VIDEO 14 : RECURSIVE FUNCTIONS

A recursive function is a function that calls itself. You may ask yourself, why would you ever want to do that? Actually certain problems can be solved more easily through recursion.

Every recursive function must contain a condition that stops the process of calling the function to execute. Then we break down solving a problem by by performing multiple simple calculations repetitively versus writing one large block of code. Take calculating a factorial as an example.

Calculating a Factorial

Calculating factorials is commonly done with a recursive function 3! = 3 * 2 * 1

CODE

```
def factorial(num):

    # Every recursive function must contain a condition
    # when it ceases to call itself
    if num <= 1:
        return 1
    else:

        result = num * factorial(num - 1)
        return result

print(factorial(4))
```

How it works

1st : result = 4 * factorial(3) = 4 * 6 = 24
2nd : result = 3 * factorial(2) = 3 * 2 = 6
3rd : result = 2 * factorial(1) = 2 * 1 = 2

To solve this problem we work down the ladder of calculations up until the factorial function is passed the value of 1. When that occurs during the 3rd pass we can complete our 1st calculation getting the result of 2. We then move the 2 up the ladder to solve the next calculation (3 * 2 = 6). Then the 6 moves up to finish the calculation (4 * 6 = 24)

Python Problem for you to Solve

To test what you have learned I now want you to calculate Fibonacci numbers.

To calculate Fibonacci numbers we sum the 2 previous values to calculate the next item in the list like this 1, 1, 2, 3, 5, 8 ...

The Fibonacci sequence is defined by:
Fn = Fn-1 + Fn-2
Where F0 = 0 and F1 = 1

Here is a sample run through to help

print(fib(3))

1st : result = fib(2) + fib(1) : 2 + 1
2nd : result = (fib(1) + fib(0)) + (fib(0)) : 1 + 0
3rd : result = fib(2) + fib(1)

print(fib(4))

1st : result = fib(3) + fib(2) : 3 + 2
2nd : result = (fib(2) + fib(1)) + (fib(1) + fib(0)) : 2 + 1
3rd : result = (fib(1) + fib(0)) + fib(0) : 1 + 0

Give it a try. The goal is to get you to think in new ways and to understand the final result.

Solution

```
def fib(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:

        result = fib(n-1) + fib(n-2)
        return result

print(fib(3))

print(fib(4))
```

2nd Python Problem for you to Solve

Previously we generated 1 number in the Fibonacci sequence. This time ask the user to define how many numbers they want and display them

Remember the formula for calculating the Fibonacci sequence is
Fn = Fn-1 + Fn-2
Where F0 = 0 and F1 = 1

Here is sample output you should aim for

```
How many Fibonacci values should be found : 30
1
1
2
3
5
All Done
```

You'll use the same function above with a while loop for output.

Solution

```
def fib(num):
    if num == 0:
        return 0
    elif num == 1:
        return 1
    else:
        result = fib(num - 1) + fib(num - 2)
        return result

num_fib_vals = int(input("How many Fibonacci values should be found : "))

i = 1

# While i is less then the number of values requested
# continue to find more
while i < num_fib_vals:

    # Call the fib()
    fib_val = fib(i)

    print(fib_val)

    i += 1

print("All Done")
```

I hope you enjoyed that tutorial. In the next video I'll teach you how to read and write files and I'll cover tuples as an added bonus.