L1 Introduction of Recursion

Saturday, December 10, 2022 10:48 AM

Recursion

→ When a function call itself Again and again still a Specific Condition Meet that is know as Recursion ☺

→ Lets understand with the help of Example
Write the Recursion program for printing (1)

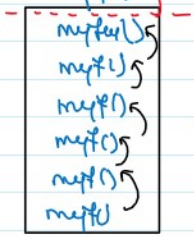
```
myfunction() {
    print(1)
    myfunction()
}
```

myfunction() → calling of the function

Myfunction() → this function call again and again because there is Not a any Stoping point

(Stack Overflow)

Recursion stack =

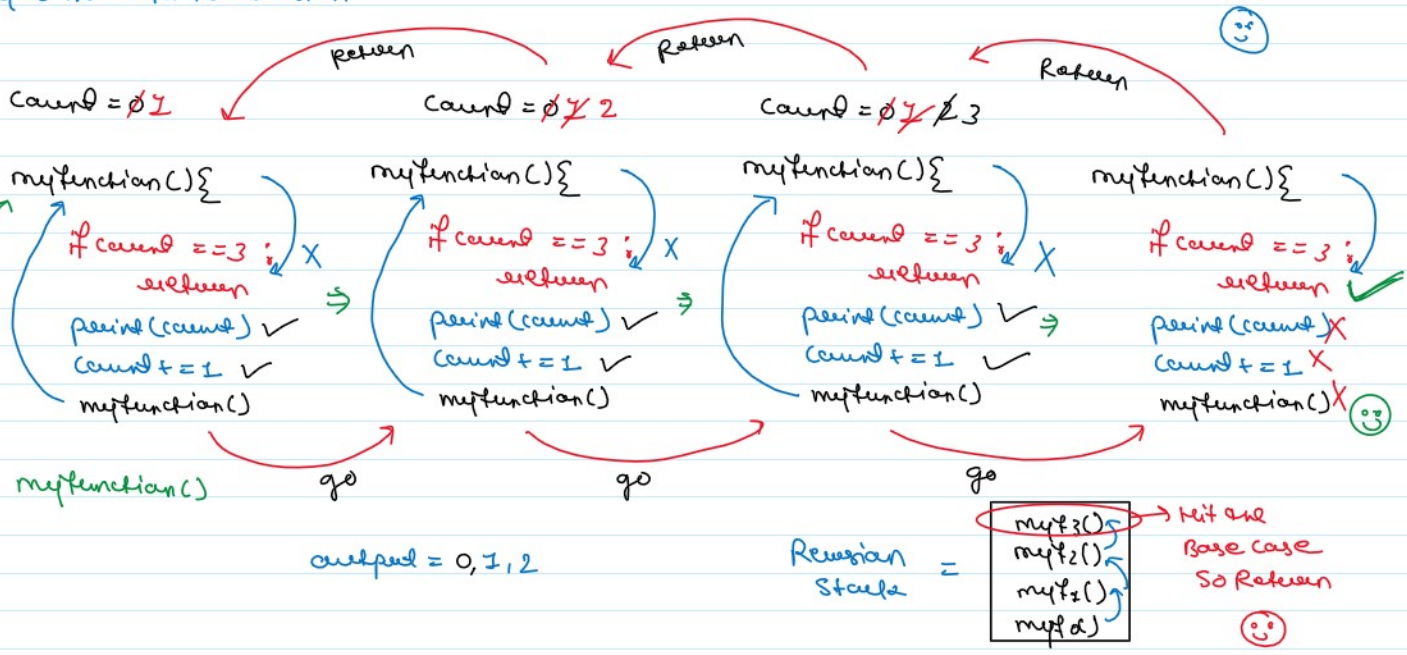


→ Once our Memory get full then show the error Recursion Indepth Exist ☺

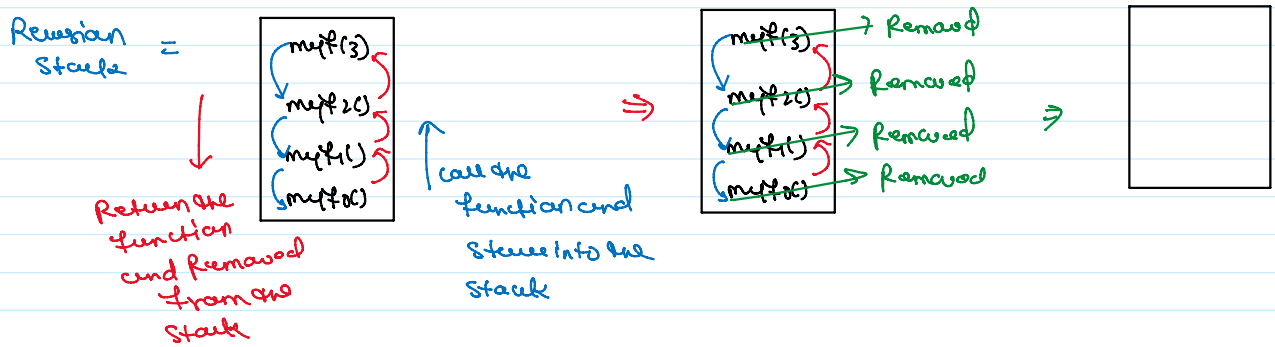
output = 1, 1, 1, 1 ..... ∞

Base Condition

Base condition of that condition where our Recursion goto the Step and prevent to the going to the Infinite Recursion

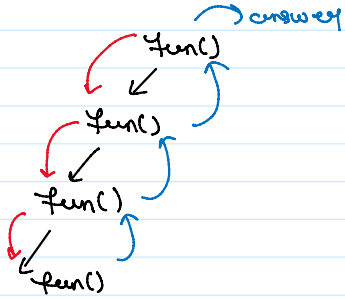


output = 0, 1, 2



Recursion Tree

Recursion Tree is the visiliuzation of the Recursion function using Tree



Example of the previous Question and is the Tree

- Recursion
- Base case
- Stack Overflow/ Stack Space
- Recursion Tree

```python
# todo Finite Recursion with base Case
2 usages
def myfunction(count):

    if count == 3:
        return
    print(count)
    count+=1

    myfunction(count)
myfunction(0)


# todo Infinite Recursion without base Case
2 usages
def myInfinitefunction(count):

    print(count)
    count+=1

    myInfinitefunction(count)
myInfinitefunction(0)
```

```java
public class L1_Introduction_of_Recursion {


    public static void main(String[] args) {

//      todo Finite Recursion with base Case

        myfunction( count: 0);

        //todo Finite Recursion with base Case

        myInfitefunction( data: 0);
    }

    2 usages
    public static void myInfitefunction(int data) {

        System.out.println(data);
        data+=1;
        myInfitefunction(data);


    }

    2 usages
    public static void myfunction(int count) {

        if(count == 3){
            return;
        }
        System.out.println(count);
        count+=1;
        myfunction(count);
    }
}
```