

Compiler Construction

Lecture 8: CPS & defunctionalization

Jeremy Yallop

`jeremy.yallop@cl.cam.ac.uk`

Lent 2024

Continuation-**P**assing **S**tyle

Continuation-passing style: motivation

CPS



D17n

Programs in **continuation-passing style** have some useful properties:

Evaluation order is explicit

Every call is a tail call

$$f (g x) \rightsquigarrow g x (\text{fun } y \rightarrow f y k)$$

Every intermediate result is named

Every *continuation* is reified

CPS +
D17n

Mutual
recursion

CPS



D17n

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

let-bind function calls

CPS +
D17n

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else let a = fib (m-1) in  
        let b = fib (m-2) in  
        a+b
```

CPS
convert

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

Mutual
recursion

CPS

1. Add a continuation parameter k to each function
2. Apply k to values returned by the function
3. Replace each application let binding with a continuation argument

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else let a = fib (m-1) in  
        let b = fib (m-2) in  
        a+b
```

CPS
convert

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

CPS +
D17n

Mutual
recursion

CPS

1. Add a continuation parameter k to each function
2. Apply k to values returned by the function
3. Replace each application let binding with a continuation argument

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else let a = fib (m-1) in  
        let b = fib (m-2) in  
        a+b
```

CPS
convert

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

CPS +
D17n

Mutual
recursion

CPS

1. Add a continuation parameter k to each function
2. Apply k to values returned by the function
3. Replace each application let binding with a continuation argument

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else let a = fib (m-1) in  
        let b = fib (m-2) in  
        a+b
```

CPS
convert

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

CPS +
D17n

Mutual
recursion

CPS conversion of fib: details

CPS

1. Add a continuation parameter k to each function
2. Apply k to values returned by the function

D17n

3. Replace each application let binding with a continuation argument

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else let a = fib (m-1) in  
       let b = fib (m-2) in  
       a+b
```

CPS
convert

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

CPS +
D17n

Mutual
recursion

CPS conversion of fib: details

CPS

1. Add a continuation parameter k to each function
2. Apply k to values returned by the function

D17n

3. Replace each application let binding with a continuation argument

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else let a = fib (m-1) in  
        let b = fib (m-2) in  
        a+b
```

CPS
convert

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

CPS +
D17n

Mutual
recursion

Use the identity continuation

CPS

`fib_cps` has the type `int → (int → int) → int`.

To recover a function of type `int → int`, pass the identity continuation `fun x → x`:

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

```
let fib_1 x = fib_cps x (fun x → x)
```

Now `fib_1` can be used like `fib`:

```
List.map fib_1 [0; 1; 2; 3; 4; 5; 6; 7; 8; 9; 10]  
~> [1; 1; 2; 3; 5; 8; 13; 21; 34; 55; 89]
```

D17n

CPS +
D17n

Mutual
recursion

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

$\text{fib_cps } (m+1) \ k$

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

$\text{fib_cps } (m+1) \ k$

$\equiv (\text{expand fib_cps}) \dots$

$\text{if } m+1 = 1 \text{ then } k \ 1 \text{ else fib_cps } ((m+1)-1) (\text{fun } a \rightarrow \text{fib_cps } ((m+1)-2) (\text{fun } b \rightarrow k (a+b)))$

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

```
if m+1 = 1 then k 1 else fib_cps ((m+1)-1) (fun a → fib_cps ((m+1)-2) (fun b → k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else fib_cps ((m+1)-1) (fun a → fib_cps ((m+1)-2) (fun b → k (a+b)))
```

\equiv (arithmetic) ...

```
if m+1 = 1 then k 1 else fib_cps m (fun a → fib_cps (m-1) (fun b → k (a+b)))
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else fib_cps m (fun a → fib_cps (m-1) (fun b → k (a+b)))
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else fib_cps m (fun a → fib_cps (m-1) (fun b → k (a+b)))
```

```
≡ (inductive assumption for m-1 and k = (fun b → k (a+b))) ...
```

```
if m+1 = 1 then k 1 else fib_cps m (fun a → (fun b → k (a+b)) (fib (m-1)))
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else fib_cps m (fun a → (fun b → k (a+b)) (fib (m-1)))
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else fib_cps m (fun a → (fun b → k (a+b)) (fib (m-1)))  
≡ (inductive assumption for m and k = (fun a → (fun b → k (a+b)) (fib (m-1)))) ...  
if m+1 = 1 then k 1 else (fun a → (fun b → k (a+b)) (fib (m-1))) (fib m)
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else (fun a → (fun b → k (a+b)) (fib (m-1))) (fib m)
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else (fun a → (fun b → k (a+b)) (fib (m-1))) (fib m)
```

