Unknown date | Chaitanya Singh

Python Recursion

A function is said to be a **recursive** if it calls itself. For example, lets say we have a function `abc()` and in the body of `abc()` there is a call to the `abc()`.

Python example of Recursion

In this example we are defining a **user-defined function** `factorial()`. This function finds the factorial of a number by calling itself repeatedly until the **base case**(We will discuss more about base case later, after this example) is reached.

```
# Example of recursion in Python to
# find the factorial of a given number

def factorial(num):
"""This function calls itself to find
the factorial of a number"""

if num == 1:
return 1
else:
return (num * factorial(num - 1))


num = 5
print("Factorial of", num, "is: ", factorial(num))
```

Output:

```
Factorial of 5 is: 120
```

Lets see what happens in the above example:

```
factorial(5) returns 5 * factorial(5-1)
i.e. 5 * factorial(4)
|___5*4*factorial(3)
|___5*4*3*factorial(2)
|___5*4*3*2*factorial(1)
```

Note: factorial(1) is a base case for which we already know the value of factorial. The base case is defined in the body of function with this code:

```
if num == 1:
return 1
```

What is a base case in recursion

When working with recursion, we should define a base case for which we already know the answer. In the above example we are finding factorial of an integer number and we already know that the factorial of 1 is 1 so this is our base case.

Each **successive recursive call** to the function should bring it closer to the base case, which is exactly what we are doing in above example.

We use base case in recursive function so that the function stops calling itself when the base case is reached. Without the base case, the function would keep calling itself indefinitely.

Why use recursion in programming?

We use recursion to break a big problem in small problems and those small problems into further smaller problems and so on. At the end the solutions of all the smaller subproblems are collectively helps in finding the solution of the big main problem.

Advantages of recursion

Recursion makes our program:
1. Easier to write.
2. Readable – Code is easier to read and understand.
3. Reduce the lines of code – It takes less lines of code to solve a problem using recursion.

Disadvantages of recursion

1. Not all problems can be solved using recursion.
2. If you don’t define the base case then the code would run indefinitely.
3. Debugging is difficult in recursive functions as the function is calling itself in a loop and it is hard to understand which call is causing the issue.
4. Memory overhead – Call to the recursive function is not memory efficient.

‹ Previous Next ›