

Stack & Tail Recursion



Stack Recursion

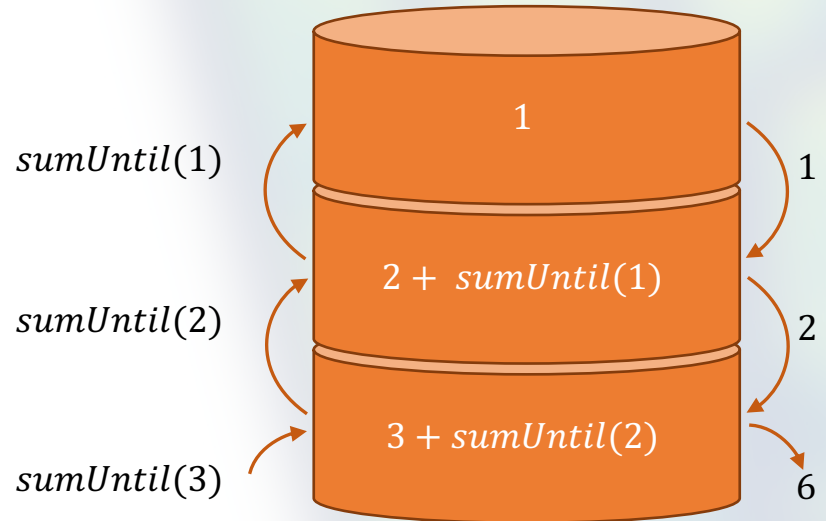
```
def sumUntil(x: Int): Int = {  
  if (x <= 1) 1  
  else x + sumUntil(x - 1)  
}
```

we need to know the result of `sumUntil(x - 1)` to compute `sumUntil(x)`

the JVM has to keep track of the call stack

The stack is limited

Can crash with stack overflow errors



Tail Recursion

The recursive call occurs as the last call of any code path

The JVM computes everything in the same stack frame

```
@tailrec def sumHelper(x: Int, acc: Int): Int = {  
  if (x <= 1) acc  
  else sumHelper(x - 1, x + acc)  
}
```

if the method is not tail recursive we get a compile error:

Recursive call not in tail position

an accumulator keeps track of partial results

Scala rocks