\equiv (beta reduction $\times 2$) ...

```
if m+1 = 1 then k 1 else k (fib m + fib (m-1))
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else k (fib m + fib (m-1))
```

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

```
if m+1 = 1 then k 1 else k (fib m + fib (m-1))
≡ (if e1 then k e2 else k e3 ≡ k (if e1 then e2 else e3)) ...
k (if m+1 = 1 then 1 else fib m + fib (m-1))
```

Proof

By strong induction on m .

```
let rec fib m =
  if m = 0 then 1
  else if m = 1 then 1
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =
  if m = 0 then k 1
  else if m = 1 then k 1
  else fib_cps (m-1) (fun a →
    fib_cps (m-2) (fun b →
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

$k (\text{if } m+1 = 1 \text{ then } 1 \text{ else fib } m + \text{fib } (m-1))$

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

D17n

CPS +
D17n

Mutual
recursion

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

$k (\text{if } m+1 = 1 \text{ then } 1 \text{ else fib } m + \text{fib } (m-1))$

\equiv (definition of fib) ...

$k (\text{fib } (m+1))$

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

$k (\text{fib } (m+1))$

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Correctness of CPS conversion for fib

CPS

Claim

For all $m \geq 0$,
for all $k : \text{int} \rightarrow \text{int}$,
 $\text{fib_cps } m \ k = k (\text{fib } m)$.

Base case ($m = 0$): $\text{fib_cps } 0 \ k = k \ 1 = k (\text{fib } 0)$.

Inductive step:

Assume for all $n \leq m$, $k (\text{fib } n) = \text{fib_cps } n \ k$.

We want to show: $\text{fib_cps } (m+1) \ k = k (\text{fib } (m+1))$.

$k (\text{fib } (m+1))$

QED

Proof

By strong induction on m .

```
let rec fib m =  
  if m = 0 then 1  
  else if m = 1 then 1  
  else fib (m-1) + fib (m-2)
```

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a →  
    fib_cps (m-2) (fun b →  
      k (a+b)))
```

D17n

CPS +
D17n

Mutual
recursion

NB: We approximate OCaml functions by mathematical functions, ignoring side effects etc.

Defunctionalization

Defunctionalization properties

CPS

D17n



CPS +
D17n

Mutual
recursion

Defunctionalized programs have some useful properties:

No higher-order functions

All values are data

~~fun x → e~~

All control-flow is first order

Every function is named

CPS

D17n



CPS +
D17n

Mutual
recursion

1. Add a constructor to `fn` for each `fun`
2. Replace each `fun` with its constructor
3. Add a case to `apply` for each `fun`
4. Replace each application `p x` with `apply p x`

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if p x  
    then x :: filter p xs  
    else filter p xs  
  
let f l y =  
  filter (fun x → x < 3) l  
@ filter (fun x → x > y) l
```

```
type fn = Lt_three  
       | Gt of int
```

```
let apply fn x =  
  match fn, x with  
  | Lt_three, x → x < 3  
  | Gt y, x → x > y
```

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if apply p x  
    then x :: filter p xs  
    else filter p xs
```

```
let f l y =  
  filter Lt_three l  
@ filter (Gt y) l
```


Defunctionalization: example

CPS

D17n



CPS +
D17n

Mutual
recursion

1. Add a constructor to `fn` for each `fun`
2. Replace each `fun` with its constructor
3. Add a case to apply for each `fun`
4. Replace each application `p x` with `apply p x`

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if p x  
    then x :: filter p xs  
    else filter p xs  
  
let f l y =  
  filter (fun x → x < 3) l  
  @ filter (fun x → x > y) l
```

```
type fn = Lt_three  
       | Gt_of int  
  
let apply fn x =  
  match fn, x with  
  | Lt_three, x → x < 3  
  | Gt y, x → x > y
```

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if apply p x  
    then x :: filter p xs  
    else filter p xs
```

```
let f l y =  
  filter Lt_three l  
  @ filter (Gt y) l
```

CPS

D17n



CPS +
D17n

Mutual
recursion

1. Add a constructor to `fn` for each `fun`
2. Replace each `fun` with its constructor
3. Add a case to apply for each `fun`
4. Replace each application `p x` with `apply p x`

