```
local lib={}
---       _ _   _ _|_|_  _
---     | | |(_| |_| | |_\

local r = math.random
function lib.normal(mu,sd)
  mu, sd = (mu or 0), (sd or 1)
  return mu + sd*math.sqrt(-2*math.log(r()))*math.cos(6.2831853*r()) end

function lib.per(t,p) return t[ ((p or .5)*#t) // 1 ] end

function lib.norm(lo,hi,x) return math.abs(hi-lo)<1E-9 and 0 or (x-lo)/(hi-lo) end

function lib.ent(t, n)
  if not n then n=0; for _,v in pairs(t) do n=n+v end end
  local e=0;         for _,v in pairs(t) do e=e-v/n*math.log(v/n,2) end
  return e,n end

function lib.sd(sorted, f)
  f=f or function(x) return x end
  local denom = 2.564 -- 2*(1.2 + 0.1*(0.9-0.88493)/(0.9032-0.88493))
  return (f(per(sorted, .9)) - f(per(sorted,.1)))/denom end

function lib.cosine(a,b,c)
  return math.max(0,math.min(1, (a^2+c^2-b^2)/(2*c+1E-32))) end

---      _ |_   _   _ |
---     (_ | | (/_ (_ |<

function lib.ish(x,y,z) return math.abs(x-y) <= (z or 0.001) end

---      |`.|_|_ _  _._ _  _
---     ~|~|| |  (/_| || |(_|
---                        _|

function lib.inc(f,a,n)      f=f or{};f[a]=(f[a] or 0) + (n or 1)    return f end
function lib.inc2(f,a,b,n)   f=f or{};f[a]=lib.inc(f[a]  or {},b,n); return f end
function lib.inc3(f,a,b,c,n) f=f or{};f[a]=lib.inc2(f[a] or{},b,c,n);return f end

function lib.has(f,a)      return f[a]                          or 0 end
function lib.has2(f,a,b)   return f[a] and lib.has( f[a],b)   or 0 end
function lib.has3(f,a,b,c) return f[a] and lib.has2(f[a],b,c) or 0 end

---     |. __|_ _
---     ||_\ | _\

lib.unpack = table.unpack

function lib.push(t,x) t[1 + #t] = x; return x end

function lib.powerset(s)
  local function fun(s)
    local t = {{}}
    for i = 1, #s do
      for j = 1, #t do
        t[#t+1] = {s[i], lib.unpack(t[j])} end end
    return t end
  return lib.sort(fun(s), function(a,b) return #a < #b end) end

function lib.merge(b4, merge)
  local j,n,tmp = 1,#b4,{}
  while j<=n do
    local a, b = b4[j], b4[j+1]
    if b then
      local c = merge(a, b) -- returns nil if merge fails
      if c then
        a,j = c,j+1 end end
    tmp[#tmp+1] = a
    j = j+1 end
  return #tmp==#b4 and tmp or lib.merge(tmp,merge) end


---      |`.|_|_ _  _._ _  _
---     ~|~|| |  (/_| || |(_|
---                        _|

function lib.map(t, f, u)
  u={}; for k,v in pairs(t) do u[1+#u]=f(v) end; return u end
function lib.collect(t,f,u)
  u={}; for k,v in pairs(t) do u[k]=f(k,v) end; return u end
function lib.copy(t,   u)
  if type(t) ~= "table" then return t end
  u={}; for k,v in pairs(t) do u[lib.copy(k)] = lib.copy(v) end; return u end

---      _  _  __|_._ _  _
---     _\(_)|   |_| | |(_|
---                      _|

function lib.sort(t,f) table.sort(t,f); return t end

function lib.upx(a,b)   return a.x < b.x end
function lib.up1(a,b)   return a[1] < b[1] end
function lib.down1(a,b) return a[1] > b[1] end

function lib.slots(t, u)
  local function public(k) return tostring(k):sub(1,1) ~= "_" end
  u={};for k,v in pairs(t) do if public(k) then u[1+#u]=k end end
  return lib.sort(u) end

---      _ _|_ _. ._ _|_      ._
---     _>  |_(_| |    |_  |_| |_)
---                            |

function lib.settings(help)
  local d,used = {},{}
  help:gsub("\n ([-]([^%s]+))[%s]+(-[^%s]+)[^\n]*%s([^%s]+)",
    -- e.g.    "  -bins -b  max.number of bins       = 16"
    --parses to ((-)(bins)) (-b) max number of bins = (16)
    -- i.e.    (long (key)) (short)                    (x)
    function(long,key,short,x)
      assert(not used[short], "repeated short flag ["..short.."]")
