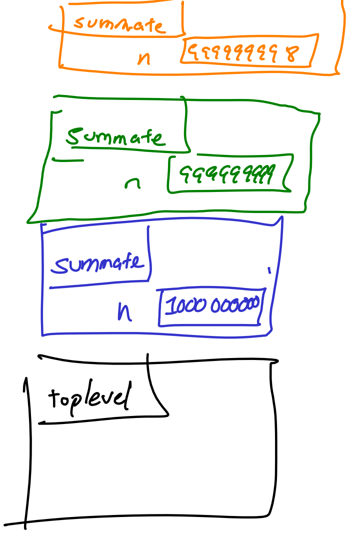
```
let rec summate n=
    if n = 1 then 1 else
        n + summate (n-1)
;;
 summate 100000000 ;;
```



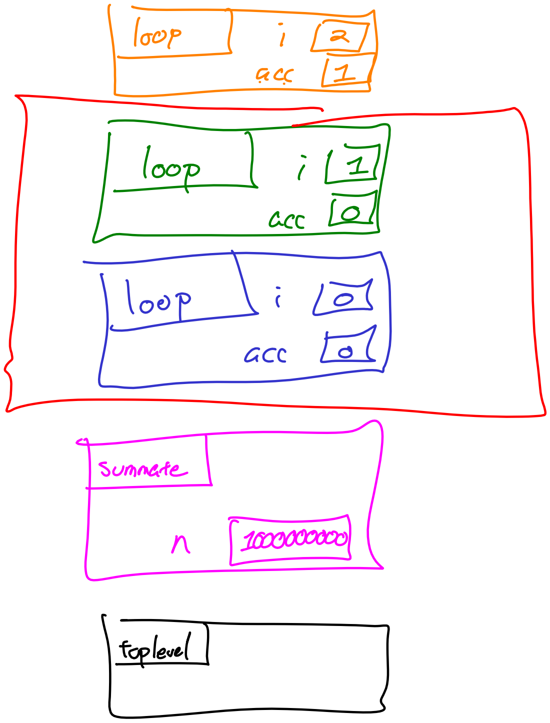
```
let rec summate n=
    if n = 1 then 1 else
        n + summate (n-1)
;;
```

1000000000 + (999999999+ (999999998 + summate (999999998-1)))

```
let summate n =
    let rec loop i acc =
        if i = n+1 then acc else
          loop (i+1) (acc+i)
    in
    loop 0 0
;;

Summate 1000000000;;
```

```
let rec loop i acc =
    if i = 1000000000 + 1 then acc else
        loop (i+1) (acc+i)
in
loop 0 0
        ↓
  loop (0+1) (0+0)
        ⇣
  loop 2 1
        ⇣
  loop (2+1) (1+2)
```



I don't need these

🟦 last operation

🟥 not last operation

$\boxed{f\ 0}$

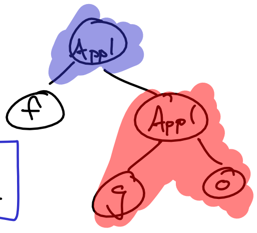
after $(\boxed{f\ 0})$

$\boxed{f\ 0} + \boxed{g\ 0}$

if $\boxed{f\ 0}$ then $\boxed{g\ 0}$ else $\boxed{h\ 0}$

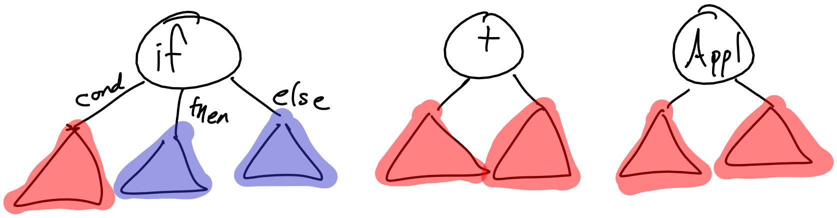
$(\boxed{f\ 0}, \boxed{g\ 0})$

$\boxed{f\ (\boxed{g\ 0})}$



let $a = \boxed{f\ 0}$ in $\boxed{g\ a}$

A "tail expression" is a subexpression which is the last thing a bigger expression will do and gives the result of the bigger expression.



A "tail call" is a tail expression that is a call.

The tail property is determined by location in the AST.