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if p x  
    then x :: filter p xs  
    else filter p xs  
  
let f l y =  
  filter (fun x → x < 3) l  
  @ filter (fun x → x > y) l
```

```
type fn = Lt_three  
       | Gt of int
```

```
let apply fn x =  
  match fn, x with  
  | Lt_three, x → x < 3  
  | Gt y, x → x > y
```

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if apply p x  
    then x :: filter p xs  
    else filter p xs
```

```
let f l y =  
  filter Lt_three l  
  @ filter (Gt y) l
```

CPS

D17n



CPS +
D17n

Mutual
recursion

1. Add a constructor to `fn` for each `fun`
2. Replace each `fun` with its constructor
3. Add a case to `apply` for each `fun`
4. Replace each application `p x` with `apply p x`

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if p x  
    then x :: filter p xs  
    else filter p xs  
  
let f l y =  
  filter (fun x → x < 3) l  
  @ filter (fun x → x > y) l
```

```
type fn = Lt_three  
       | Gt of int
```

```
let apply fn x =  
  match fn, x with  
  | Lt_three, x → x < 3  
  | Gt y, x → x > y
```

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if apply p x  
    then x :: filter p xs  
    else filter p xs
```

```
let f l y =  
  filter Lt_three l  
  @ filter (Gt y) l
```

CPS

D17n



CPS +
D17n

Mutual
recursion

1. Add a constructor to `fn` for each `fun`
2. Replace each `fun` with its constructor
3. Add a case to `apply` for each `fun`
4. Replace each application `p x` with `apply p x`

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if p x  
    then x :: filter p xs  
    else filter p xs  
  
let f l y =  
  filter (fun x → x < 3) l  
  @ filter (fun x → x > y) l
```

```
type fn = Lt_three  
       | Gt of int
```

```
let apply fn x =  
  match fn, x with  
  | Lt_three, x → x < 3  
  | Gt y, x → x > y
```

```
let rec filter p l =  
  match l with  
  | [] → []  
  | x :: xs →  
    if apply p x  
    then x :: filter p xs  
    else filter p xs
```

```
let f l y =  
  filter Lt_three l  
  @ filter (Gt y) l
```

Combining CPS & defunctionalization

CPS

```
let rec fib_cps m k =  
  if m = 0 then k 1  
  else if m = 1 then k 1  
  else fib_cps (m-1) (fun a → (* K1 *)  
    fib_cps (m-2) (fun b → (* K2 *)  
      k (a+b)))
```

D17n

```
let fib_1 x = fib_cps x (fun x → x) (* ID *)
```

To defunctionalize `fib_cps`, define a constructor for each `fun`:

```
type cont = ID | K1 of int * cont | K2 of int * cont
```

Constructor arguments are free variables, and we treat `k2` as free in `k1`:

```
let k2 = fun a b → k (a+b)  
let k1 = fun a → fib_cps (m-2) (k2 a)
```

CPS +
D17n

Mutual
recursion



CPS

Now define an apply function of type `cont → int → int`

```
type cont = ID | K1 of int * cont | K2 of int * cont
```

```
let rec apply_cont k v = match k, v with  
| ID, a → a  
| K1 (m, k), a → fib_cps_defun (m-2) (K2 (a, k))  
| K2 (a, k), b → apply_cont k (a+b)
```

D17n

and call `apply_cont` at every application of a continuation:

```
and fib_cps_defun m k =  
  if m = 0 then apply_cont k 1  
  else if m = 1 then apply_cont k 1  
  else fib_cps_defun (m - 1) (K1 (m, k))
```

```
let fib_2 m = fib_cps_defun m ID
```

CPS +
D17n



Mutual
recursion

Correctness of fib_cps defunctionalization

CPS

Claim

Let $\langle c \rangle$ represent a continuation $c : \text{int} \rightarrow \text{int}$ constructed by `fib_cps`.
Then

$$\text{apply_cont } \langle c \rangle \ m \quad = \quad c \ m$$

and

$$\text{fib_cps } n \ c \quad = \quad \text{fib_cps_defun } n \ \langle c \rangle$$

CPS +
D17n



Mutual
recursion

(**Proof** left as an exercise)

Observation: continuations have list (stack) structure

CPS

D17n

```
type int_list =
  NIL
  | CONS of int * int_list

type cont =
  ID (* 'Nil' *)
  | K1 of int * cont (* 'Cons' *)
  | K2 of int * cont (* 'Cons' *)
```

Idea: replace cont with standard lists:

```
type tag = SUB2 of int (* K1: k (a+b) *)
          | PLUS of int (* K2: fib_cps (m-2) (k2 a) *)

type tag_list_cont = tag list
```

CPS +
D17n



Mutual
recursion

fib_cps_defun revisited, using lists for continuations

CPS

