Number System Converter - Program Documentation

Program Name: Number_System_Converter
Author: Owaya Augustine
Version: 0.1
Release Date: October 2025

Purpose

This program converts a number from one numeral base (ranging between 2 and 16) to another base, including fractional parts. It supports bases such as Binary (2), Octal (8), Decimal (10), and Hexadecimal (16)

Function Definition

def convert_n(num_str, from_base, to_base):

Parameters

- num_str (str): The number to convert, can include a fractional part (e.g., '101.11').
- from_base (int): The base of the input number (2–16).
- to_base (int): The target base (2–16).

Return Value

Returns a string representing the converted number in the target base.

Error Handling

Raises ValueError if:

- Bases are not between 2 and 16.
- Input is empty.
- Input contains invalid digits for the specified base.

Conversion Steps

Step 1 – Validate Inputs

- Ensure both bases are between 2 and 16.
- Strip whitespace, convert input to uppercase.
- Validate digits against the base.

Step 2 – Split Integer and Fractional Parts

- If the number contains ., it is split into integer_part and fraction_part.
- Otherwise, the entire number is treated as integer_part.

Step 3 – Convert Input Number to Decimal

- Integer part: int(integer_part, from_base)
- Fractional part: calculated using positional values: sum(d_i / base^i).

Step 4 – Convert Decimal to Target Base

- **(a) Integer part**: Divide by target base, collect remainders, and reverse them.
- **(b) Fractional part**: Multiply repeatedly by target base and collect integer portions.

Step 5 – Combine Integer and Fractional Parts

- If there is a fractional part, join with a decimal point.
- Otherwise, return integer part only.

Example Test Cases

```
print(convert_n("1001.111", 2, 10)) # Binary → Decimal => 9.875
 print(convert_n("9.875", 10, 2)) # Decimal → Binary => 1001.111
 print(convert_n("1A.3", 16, 10)) # Hexadecimal → Decimal => 26.1875
```

Entry Point

```
if __name__ == '__main__': print(convert_n("1001.111", 2, 10)) # Test case
```

Notes

- Fractional precision is limited to 10 digits to prevent infinite loops.
- The function works for all bases between 2 and 16.
- Manual fractional conversion is used for accuracy.