas como Tkinter, lo que haría que el programa sea más intuitivo y fácil de usar.

3.3 Optimización del código

Aunque el código actual es simple, se podría mejorar usando funciones personalizadas, lo que haría el programa más modular y fácil de expandir en el futuro.

Conclusión Final

Este ejercicio nos permitió comprender la importancia de los sistemas numéricos en la programación y cómo manipularlos en Python. Aunque encontramos algunas dificultades, logramos resolverlas con investigación y práctica. En el futuro, mejoraremos este programa agregando más funciones y optimizando su estructura para hacerlo más robusto y fácil de usar.

Bibliografía