```
type tag = SUB2 of int | PLUS of int
type tag_list_cont = tag list
```

D17n

```
let rec apply_tag_list_cont k v = match k, v with
  | [], a → a
  | SUB2 m :: k, a → fib_cps_defun_tags (m-2) (PLUS a :: k)
  | PLUS a :: k, b → apply_tag_list_cont k (a+b)
```

```
and fib_cps_defun_tags m k =
  if m = 0 then apply_tag_list_cont k 1
  else if m = 1 then apply_tag_list_cont k 1
  else fib_cps_defun_tags (m-1) (SUB2 m :: k)
```

```
let fib_3 m = fib_cps_defun_tags m []
```

CPS +
D17n



Mutual
recursion

Mutual recursion

Mutual recursion \rightsquigarrow single recursion

CPS

Mutual recursion can be eliminated using **indexing**.

Given a set of mutually-recursive functions:

D17n

```
let rec is_even n = n = 0 || is_odd (n - 1)
and is_odd n = n <> 0 && is_even (n - 1)
```

define an index datatype with one constructor for each function:

```
type eo = Even | Odd
```

CPS +
D17n

and define a function that maps an index argument to a corresponding body:

```
let rec is f n =
  match f with
  | Even → n = 0 || is Odd (n - 1)
  | Odd → n <> 0 && is Even (n - 1)
```

Mutual
recursion



Mutual recursion \rightsquigarrow single recursion for fib

CPS

```
type state_type =  
| FIB (* for right-hand-sides starting with fib_ *)  
| APP (* for right-hand-sides starting with apply_ *)
```

D17n

```
type state = (state_type * int * tag_list_cont) → int
```

```
(* eval acts as either apply_tag_list_cont or fib_cps_defun_tags *)
```

```
let rec eval = function
```

```
| FIB, 0, k → eval (APP, 1, k)
```

```
| FIB, 1, k → eval (APP, 1, k)
```

```
| FIB, m, k → eval (FIB, m-1, SUB2 m :: k)
```

```
| APP, a, SUB2 m :: k → eval (FIB, m-2, PLUS a :: k)
```

```
| APP, b, PLUS a :: k → eval (APP, a+b, k)
```

```
| APP, a, [] → a
```

```
let fib_4 m = eval (FIB, m, [])
```

Mutual
recursion



Eliminate tail recursion to obtain *The Fibonacci Machine*

CPS

```
(* step : state → state *)
let step = function
| FIB, 0, k → (APP, 1, k)
| FIB, 1, k → (APP, 1, k)
| FIB, m, k → (FIB, m-1, SUB2 m :: k)
| APP, a, SUB2 m :: k → (FIB, m-2, PLUS a :: k)
| APP, b, PLUS a :: k → (APP, a+b, k)
| _ → failwith "step : runtime error!"
```

D17n

CPS +
D17n

```
let rec driver state = function (* clearly tail recursive! *)
| APP, a, [] → a
| state → driver (step state)
```

```
(* fib_5 : int → int *)
let fib_5 m = driver (FIB, m, [])
```

Mutual
recursion

(This version makes the tail-recursive structure very explicit.)



Tracing of fib_5 6

CPS

```
1 FIB 6 []
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []  
2  FIB  5  [SUB2 6]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []  
2  FIB  5  [SUB2 6]  
3  FIB  4  [SUB2 6, SUB2 5]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []  
2  FIB  5  [SUB2 6]  
3  FIB  4  [SUB2 6, SUB2 5]  
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]  
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]  
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []  
2  FIB  5  [SUB2 6]  
3  FIB  4  [SUB2 6, SUB2 5]  
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]  
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]  
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]  
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
```

D17n

CPS +
D17n

Mutual
recursion



CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
```

