Link github:

https://github.com/MauricioCifuentes2000/arq_proy.git

Codigo .py:

```
#binario a hexa (enteros y fracciones)
#complemento restringido de binario
#decimal a hexadecimal (fracciones y periodicidad)
import sys


def binary_to_hexadecimal(binary_str):
    """
    Convierte un número binario (con parte entera y fraccionaria) a hexadecimal.
    """
    if '.' in binary_str:
        integer_part, fractional_part = binary_str.split('.')
    else:
        integer_part, fractional_part = binary_str, ''

    # Conversión de la parte entera
    hex_integer = hex(int(integer_part, 2))[2:].upper() if integer_part else '0'

    # Conversión de la parte fraccionaria
    hex_fractional = ''
    if fractional_part:
        # Convertir la fracción binaria a decimal
        fractional_decimal = 0
        for i, digit in enumerate(fractional_part, 1):
            fractional_decimal += int(digit) * (2 ** -i)
```

```
        # Convertir la fracción decimal a hexadecimal
        fractional_decimal *= 16
        for _ in range(10):  # Limitar a 10 dígitos hexadecimales para evitar infinitas repeticiones
            digit = int(fractional_decimal)
            hex_fractional += hex(digit)[2:].upper()
            fractional_decimal = (fractional_decimal - digit) * 16
            if fractional_decimal == 0:
                break

    if hex_fractional:
        return f"{hex_integer}.{hex_fractional}"
    else:
        return hex_integer

def restricted_complement(binary_str):
    """
    Calcula el complemento restringido de un número binario.
    Suposición: El primer bit es el bit de signo y no se invierte.
    """
    if not all(c in '01' for c in binary_str):
        return "Error: La entrada no es un número binario válido."

    if len(binary_str) == 0:
        return "Error: Entrada vacía."

    # El primer bit (bit de signo) permanece igual
    sign_bit = binary_str[0]
    # Invertir los demás bits
    complemented_bits = ''.join('1' if bit == '0' else '0' for bit in binary_str[1:])
```

```
    return sign_bit + complemented_bits


def decimal_to_hexadecimal(decimal_str):
    """
    Convierte un número decimal (con parte entera y fraccionaria) a hexadecimal,
    mostrando periodicidad si existe.
    """
    if '.' in decimal_str:
        integer_part, fractional_part = decimal_str.split('.')
    else:
        integer_part, fractional_part = decimal_str, ''

    # Conversión de la parte entera
    hex_integer = hex(int(integer_part))[2:].upper() if integer_part else '0'

    # Conversión de la parte fraccionaria
    if fractional_part:
        fractional_decimal = float("0." + fractional_part)
        seen = {}
        hex_fractional = ''
        repeating_start = -1

        for i in range(20):  # Limitar a 20 iteraciones para detectar repeticiones
            fractional_decimal *= 16
            digit = int(fractional_decimal)
            hex_digit = hex(digit)[2:].upper()
            hex_fractional += hex_digit
            fractional_decimal -= digit
```

```
            # Detectar periodicidad
            if fractional_decimal in seen:
                repeating_start = seen[fractional_decimal]
                break
            seen[fractional_decimal] = i + 1

            if fractional_decimal == 0:
                break

        if repeating_start != -1:
            non_repeating = hex_fractional[:repeating_start]
            repeating = hex_fractional[repeating_start:]
            hex_fractional = f"{non_repeating}({repeating})"

        return f"{hex_integer}.{hex_fractional}"
    else:
        return hex_integer

def display_menu():
    """
    Muestra el menú de opciones al usuario.
    """
    print("\n===== Menú de Operaciones =====")
    print("1. Conversión de Binario a Hexadecimal")
    print("2. Obtener el Complemento Restringido de un Número Binario")
    print("3. Conversión de Decimal a Hexadecimal")
    print("4. Salir")
    print("================================")
```

```
def main():
    while True:
        display_menu()
        choice = input("Selecciona una opción (1-4): ").strip()

        if choice == '1':
            binary_input = input("Introduce el número binario (ejemplo: 1010.101): ").strip()
            result = binary_to_hexadecimal(binary_input)
            print(f"Hexadecimal: {result}")

        elif choice == '2':
            binary_input = input("Introduce el número binario para obtener su complemento
restringido: ").strip()
            result = restricted_complement(binary_input)
            print(f"Complemento Restringido: {result}")

        elif choice == '3':
            decimal_input = input("Introduce el número decimal (ejemplo: 10.625): ").strip()
            try:
                # Validar entrada decimal
                float(decimal_input)
                result = decimal_to_hexadecimal(decimal_input)
                print(f"Hexadecimal: {result}")
            except ValueError:
                print("Error: Entrada no es un número decimal válido.")

        elif choice == '4':
            print("¡Gracias por usar el programa! Hasta luego.")
            sys.exit()
```

```
        else:
            print("Opción inválida. Por favor, elige una opción entre 1 y 4.")
            continue

        # Preguntar si el usuario desea realizar otra operación
        while True:
            again = input("¿Deseas realizar otra operación? (s/n): ").strip().lower()
            if again == 's':
                break
            elif again == 'n':
                print("¡Gracias por usar el programa! Hasta luego.")
                sys.exit()
            else:
                print("Entrada inválida. Por favor, responde con 's' o 'n'.")

if __name__ == "__main__":
    main()
```