      used[short]=short
      for n,flag in ipairs(arg) do
        if flag==short or flag==long then
          x = x=="false" and true or x=="true" and "false" or arg[n+1] end end
      d[key] = lib.coerce(x) end)
  if d.help then os.exit(print(help)) end
  return d end

lib.go = {_fails=0}
function lib.ok(test,msg)
  print("", test and "PASS "or "FAIL ",msg or "")
  if not test then
    lib.go._fails= lib.go._fails+1
    if the and the.dump then assert(test,msg) end end end

function lib.main(the,go,b4,           resets,todos)
  todos = the.todo == "all" and slots(go) or {the.todo}
  resets={}; for k,v in pairs(the) do resets[k]=v end
  go._fails = 0
  for _,todo in pairs(todos) do
    math.randomseed(the.seed or 10019)
    if go[todo] then print("\n"..todo); go[todo]() end
    for k,v in pairs(resets) do the[k]=v end end
  for k,v in pairs(_ENV) do
    if b4 and not b4[k] then print("?",k,type(v)) end end
  os.exit(go._fails) end

---       _  _ | _  _|_. _ ._
---     _\(/_|(/_(_ |_|(_)| |

function lib.any(a,lo,hi)
  lo,hi = lo or 1, hi or #a; return a[ (lo+(hi-lo)*math.random())//1 ] end

function lib.many(a,n,lo,hi,  u)
  u={}; for j=1,n do lib.push(u, lib.any(a,lo,hi)) end; return u end

function lib.slice(a,lo,hi,    u)
  u,lo,hi = {},lo or 1,hi or #a; for j=lo,hi do u[1+#u]=a[j] end; return u end

---      __|_ _._ _  _   '~)  _|_|_ ._ _  _
---     _\ |_| | | |(_|  /_    |_| | || | |(_|
---                   _|                    _|

function lib.words(s,sep,   t)
  sep="([^" .. (sep or ",")  .. "]+)"
  t={}; for y in s:gmatch(sep) do t[1+#t] = y end; return t end

function lib.coerces(s)
  return lib.map(lib.words(s), lib.coerce) end

function lib.coerce(x)
  if type(x) ~= "string" then return x end
  x = x:match"^%s*(.-)%s*$"
  if x=="true" then return true elseif x=="false" then return false end
  return math.tointeger(x) or tonumber(x) or x end

function lib.items(src,f)
  local function file(f)
    src,f = io.input(src),(f or lib.coerces)
    return function(x) x=io.read()
             if x then return f(x) else io.close(src) end end end
  local function tbl(   x)
    x,f = 0, f or function(z) return z end
    return function() if x< #src then x=x+1; return f(src[x]) end end end
  if src then
    return type(src) == "string" and file(f) or tbl() end end

---     _|_|_ ._ _  _  _   '~)  __|_ _._ _  _
---      |_| | || | |(_|_\   /_  _\ |_| | | |(_|
---                    _|                    _|

lib.fmt = string.format

function lib.oo(t, slots) print(lib.o(t,slots)) end

function lib.o(t,slots,   seen, u)
  if type(t)~="table" then return tostring(t) end
  seen = seen or {}
  if seen[t] then return "..." end
  seen[t] = t
  local function show1(x) return lib.o(x, nil, seen) end
  local function show2(k) return lib.fmt(":%s %s",k, lib.o(t[k], nil, seen)) end
  u = #t>0 and lib.map(t,show1) or lib.map(slots or lib.slots(t),show2)
  return (t._is or "").. "{"..table.concat(u," ").."}" end

function lib.dent(t,  seen,pre)
  pre,seen = pre or "", seen or {}
  if seen[t] then t= "..." end
  if type(t)~="table" then return print(pre .. tostring(t)) end
  seen[t]=t
  for key,k in pairs(lib.slots(t)) do
    local v = t[k]
    io.write(lib.fmt("%s:%s%s",pre,k, type(v)=="table" and "\n" or " "))
    if   type(v)=="table"
    then lib.dent(v,seen,"| "..pre)
    else print(v) end end end

function lib.rnds(t,f)
  return lib.map(t, function(x) return lib.rnd(x,f) end) end

function lib.rnd(x,f)
  return lib.fmt(type(x)=="number" and (x~=x//1 and f or "%5.2f") or "%s",x) end

---      _  |_   o  _   _ _|_
---     (_) |_)  | (/_ (_  |_
---             _|

local _id=0
function lib.id() _id=_id+1; return _id end

function lib.class(name,base)
  local klass, base_ctor = {}
  if base then
    for k,v in pairs(base) do klass[k] = v end
    klass._base = base
    base_ctor   = rawget(base,'new') end
  klass.__index = klass
  klass._is     = name
  klass._class  = klass
  return setmetatable(klass,{
    __call = function(klass,...)
      local obj = setmetatable({},klass)
      if     rawget(klass,'new')
      then   klass.super = base_ctor
             local res = klass.new(obj,...)
             if res then obj = setmetatable(res,klass) end
      elseif base_ctor then base_ctor(obj,...) end
      return obj end }) end

lib.Obj = lib.class("Obj")

function lib.Obj:show(  t)
  t={}
  for k,v in pairs(self) do if tostring(k):sub(1,1)~="_" then t[1+#t]=k end end
  return lib.sort(t) end

function lib.Obj:__tostring(  u) return lib.o(self,self:show()) end

--u={}; for _,k in pairs(self:show()) do u[1+#u]=lib.fmt(":%s %s",k,self[k]) end
--  return self._is .."{"..table.concat(u," ").."}" end

return lib
```