Tutorial Link https://codequotient.com/tutorials/Recursion - Order of Execution/5a0149a2cbb2fe34b7775053

TUTORIAL

Recursion - Order of Execution

Chapter

1. Recursion - Order of Execution

Topics

1.3 Code Implementation

A function can implement recursion in two ways,

1. Make the recursive call first, let the call executes and end and then do something.
2. Do something, and then make a recursive call and let the call execute and end.

These two orders will describe how the evaluation of statements will be done. For example, consider the following two recursive functions: -

Function1: -

```
void recFunc1(int num)
{
        if (num < 4)
        recFunc1(num + 1);
        printf("%d\n", num);
}
```

Function2: -

```
void recFunc2(int num)
{
        printf("%d\n", num);
        if (num < 4)
        recFunc2(num + 1);
}
```

Function1 is making the recursive call first, and after the recursive call finishes, it will print the num, whereas function2 will print the num first, then will make the recursive call. So order for function1 is: -

```
recFunc1(0)
        recFunc1(0+1)
                recFunc1(1+1)
                        recFunc1(2+1)
                                recFunc1(3+1)
                                print(4)
                        print(3)
                print(2)
        print(1)
print(0)
Hence the output will be 4 3 2 1 0.
```

For function2 the order of execution will be: -

```
recFunc2(0)
        print(0)
        recFunc2(0+1)
                print(1)
                recFunc2(1+1)
                        print(2)
                        recFunc2(2+1)
                                print(3)
                                recFunc2(3+1)
                                        print(4)
Hence the output this time is 0 1 2 3 4.
```

So, choose the behavior whatever is required by the application. Otherwise things will not be as per expectations. You must be aware of the execution of recursion stack.

The calling sequence of factorial function can be written as: -

Code Implementation

Javascript

```javascript
function fact_recursive(num){
    console.log(`I am coming to calculate the factorial of ${num}`);
    if (num == 0)
    {
        console.log(`I am returning the factorial of 0.`);
        return 1;
    }
    else{
        let fact = num * fact_recursive(num-1);      // Call recursively with lesser number.
        console.log(`I am done calculating the factorial of ${num} = ${fact}`);
        return fact;
    }
}

function main(){
    let number, fact1;
    number = 4;
    fact_recursive(number);        // Call the Recursive version
}

main()
```

C

```c
#include<stdio.h>

int fact_recursive(int num)
{
  printf("I am coming to calculate factorial of %d\n",num);     // On entering recursive call
  int fact=1;
  if (num == 0)
```

```
  {
    printf("I am returning the factorial of 0.\n");
    return 1;     // fact() return 1 if argument is 0
  }
  else
  {
    fact= num * fact_recursive(num-1);      // Call recursively with lesser number.
    printf("I am done calculating the factorial of %d = %d\n", num, fact);
    return fact;
  }
}

int main()
{
  int number;
  number = 4;
  fact_recursive(number);        // Call the Recursive version
  return 0;
}

```

Java

```
class Main{


    static int fact_recursive(int num){
        System.out.println("I am coming to calculate the factorial of "+num);
        if (num == 0)
        {
            System.out.println("I am returning the factorial of 0.");
            return 1;
        }
        else{
            int fact = num * fact_recursive(num-1);  // Call recursively with lesser number.
            System.out.println("I am done calculating the factorial of "+num+" = "+fact);
            return fact;
        }
    }

```

```
    public static void main(String[] args){
        int number, fact1;
        number = 4;
        fact_recursive(number);        // Call the Recursive version
    }
}
```

Python 3

```
def fact_recursive(num):
    print('I am coming to calculate the factorial of',num)
    if (num == 0):
        print('I am returning the factorial of 0')
        return 1
    else:
        result = num * fact_recursive(num-1);      # Call recursively with lesser number.
        print('I am done calculating the factorial of',num,'=',result)
        return result

if __name__ == '__main__':
    number = 4
    fact_recursive(number)        # Call the Recursive version
```

C++

```
#include<iostream>
using namespace std;

int fact_recursive(int num)
{
    cout<<"I am coming to calculate the factorial of "<<num<<endl;
    if (num == 0)
    {
        cout<<"I am returning the factorial of 0.\n";
        return 1;
    }
    else{
        int fact = num * fact_recursive(num-1);      // Call recursively with lesser number.
        cout<<"I am done calculating the factorial of "<<num<<" = "<<fact<<endl;
        return fact;
    }
```

```
}
int main(){
  int number, fact1;
  int num1, num2;
  number = 4;
  fact_recursive(number);        // Call the Recursive version
}
```

The output of above program will show when the call of a value begins and when it finishes: -

```
I am coming to calculate factorial of 4
I am coming to calculate factorial of 3
I am coming to calculate factorial of 2
I am coming to calculate factorial of 1
I am coming to calculate factorial of 0
I am returning the factorial of 0.
I am done calculating the factorial of 1 = 1
I am done calculating the factorial of 2 = 2
I am done calculating the factorial of 3 = 6
I am done calculating the factorial of 4 = 24
```