D17n

CPS +
D17n

Mutual
recursion



CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
```

D17n

CPS +
D17n

Mutual
recursion



CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

1	FIB	6	[]	26	APP	1	[SUB2 6,PLUS 5,SUB2 3,PLUS 1]
2	FIB	5	[SUB2 6]	27	APP	2	[SUB2 6,PLUS 5,SUB2 3]
3	FIB	4	[SUB2 6,SUB2 5]				
4	FIB	3	[SUB2 6,SUB2 5,SUB2 4]				
5	FIB	2	[SUB2 6,SUB2 5,SUB2 4,SUB2 3]				
6	FIB	1	[SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]				
7	APP	1	[SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]				
8	FIB	0	[SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]				
9	APP	1	[SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]				
10	APP	2	[SUB2 6,SUB2 5,SUB2 4,SUB2 3]				
11	FIB	1	[SUB2 6,SUB2 5,SUB2 4,PLUS 2]				
12	APP	1	[SUB2 6,SUB2 5,SUB2 4,PLUS 2]				
13	APP	3	[SUB2 6,SUB2 5,SUB2 4]				
14	FIB	2	[SUB2 6,SUB2 5,PLUS 3]				
15	FIB	1	[SUB2 6,SUB2 5,PLUS 3,SUB2 2]				
16	APP	1	[SUB2 6,SUB2 5,PLUS 3,SUB2 2]				
17	FIB	0	[SUB2 6,SUB2 5,PLUS 3,PLUS 1]				
18	APP	1	[SUB2 6,SUB2 5,PLUS 3,PLUS 1]				
19	APP	2	[SUB2 6,SUB2 5,PLUS 3]				
20	APP	5	[SUB2 6,SUB2 5]				
21	FIB	3	[SUB2 6,PLUS 5]				
22	FIB	2	[SUB2 6,PLUS 5,SUB2 3]				
23	FIB	1	[SUB2 6,PLUS 5,SUB2 3,SUB2 2]				
24	APP	1	[SUB2 6,PLUS 5,SUB2 3,SUB2 2]				
25	FIB	0	[SUB2 6,PLUS 5,SUB2 3,PLUS 1]				

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

D17n

CPS +
D17n

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
```

