Day 18

Recursion in Python:

What is Recursion?

Recursion is when a function calls itself to solve a problem. A big problem is broken into smaller versions of the same problem.

Why do we use Recursion?

- To solve problems that repeat the same logic
- Useful for:
  - factorial
  - Fibonacci
  - tree / file traversal
- Makes code short and clean (for some problems)

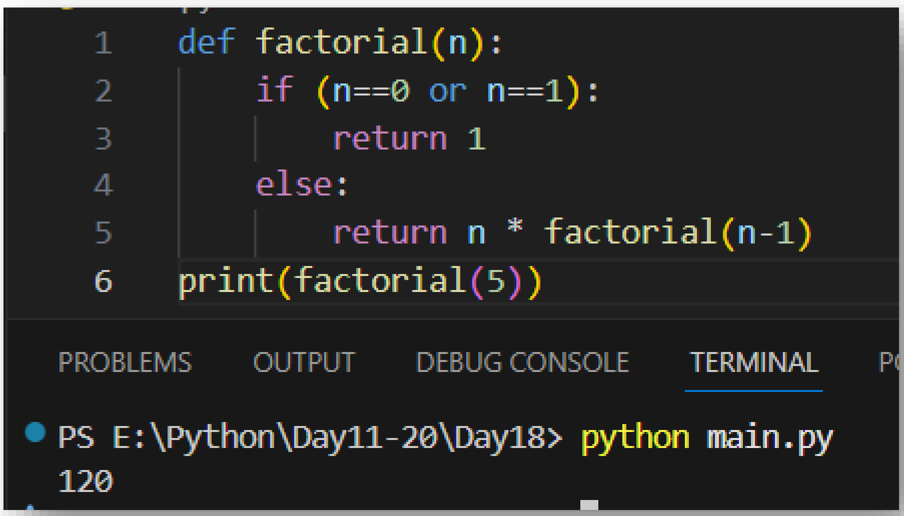
Two Important Parts of Recursion (VERY IMPORTANT)

Every recursive function must have:

- Base Case: Condition where recursion stops
- Recursive Case: Function calls itself

Note: Without base case → infinite recursion → error

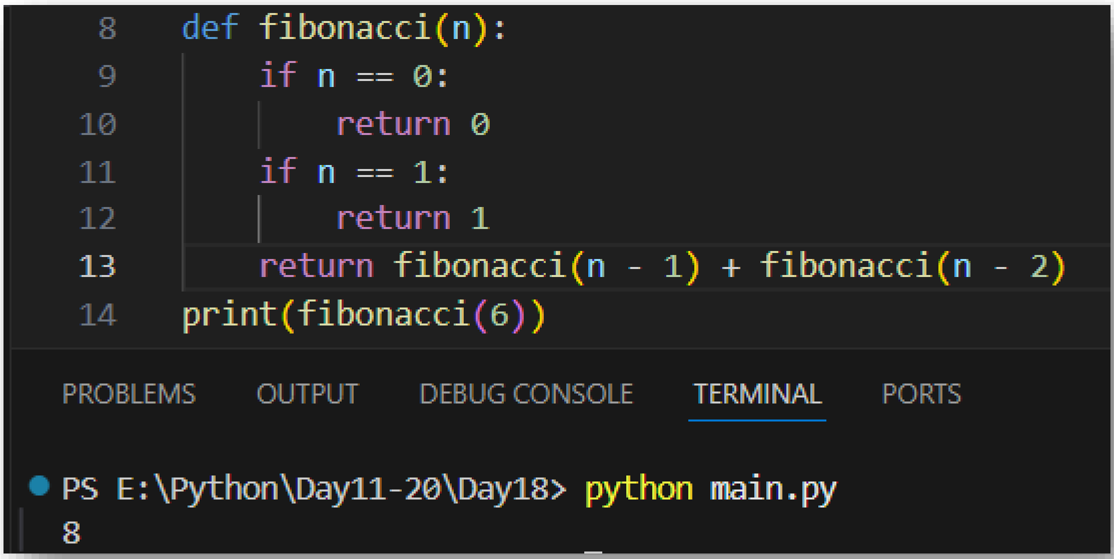
Example: a basic example of recursion -> factorial.



```python
def factorial(n):
    if (n==0 or n==1):
        return 1
    else:
        return n * factorial(n-1)
print(factorial(5))
```

```
PS E:\Python\Day11-20\Day18> python main.py
120
```

Example: Fibonacci using the recursion.



```python
def fibonacci(n):
    if n == 0:
        return 0
    if n == 1:
        return 1
    return fibonacci(n - 1) + fibonacci(n - 2)
print(fibonacci(6))
```

```
PS E:\Python\Day11-20\Day18> python main.py
8
```

Summary:

- Recursion = function calling itself
- Needs base case to stop
- Breaks problem into smaller parts
- Useful for factorial, Fibonacci, trees
- Loops are better for large inputs

--The End--