Let's add **first class functions**

```
e ::= ...
    | (defn (f x1... xn) e) ; definition
    | (f e1 ... en)         ; function call
```

```
(defn (incr x)
 (+ x 1))

(defn (f it)
 (it 5))

(f incr)
```

```
pub struct Defn {
    pub name: Option<String>,
    pub params: Vec<String>,
    pub body: Box<Expr>,
}

pub enum Expr {
    ...
    Fun(Defn),
    Call(String, Vec<Expr>),
}
```

```
;; definition of incr
fun_start_incr:
 push rbp
 mov rbp, rsp
 sub rsp, 8*100
fun_body_incr:
 mov rax, [rbp - 8*-2]   ; load x
 add rax, 2              ; add <1>
fun_exit_incr:
 mov rsp, rbp
 pop rbp
 ret
;; definition of f
fun_start_f:
 push rbp
 mov rbp, rsp
 sub rsp, 8*100
fun_body_f:
 mov rax, 10
 push rax
 call FIXME1
 add rsp, 8*1
fun_exit_f:
 mov rsp, rbp
 pop rbp
 ret
;; definition of main
our_code_starts_here:
 ; setup stack frame
 push rbp
 mov rbp, rsp
 sub rsp, 8*100
 ; body of `main`
 mov [rbp - 8], rdi  ; save `input`
 mov r11, rsi        ; save start of heap
 push FIXME2
 call fun_start_f
 add rsp, 8*1
 ; teardown stack frame
 mov rsp, rbp
 pop rbp
 ret
```

```
(defn (sum n)
 (if (= n 0)
     0
     (+ n (sum (+ n -1)))))

(sum input)
```

```
(let (f (fn (f it) (it 5)))
  (let (add (fn (x y) (+ x y)))
     (f add)
  )
)
```