

Introducción

Somos el grupo número 2, compuesto por Batista Federico, Benitez Rodrigo, Bertotti Franco, Bovio Jose y Caballero Myriam, el tema que seleccionamos para abordar es el traspaso de números decimales a binarios, octales y hexadecimales.

A continuación, mostraremos como desarrollamos dos códigos en Python, seleccionamos uno para resolver la actividad, y como utilizamos la ayuda de la IA en ciertas instancias.

Comenzamos desde la selección del tema, siguiendo por el desarrollo del código, y por último la formulación de la documentación con la grabación del video que fue subido a YouTube link video. [Video en Youtube de la explicacion](https://www.youtube.com/watch?v=yivWs8EsCLU)

<https://www.youtube.com/watch?v=yivWs8EsCLU> (opcional)

ALGORITMO 1

```
numero = int(input("Ingresa un numero entero: "))
print("Este programa pasa un numero decimal a binario, octal, o
hexadecinal")
print("Selecciona una opcion")
print("decimal a binario:1")
print("decimal a octal: 2")
print("decimal a hexadecimal: 3")
opcion = int(input("Ingresa una opcion: "))
acumulador = ""
hexa = "0123456789ABCDEF"
cociente = numero
match opcion:
    case 1:
        b = 2
    case 2:
        b = 8
    case 3:
        b = 16
    case _:
        b = 0
if b != 0:
    while cociente != 0:
        resto = cociente %
        b
        cociente = cociente // b
        acumulador = hexa [resto] + acumulador
    print(f"El numero decimal {numero}, es equivalente al numero de
base {opcion}: {acumulador}")
else:
    print("Las opciones posibles son: 1, 2, o 3. Vuelva a intentarlo")
```

ALGORITMO 2

```
numero = int(input("Ingresa un numero entero positivo: "))  
print("Este programa pasa un numero decimal a binario, octal, o  
hexadecimal")  
print("Selecciona una opcion")  
print("decimal a binario:1")  
print("decimal a octal: 2")  
print("decimal a hexadecimal: 3")
```

```
opcion = int(input("Ingrese una opcion: "))
match opcion:
    case 1:
        #AL NUMERO ENTERO LO CONVIERTE A BINARIO
        potencia = 1
        acumulador = 0
        cociente = numero
        while cociente != 0:
            resto = cociente % 2
            cociente = cociente // 2
            acumulador = resto * potencia + acumulador
            potencia = potencia * 10
        print(f"El numero decimal {numero}, es equivalente al numero
binario {acumulador}")
    case 2:
        #AL NUMERO ENTERO LO CONVIERTE A OCTAL
        potencia = 1
        acumulador = 0
        cociente = numero
        while cociente != 0:
            resto = cociente % 8
            cociente = cociente // 8
            acumulador = resto * potencia + acumulador
            potencia = potencia * 10
        print(f"El numero decimal {numero}, es equivalente al numero
octal {acumulador}")
    case 3:
        #AL NUMERO ENTERO LO CONVIERTE A HEXADECIMAL
        acumulador = ""
        hexa = "0123456789ABCDEF"
        cociente = numero
        while cociente != 0:
            resto = cociente % 16
            cociente = cociente // 16
            acumulador = hexa [resto] + acumulador
        print(f"El numero decimal {numero}, es equivalente al numero
hexadecimal {acumulador}")
    case _:
        print("no corresponde a una opcion, vuelve a intentar")
print("Fin del programa")
```

Elección del Algoritmo para el video

Se le pide a ChatGPT, que compare los 2 algoritmos y esto es lo que obtenemos:

◆ ¿Qué es más eficiente?

- Algoritmo 1 es más eficiente y profesional porque no repite código. Usa una sola lógica para todas las bases.
- Algoritmo 2 es más largo pero más educativo para aprender, porque permite ver el proceso de cada base por separado.

◆ ¿Qué algoritmo conviene usar?

- Si estás aprendiendo, el Algoritmo 2 te ayuda a entender mejor cómo funciona cada conversión.
- Si estás programando algo más avanzado o profesional, el Algoritmo 1 es más limpio, reutilizable y elegante.

Conclusión

Para aprender 

Para eficiencia 

Usa el Algoritmo 2 para ver cómo funciona cada base.

Usa el Algoritmo 1 si querés un código más corto y versátil.

Elegimos el algoritmo 2 por ser mejor desde el punto de vista pedagógico, para entender de forma más fácil la conversión de decimal a binario, octal, o hexadecimal.

Diálogos creados con la IA ChatGpt de OPENAI

Como grupo tomamos la decisión de consultarle a este modelo de IA un dialogo para elaborar el video, primero creamos un prompt básico, lo más conciso posible. Luego de que la IA nos confirma y nos solicita el código, se lo enviamos y a partir de eso nos devuelve un dialogo.

ChatGPT ▾

Somos alumnos de programación, somos 3 personas que tenemos que describir este código en un video, necesito que generes diálogos de forma tal que podamos explicarlo, se bien claro con las funciones de python

¡Perfecto! Para ayudarte mejor, necesito que me compartas el código que tienen que explicar. Una vez que lo tenga, puedo generar un guion dividido para que cada uno de ustedes tenga una parte clara para decir en el video, explicando cada función y estructura del código en lenguaje simple y entendible.

¿Podés copiar y pegar el código acá?

ChatGPT ▾

¡Perfecto! Gracias por pasar el código completo. Como son tres personas y tienen que explicarlo en un video, les preparé un **guion claro, en partes**, para que cada uno explique una sección del código de forma entendible. Lo divido así:

Guion para 3 personas - Explicación del conversor de base

Persona 1 – Entrada de datos y menú de opciones

"Hola, somos estudiantes de programación, y vamos a explicar un programa en Python que convierte números decimales a binario, octal o hexadecimal."