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
35 FIB  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
36 APP  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
37 FIB  0  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
37 FIB  0  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
38 APP  1  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
35 FIB  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
36 APP  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
37 FIB  0  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
38 APP  1  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
39 APP  2  [PLUS 8,SUB2 4,SUB2 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
37 FIB  0  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
38 APP  1  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
39 APP  2  [PLUS 8, SUB2 4, SUB2 3]
40 FIB  1  [PLUS 8, SUB2 4, PLUS 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
35 FIB  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
36 APP  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
37 FIB  0  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
38 APP  1  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
39 APP  2  [PLUS 8,SUB2 4,SUB2 3]
40 FIB  1  [PLUS 8,SUB2 4,PLUS 2]
41 APP  1  [PLUS 8,SUB2 4,PLUS 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
37 FIB  0  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
38 APP  1  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
39 APP  2  [PLUS 8, SUB2 4, SUB2 3]
40 FIB  1  [PLUS 8, SUB2 4, PLUS 2]
41 APP  1  [PLUS 8, SUB2 4, PLUS 2]
42 APP  3  [PLUS 8, SUB2 4]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
37 FIB  0  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
38 APP  1  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
39 APP  2  [PLUS 8, SUB2 4, SUB2 3]
40 FIB  1  [PLUS 8, SUB2 4, PLUS 2]
41 APP  1  [PLUS 8, SUB2 4, PLUS 2]
42 APP  3  [PLUS 8, SUB2 4]
43 FIB  2  [PLUS 8, PLUS 3]
44 APP  1  [PLUS 8, PLUS 3]
45 APP  2  [PLUS 8, PLUS 3]
46 APP  3  [PLUS 8, PLUS 3]
47 APP  4  [PLUS 8, PLUS 3]
48 APP  5  [PLUS 8, PLUS 3]
49 APP  6  [PLUS 8, PLUS 3]
50 APP  7  [PLUS 8, PLUS 3]
51 APP  8  [PLUS 8, PLUS 3]
52 APP  9  [PLUS 8, PLUS 3]
53 APP 10  [PLUS 8, PLUS 3]
54 APP 11  [PLUS 8, PLUS 3]
55 APP 12  [PLUS 8, PLUS 3]
56 APP 13  [PLUS 8, PLUS 3]
57 APP 14  [PLUS 8, PLUS 3]
58 APP 15  [PLUS 8, PLUS 3]
59 APP 16  [PLUS 8, PLUS 3]
60 APP 17  [PLUS 8, PLUS 3]
61 APP 18  [PLUS 8, PLUS 3]
62 APP 19  [PLUS 8, PLUS 3]
63 APP 20  [PLUS 8, PLUS 3]
64 APP 21  [PLUS 8, PLUS 3]
65 APP 22  [PLUS 8, PLUS 3]
66 APP 23  [PLUS 8, PLUS 3]
67 APP 24  [PLUS 8, PLUS 3]
68 APP 25  [PLUS 8, PLUS 3]
69 APP 26  [PLUS 8, PLUS 3]
70 APP 27  [PLUS 8, PLUS 3]
71 APP 28  [PLUS 8, PLUS 3]
72 APP 29  [PLUS 8, PLUS 3]
73 APP 30  [PLUS 8, PLUS 3]
74 APP 31  [PLUS 8, PLUS 3]
75 APP 32  [PLUS 8, PLUS 3]
76 APP 33  [PLUS 8, PLUS 3]
77 APP 34  [PLUS 8, PLUS 3]
78 APP 35  [PLUS 8, PLUS 3]
79 APP 36  [PLUS 8, PLUS 3]
80 APP 37  [PLUS 8, PLUS 3]
81 APP 38  [PLUS 8, PLUS 3]
82 APP 39  [PLUS 8, PLUS 3]
83 APP 40  [PLUS 8, PLUS 3]
84 APP 41  [PLUS 8, PLUS 3]
85 APP 42  [PLUS 8, PLUS 3]
86 APP 43  [PLUS 8, PLUS 3]
87 APP 44  [PLUS 8, PLUS 3]
88 APP 45  [PLUS 8, PLUS 3]
89 APP 46  [PLUS 8, PLUS 3]
90 APP 47  [PLUS 8, PLUS 3]
91 APP 48  [PLUS 8, PLUS 3]
92 APP 49  [PLUS 8, PLUS 3]
93 APP 50  [PLUS 8, PLUS 3]
94 APP 51  [PLUS 8, PLUS 3]
95 APP 52  [PLUS 8, PLUS 3]
96 APP 53  [PLUS 8, PLUS 3]
97 APP 54  [PLUS 8, PLUS 3]
98 APP 55  [PLUS 8, PLUS 3]
99 APP 56  [PLUS 8, PLUS 3]
100 APP 57  [PLUS 8, PLUS 3]
101 APP 58  [PLUS 8, PLUS 3]
102 APP 59  [PLUS 8, PLUS 3]
103 APP 60  [PLUS 8, PLUS 3]
104 APP 61  [PLUS 8, PLUS 3]
105 APP 62  [PLUS 8, PLUS 3]
106 APP 63  [PLUS 8, PLUS 3]
107 APP 64  [PLUS 8, PLUS 3]
108 APP 65  [PLUS 8, PLUS 3]
109 APP 66  [PLUS 8, PLUS 3]
110 APP 67  [PLUS 8, PLUS 3]
111 APP 68  [PLUS 8, PLUS 3]
112 APP 69  [PLUS 8, PLUS 3]
113 APP 70  [PLUS 8, PLUS 3]
114 APP 71  [PLUS 8, PLUS 3]
115 APP 72  [PLUS 8, PLUS 3]
116 APP 73  [PLUS 8, PLUS 3]
117 APP 74  [PLUS 8, PLUS 3]
118 APP 75  [PLUS 8, PLUS 3]
119 APP 76  [PLUS 8, PLUS 3]
120 APP 77  [PLUS 8, PLUS 3]
121 APP 78  [PLUS 8, PLUS 3]
122 APP 79  [PLUS 8, PLUS 3]
123 APP 80  [PLUS 8, PLUS 3]
124 APP 81  [PLUS 8, PLUS 3]
125 APP 82  [PLUS 8, PLUS 3]
126 APP 83  [PLUS 8, PLUS 3]
127 APP 84  [PLUS 8, PLUS 3]
128 APP 85  [PLUS 8, PLUS 3]
129 APP 86  [PLUS 8, PLUS 3]
130 APP 87  [PLUS 8, PLUS 3]
131 APP 88  [PLUS 8, PLUS 3]
132 APP 89  [PLUS 8, PLUS 3]
133 APP 90  [PLUS 8, PLUS 3]
134 APP 91  [PLUS 8, PLUS 3]
135 APP 92  [PLUS 8, PLUS 3]
136 APP 93  [PLUS 8, PLUS 3]
137 APP 94  [PLUS 8, PLUS 3]
138 APP 95  [PLUS 8, PLUS 3]
139 APP 96  [PLUS 8, PLUS 3]
