20 Python Programs Using Recursion (Without Comprehension)

1. Factorial of a number

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)
```

2. Sum of first n natural numbers

```
def sum_natural(n):
    if n == 0:
        return 0
    else:
        return n + sum_natural(n-1)
```

3. Sum of first n odd numbers

```
def sum_odd(n):
    if n == 1:
        return 1
    else:
        return (2 * n - 1) + sum_odd(n - 1)
```

4. Print odd numbers from 1 to n

```
def print_odds(n):
    if n == 0:
        return
    print_odds(n - 1)
    if n % 2 != 0:
        print(n)
```

5. Fibonacci number (nth term)

```
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)
```

6. GCD of two numbers

```
def gcd(a, b):
    if b == 0:
        return a
    else:
        return gcd(b, a % b)
```

7. LCM using GCD

20 Python Programs Using Recursion (Without Comprehension)

```
def lcm(a, b):
    return abs(a * b) // gcd(a, b)
```

8. Reverse digits of a number

```
def reverse_number(n, result=0):
    if n == 0:
        return result
    else:
        return reverse_number(n // 10, result * 10 + n % 10)
```

9. Check if a number is palindrome

```
def is_palindrome(n):
    return n == reverse_number(n)
```

10. Sum of digits of a number

```
def sum_digits(n):
    if n == 0:
        return 0
    else:
        return n % 10 + sum_digits(n // 10)
```

11. Reverse a string

```
def reverse_string(s):
    if len(s) <= 1:
        return s
    else:
        return reverse_string(s[1:]) + s[0]
```

12. Check if string is palindrome

```
def is_palindrome_str(s):
    if len(s) <= 1:
        return True
    if s[0] != s[-1]:
        return False
    return is_palindrome_str(s[1:-1])
```

13. Count vowels in a string

```
def count_vowels(s):
    if s == "":
        return 0
    if s[0].lower() in 'aeiou':
        return 1 + count_vowels(s[1:])
    else:
        return count_vowels(s[1:])
```

20 Python Programs Using Recursion (Without Comprehension)

14. Sum of elements in a list

```
def sum_list(lst):
    if not lst:
        return 0
    else:
        return lst[0] + sum_list(lst[1:])
```

15. Find maximum in a list

```
def max_list(lst):
    if len(lst) == 1:
        return lst[0]
    else:
        max_rest = max_list(lst[1:])
        if lst[0] > max_rest:
            return lst[0]
        else:
            return max_rest
```

16. Count even numbers in a list

```
def count_even(lst):
    if not lst:
        return 0
    if lst[0] % 2 == 0:
        return 1 + count_even(lst[1:])
    else:
        return count_even(lst[1:])
```

17. Power of a number (a^b)

```
def power(a, b):
    if b == 0:
        return 1
    else:
        return a * power(a, b - 1)
```

18. Print numbers from n to 1

```
def print_desc(n):
    if n == 0:
        return
    print(n)
    print_desc(n - 1)
```

19. Print first n even numbers

```
def print_even(n, current=2):
    if n == 0:
        return
```

```
    print(current)
    print_even(n - 1, current + 2)
```

20. Check if array is sorted

```
def is_sorted(arr):
    if len(arr) <= 1:
        return True
    if arr[0] > arr[1]:
        return False
    return is_sorted(arr[1:])
```