"El programa comienza pidiendo al usuario que ingrese un número entero positivo con `input()`. Ese valor se guarda en la variable `numero`, y se convierte a entero usando `int()`."

"Después, el programa verifica si el número es negativo usando un `if`. Si lo es, muestra un mensaje de error. Si no, muestra un menú con tres opciones: convertir a binario, octal o hexadecimal. El usuario elige una opción escribiendo un número del 1 al 3."

"Esa opción se guarda en la variable `opcion`, y luego se usa un bloque `match` para elegir qué conversión hacer, según el número que el usuario eligió."

ChatGPT ▾

Persona 2 – Conversión a binario y octal

"Si el usuario elige la opción 1, el programa convierte el número decimal a binario. Para eso se usan tres variables: `potencia`, `acumulador` y `cociente`. La conversión se hace en un bucle `while` que se repite hasta que el cociente llegue a cero."

"Dentro del bucle, se calcula el resto de dividir el cociente por 2, que es el sistema binario. Ese resto se multiplica por la `potencia`, que arranca en 1 y se multiplica por 10 en cada vuelta. Así se va formando el número binario en la variable `acumulador`, que se va armando al revés."

"Para la opción 2, el proceso es muy parecido, pero en vez de dividir por 2, se divide por 8, porque estamos convirtiendo a sistema octal. Todo lo demás funciona igual, solo cambia el número base."

Persona 3 – Conversión a hexadecimal y cierre

"En la opción 3, se convierte a hexadecimal, que usa 16 símbolos: del 0 al 9, y de la A a la F. Por eso, el programa usa una cadena llamada `hexa` que contiene esos caracteres."

"En este caso, el `acumulador` es una cadena de texto vacía. En cada vuelta del bucle, se calcula el resto de dividir por 16, y se usa ese resto como índice para buscar el carácter correcto en la cadena `hexa`. Luego, ese carácter se va agregando al inicio del `acumulador`, para formar el número hexadecimal completo."

"Si el usuario elige una opción que no está en el menú, el programa entra en el `case _:` y muestra un mensaje de error."

"Al final, se imprime un mensaje que dice 'Fin del programa'. Y eso es todo: así funciona este conversor de números en Python."

✓ Extras (opcional para agregar)

Si quieren hacerlo más dinámico o simpático en el video, pueden sumar comentarios como:

- "Este código es útil para entender cómo funcionan los sistemas de numeración."
- "¡Y lo mejor es que lo hacemos sin usar funciones predefinidas como `bin()`, `oct()` o `hex()`!"
- "Podés probar el código con distintos números para ver cómo cambia la conversión."

¿Querés que lo pase a un formato de guion tipo teatro o presentación más formal (con nombres o roles)? ¿O querés que lo hagamos más informal y conversado como charla entre amigos?

Luego las 3 personas haremos una puesta en común, necesito además de lo otro, un diálogo para cada uno hablando de porque fue mejor usar `match` que usar condicionales comunes, y otra cosa que observes

¡Genial! Vamos a sumar esa **puesta en común final**, con un pequeño diálogo para cada uno sobre **por qué se usó `match` en lugar de condicionales comunes** como `if/elif`, y además les agrego una **observación extra** para cada uno, así cada quien tiene una reflexión distinta que aporta valor a la explicación.

ChatGPT ▾

PUESTA EN COMÚN FINAL – Diálogo entre los 3

Persona 1 – Ventaja de `match` y claridad

"Algo que nos pareció importante es que se usó `match` en lugar de varios `if` o `elif`. Esto hace que el código sea mucho más claro cuando tenemos varias opciones posibles, como en este caso, donde hay tres tipos de conversión. Con `match`, cada caso se ve más ordenado y fácil de leer."

"Además, me di cuenta de que el programa también usa validación al principio: si el número ingresado es negativo, directamente no sigue. Eso está bueno porque evita errores más adelante."

Persona 2 – `match` como estructura moderna y mantenible

"Sí, y además `match` es una estructura más moderna que se incorporó en Python 3.10. Sirve mucho cuando hay que tomar decisiones según un valor específico, como un menú de opciones. Hace que el código sea más mantenible, porque si después queremos agregar una opción nueva, solo sumamos otro `case`, sin tocar el resto."

"Y otra cosa que noté es que en la conversión a binario y octal, no se usaron cadenas de texto, sino cálculos con potencias de 10. Eso es interesante porque simula cómo se forma el número, cifra por cifra, sin depender de funciones avanzadas."

ChatGPT ▾

Persona 3 – Seguridad, orden y uso de estructuras

"Exacto, y también me gusta que el `match` tiene un `case _:` que actúa como un 'por defecto', como un `else`. Eso hace que el programa sea más seguro, porque si alguien escribe una opción que no existe, no se rompe nada, simplemente se muestra un mensaje de error."

"Otra cosa que observé es que se usan diferentes tipos de datos según la base: números enteros para binario y octal, y cadenas de texto para hexadecimal. Eso muestra cómo en Python elegimos la estructura según lo que queremos representar."

OPCIONAL – Cierre grupal:

"En resumen, este código muestra cómo convertir un número decimal a distintas bases usando estructuras de control modernas y operaciones básicas. Además de resolver el problema, es un buen ejemplo de escritura clara y lógica."

"¡Esperamos que les haya servido esta explicación!"

¿Querés que también lo pase todo junto como guion final completo (intro + explicación + puesta en común), listo para leerlo tipo presentación? ¿O te gustaría que le dé un tono más natural tipo charla espontánea?