140 APP 97  [PLUS 8, PLUS 3]
141 APP 98  [PLUS 8, PLUS 3]
142 APP 99  [PLUS 8, PLUS 3]
143 APP 100 [PLUS 8, PLUS 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
35 FIB  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
36 APP  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
37 FIB  0  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
38 APP  1  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
39 APP  2  [PLUS 8,SUB2 4,SUB2 3]
40 FIB  1  [PLUS 8,SUB2 4,PLUS 2]
41 APP  1  [PLUS 8,SUB2 4,PLUS 2]
42 APP  3  [PLUS 8,SUB2 4]
43 FIB  2  [PLUS 8,PLUS 3]
44 FIB  1  [PLUS 8,PLUS 3,SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
35 FIB  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
36 APP  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
37 FIB  0  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
38 APP  1  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
39 APP  2  [PLUS 8,SUB2 4,SUB2 3]
40 FIB  1  [PLUS 8,SUB2 4,PLUS 2]
41 APP  1  [PLUS 8,SUB2 4,PLUS 2]
42 APP  3  [PLUS 8,SUB2 4]
43 FIB  2  [PLUS 8,PLUS 3]
44 FIB  1  [PLUS 8,PLUS 3,SUB2 2]
45 APP  1  [PLUS 8,PLUS 3,SUB2 2]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
35 FIB  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
36 APP  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
37 FIB  0  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
38 APP  1  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
39 APP  2  [PLUS 8,SUB2 4,SUB2 3]
40 FIB  1  [PLUS 8,SUB2 4,PLUS 2]
41 APP  1  [PLUS 8,SUB2 4,PLUS 2]
42 APP  3  [PLUS 8,SUB2 4]
43 FIB  2  [PLUS 8,PLUS 3]
44 FIB  1  [PLUS 8,PLUS 3,SUB2 2]
45 APP  1  [PLUS 8,PLUS 3,SUB2 2]
46 FIB  0  [PLUS 8,PLUS 3,PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
37 FIB  0  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
38 APP  1  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
39 APP  2  [PLUS 8, SUB2 4, SUB2 3]
40 FIB  1  [PLUS 8, SUB2 4, PLUS 2]
41 APP  1  [PLUS 8, SUB2 4, PLUS 2]
42 APP  3  [PLUS 8, SUB2 4]
43 FIB  2  [PLUS 8, PLUS 3]
44 FIB  1  [PLUS 8, PLUS 3, SUB2 2]
45 APP  1  [PLUS 8, PLUS 3, SUB2 2]
46 FIB  0  [PLUS 8, PLUS 3, PLUS 1]
47 APP  1  [PLUS 8, PLUS 3, PLUS 1]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
37 FIB  0  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
38 APP  1  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
39 APP  2  [PLUS 8, SUB2 4, SUB2 3]
40 FIB  1  [PLUS 8, SUB2 4, PLUS 2]
41 APP  1  [PLUS 8, SUB2 4, PLUS 2]
42 APP  3  [PLUS 8, SUB2 4]
43 FIB  2  [PLUS 8, PLUS 3]
44 FIB  1  [PLUS 8, PLUS 3, SUB2 2]
45 APP  1  [PLUS 8, PLUS 3, SUB2 2]
46 FIB  0  [PLUS 8, PLUS 3, PLUS 1]
47 APP  1  [PLUS 8, PLUS 3, PLUS 1]
48 APP  2  [PLUS 8, PLUS 3]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6,SUB2 5]
4  FIB  3  [SUB2 6,SUB2 5,SUB2 4]
5  FIB  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
6  FIB  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
7  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,SUB2 2]
8  FIB  0  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
9  APP  1  [SUB2 6,SUB2 5,SUB2 4,SUB2 3,PLUS 1]
10 APP  2  [SUB2 6,SUB2 5,SUB2 4,SUB2 3]
11 FIB  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
12 APP  1  [SUB2 6,SUB2 5,SUB2 4,PLUS 2]
13 APP  3  [SUB2 6,SUB2 5,SUB2 4]
14 FIB  2  [SUB2 6,SUB2 5,PLUS 3]
15 FIB  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
16 APP  1  [SUB2 6,SUB2 5,PLUS 3,SUB2 2]
17 FIB  0  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
18 APP  1  [SUB2 6,SUB2 5,PLUS 3,PLUS 1]
19 APP  2  [SUB2 6,SUB2 5,PLUS 3]
20 APP  5  [SUB2 6,SUB2 5]
21 FIB  3  [SUB2 6,PLUS 5]
22 FIB  2  [SUB2 6,PLUS 5,SUB2 3]
23 FIB  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
24 APP  1  [SUB2 6,PLUS 5,SUB2 3,SUB2 2]
25 FIB  0  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
26 APP  1  [SUB2 6,PLUS 5,SUB2 3,PLUS 1]
27 APP  2  [SUB2 6,PLUS 5,SUB2 3]
28 FIB  1  [SUB2 6,PLUS 5,PLUS 2]
29 APP  1  [SUB2 6,PLUS 5,PLUS 2]
30 APP  3  [SUB2 6,PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8,SUB2 4]
34 FIB  2  [PLUS 8,SUB2 4,SUB2 3]
35 FIB  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
36 APP  1  [PLUS 8,SUB2 4,SUB2 3,SUB2 2]
37 FIB  0  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
38 APP  1  [PLUS 8,SUB2 4,SUB2 3,PLUS 1]
39 APP  2  [PLUS 8,SUB2 4,SUB2 3]
40 FIB  1  [PLUS 8,SUB2 4,PLUS 2]
41 APP  1  [PLUS 8,SUB2 4,PLUS 2]
42 APP  3  [PLUS 8,SUB2 4]
43 FIB  2  [PLUS 8,PLUS 3]
44 FIB  1  [PLUS 8,PLUS 3,SUB2 2]
45 APP  1  [PLUS 8,PLUS 3,SUB2 2]
46 FIB  0  [PLUS 8,PLUS 3,PLUS 1]
47 APP  1  [PLUS 8,PLUS 3,PLUS 1]
48 APP  2  [PLUS 8,PLUS 3]
49 APP  5  [PLUS 8]
```

D17n

CPS +
D17n

Mutual
recursion



Tracing of fib_5 6

CPS

