```lua
-- vim: ft=lua ts=2 sw=2 et:

local o,oo,obj,from,within,COCOMO

fmt = string.format
function oo(t) print(o(t)) end
function o(t,    u)
  if #t>0 then return "{"..table.concat(map(t,tostring)," ")..")" else
    u={}; for k,v in pairs(t) do u[1+#u] = fmt(":%s %s",k,v) end
    return (t.is or "")..":{"..table.concat(sort(u)," ")..")" end end

function obj(name,    t,new)
  function new(kl,...)
    local x=setmetatable({},kl); kl.new(x,...); return x end
  t = {__tostring=o, is=name or ""}; t.__index=t
  _ = t
  return setmetatable(t, {__call=new}) end

function _cocomo()
  local p,n,s="+","-","*"
  return { loc = {"!",2,200},
    acap= {n,1,5}, cplx={p,1,6}, prec={s,1,6},
    aexp= {n,1,5}, data={p,2,5}, flex={s,1,6},
    ltex= {n,1,5}, docu={p,1,5}, arch={s,1,6},
    pcap= {n,1,5}, pvol={p,2,5}, team={s,1,6},
    pcon= {n,1,5}, rely={p,1,5}, pmat={s,1,6},
    plex= {n,1,5}, ruse={p,2,6},
    sced= {n,1,5}, stor={p,3,6},
    site= {n,1,5}, time={p,3,6},
    tool= {n,1,5} } end

function _risk()
  local _,ne,nw,nw4,sw,sw4,ne46,w26,sw46
  o = 0
  ne={{o,o,o,1,2,o}, -- bad if lohi
      {o,o,o,o,1,o},
      {o,o,o,o,o,o},
      {o,o,o,o,o,o},
      {o,o,o,o,o,o},
      {o,o,o,o,o,o}}
  nw={{2,1,o,o,o,o}, -- bad if lolo
      {1,o,o,o,o,o},
      {o,o,o,o,o,o},
      {o,o,o,o,o,o},
      {o,o,o,o,o,o},
      {o,o,o,o,o,o}}
  nw4={{4,2,1,o,o,o}, -- very bad if  lolo
       {2,1,o,o,o,o},
       {1,o,o,o,o,o},
       {o,o,o,o,o,o},
       {o,o,o,o,o,o},
       {o,o,o,o,o,o}}
  sw={{o,o,o,o,o,o}, -- bad if  hilo
      {o,o,o,o,o,o},
      {o,o,o,o,o,o},
      {1,o,o,o,o,o},
      {2,1,o,o,o,o},
      {o,o,o,o,o,o}}
  sw4={{o,o,o,o,o,o}, -- very bad if  hilo
       {o,o,o,o,o,o},
       {1,o,o,o,o,o},
       {2,1,o,o,o,o},
       {4,2,1,o,o,o},
       {o,o,o,o,o,o}}
  -- bounded by 1..6
  ne46={{o,o,o,1,2,4}, -- very bad if lohi
        {o,o,o,o,1,2},
        {o,o,o,o,o,1},
        {o,o,o,o,o,o},
        {o,o,o,o,o,o},
        {o,o,o,o,o,o}}
  sw26={{o,o,o,o,o,o}, -- bad if hilo
        {o,o,o,o,o,o},
        {o,o,o,o,o,o},
        {o,o,o,o,o,o},
        {1,o,o,o,o,o},
        {2,1,o,o,o,o}}
  sw46={{o,o,o,o,o,o}, -- very bad if hilo
        {o,o,o,o,o,o},
        {o,o,o,o,o,o},
        {1,o,o,o,o,o},
        {2,1,o,o,o,o},
        {4,2,1,o,o,o}}
  return {
    cplx= {acap=sw46, pcap=sw46, tool=sw46}, --12
    ltex= {pcap=nw4},  -- 4
    pmat= {acap=nw,   pcap=sw46}, -- 6
    pvol= {plex=sw},  --2
    rely= {acap=sw4,  pcap=sw4,  pmat=sw4}, -- 12
    ruse= {aexp=sw46, ltex=sw46},  --8
    sced= {cplx=ne46, time=ne46, pcap=nw4, aexp=nw4, acap=nw4,
           plex=nw4,  ltex=nw, pmat=nw, rely=ne, pvol=ne, tool=nw}, -- 34
    stor= {acap=sw46, pcap=sw46}, --8
    team= {aexp=nw,   sced=nw,  site=nw}, --6
    time= {acap=sw46, pcap=sw46, tool=sw26}, --10
    tool= {acap=nw,   pcap=nw,  pmat=nw}} end -- 6

function from(lo,hi) return lo+(hi-lo)*math.random() end
function within(t)   return t[math.random(#t)] end

COCOMO=obj"COCOMO"

function _:NEW(coc,risk)
  self.x={}; self.y={}

function _:set(t)
  self.y = {}
  for k,v in pairs(t) do self.x[k] = v end end

function _:effort()
  local em,sf=1,0
  for k,t in pairs(self.coc) do
    if     t[1] == "+" then em = em * self.y[k]
    elseif t[1] == "-" then em = em * self.y[k]
    elseif t[1] == "*" then sf = sf + self.y[k] end end
  return self.y.a*self.x.loc^(self.y.b + 0.01*sf) * em end

function _:risks()
  local n=0
  for a1,t in pairs(self.risk) do
    for a2,m in pairs(t) do
      n  = n  + m[self.x[a1]][self.x[a2]] end end
  return n end

function _:y(w,x)
  if w=="!" then return x end
  if w=="+" then return (x-3)*from( 0.073,  0.21 ) + 1 end
  if w=="-" then return (x-3)*from(-0.187, -0.078) + 1 end
  return               (x-6)*from(-1.56,  -1.014) end

function _:ready(coc,risk)
  local y,effort,ready,lo,hi
  coc0, risk0 = cocomo.defaults()
  coc  = coc or coc0
  risk = risk  or risk0
  for k,t in pairs(coc) do
    lo,hi = t[2],t[3]
    self.x[k] = int(self.x[k] and within(self.x[k],lo,hi) or
                    from(lo,hi))
    self.y[k] = self.y[k] or self:y(t[1], self.x[k])
  end
  self.y.a      = self.y.a or from(2.3, 9.18)
  self.y.b      = self.y.b or (.85-1.1)/(9.18-2.2)*self.y.a+.9+(1.2-.8)/2
  self.y.effort = self.y.effort or cocomo:effort()
  self.y.risk   = self.y.risk or cocomo.risks()
  return self end

Eg.all {
  one = function(self)
  local function say()
    print("")
    --lib.o(i.x)
    lib.oo {effort= self.y.effort,
            loc   = self.x.loc,
            risk  = self.y.risk,
            pcap  = self.x.pcap}
  end
  self = cocomo.ready()
  cocomo.new(self, {pcap=4})
  self = cocomo.ready(self)
  say()

  cocomo.set(self, {pcap=1})
  self = cocomo.ready(self)
  say()
end}
```