Recursive Base Conversion

Given a base 10 number and a target base, convert the number to the target base using a **recursive approach**.

Important: Your solution **MUST** use recursion. Any non-recursive solution will receive zero points from the judge, regardless of correctness.

Input Format

Multiple lines, each containing two integers b and n:

- b: target base ($2 \leq b < 10$)
- n: base 10 number to convert ($0 \leq n \leq 10^6$)

Output Format

For each query, return a string representing the number n in base b

Constraints

- $2 \leq b < 10$ (target base)
- $0 \leq n \leq 10^6$ (base 10 number)
- Each test case will have at least 10 queries

Sample Input

```
2 10
8 255
3 27
```

Sample Output

```
1010
377
1000
```

Explanation

- $10_{10} = 1010_2$ (binary): convert 10 to base 2
- $255_{10} = 377_8$ (octal): convert 255 to base 8
- $27_{10} = 1000_3$ (base 3): convert 27 to base 3

Implementation

Goal: Fill in the following function using **recursion only**:

```
def recur_base_convert(b: int, n: int) -> str:
    ...
    return ...

if __name__ == "__main__":
    # You can test anything inside this block and can send it to grader
    # The grader will use only the function that you have implemented
    # !!! DO NOT write anything outside this block
    print(recur_base_convert(2, 10))  # Should print "1010"
    print(recur_base_convert(8, 255))  # Should print "377"
```

WARNING: The judge will automatically check if your solution uses recursion. Any non-recursive solution (using loops, built-in functions like `bin()`, `oct()`, `hex()`, etc.) will receive **zero points** regardless of correctness.