```
1  FIB  6  []
2  FIB  5  [SUB2 6]
3  FIB  4  [SUB2 6, SUB2 5]
4  FIB  3  [SUB2 6, SUB2 5, SUB2 4]
5  FIB  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
6  FIB  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
7  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, SUB2 2]
8  FIB  0  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
9  APP  1  [SUB2 6, SUB2 5, SUB2 4, SUB2 3, PLUS 1]
10 APP  2  [SUB2 6, SUB2 5, SUB2 4, SUB2 3]
11 FIB  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
12 APP  1  [SUB2 6, SUB2 5, SUB2 4, PLUS 2]
13 APP  3  [SUB2 6, SUB2 5, SUB2 4]
14 FIB  2  [SUB2 6, SUB2 5, PLUS 3]
15 FIB  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
16 APP  1  [SUB2 6, SUB2 5, PLUS 3, SUB2 2]
17 FIB  0  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
18 APP  1  [SUB2 6, SUB2 5, PLUS 3, PLUS 1]
19 APP  2  [SUB2 6, SUB2 5, PLUS 3]
20 APP  5  [SUB2 6, SUB2 5]
21 FIB  3  [SUB2 6, PLUS 5]
22 FIB  2  [SUB2 6, PLUS 5, SUB2 3]
23 FIB  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
24 APP  1  [SUB2 6, PLUS 5, SUB2 3, SUB2 2]
25 FIB  0  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
26 APP  1  [SUB2 6, PLUS 5, SUB2 3, PLUS 1]
27 APP  2  [SUB2 6, PLUS 5, SUB2 3]
28 FIB  1  [SUB2 6, PLUS 5, PLUS 2]
29 APP  1  [SUB2 6, PLUS 5, PLUS 2]
30 APP  3  [SUB2 6, PLUS 5]
31 APP  8  [SUB2 6]
32 FIB  4  [PLUS 8]
33 FIB  3  [PLUS 8, SUB2 4]
34 FIB  2  [PLUS 8, SUB2 4, SUB2 3]
35 FIB  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
36 APP  1  [PLUS 8, SUB2 4, SUB2 3, SUB2 2]
37 FIB  0  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
38 APP  1  [PLUS 8, SUB2 4, SUB2 3, PLUS 1]
39 APP  2  [PLUS 8, SUB2 4, SUB2 3]
40 FIB  1  [PLUS 8, SUB2 4, PLUS 2]
41 APP  1  [PLUS 8, SUB2 4, PLUS 2]
42 APP  3  [PLUS 8, SUB2 4]
43 FIB  2  [PLUS 8, PLUS 3]
44 FIB  1  [PLUS 8, PLUS 3, SUB2 2]
45 APP  1  [PLUS 8, PLUS 3, SUB2 2]
46 FIB  0  [PLUS 8, PLUS 3, PLUS 1]
47 APP  1  [PLUS 8, PLUS 3, PLUS 1]
48 APP  2  [PLUS 8, PLUS 3]
49 APP  5  [PLUS 8]
50 APP  13 []
```

D17n

CPS +
D17n

Mutual
recursion



CPS

D17n

We turned the recursive cps into a function that **uses no OCaml stack space**

The transformed cps function **carries its own stack** as an extra argument

CPS +
D17n

We **transformed cps incrementally**, with each step easily proved correct

Mutual
recursion



Next time: application to **interpreter 0**