Tail-Call Optimization

Tail Expression: an expression which is executed last (w.r.t. a larger expr) and whose result is the answer (to larger expr)

Tail Calls: a call which is a tail expression

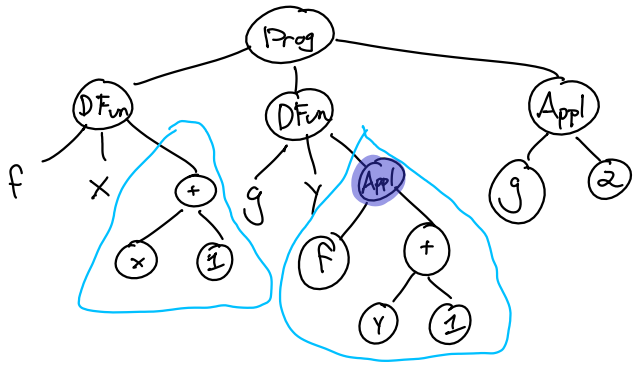
TCO: we eliminate unnecessary stack frames when tail calls occur

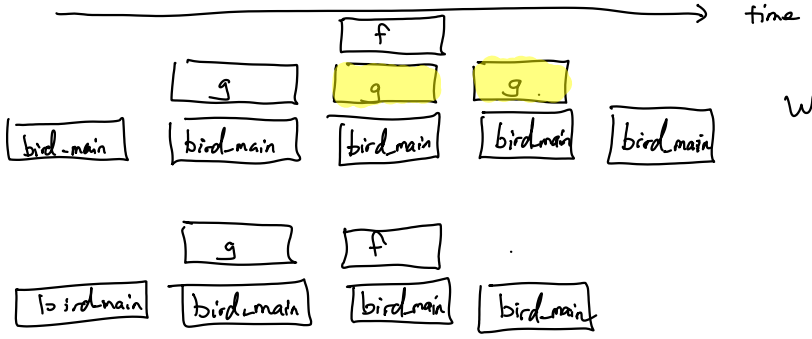
```
def f x =
  x+1
end

def g y =
  f (y+1)
end

g 2
```

↑ we will not TCO in bird_main





w/o TCO

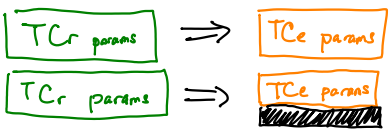
w/ TCO

Tail Callee (TCe) (f)
Tail Caller (TCr) (g)
Original Caller (OC) (bird_main)

Before Call | w/o TCO | w/ TCO

1. Replace TCr params w/ TCe params
2. Tear down TCr stack (including pop rbp)
3. Jmp (not call) to TCe

- TCr #params = TCe #params
- TCr #params > TCe #params
- TCr #params < TCe #params



DON'T TCO

Deciding to TCO

1. Call is a tail expression: compile-time
2. TCr #params ≥ TCe #params: runtime

- Overallocate param memory
  - (but only for TCO)
- On return, rdx contains # params that are in use
- Make stack cleanup the job of callee