```lua
local b4={}; for k,_ in pairs(_ENV) do b4[k]=k end
local class = require"class"

local r = math.random
local fmt=string.format
local function per(a,p) return a[1+((p or .5)*#a)//1] end
local function sort(t,f) table.sort(t,f); return t end

local Obj=class("Obj")

function 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 sort(t) end

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

local Col = class("Col", Obj)
function Col:new(at,name)
  self.n = 0
  self.at = at or 0
  self.name = name or "" end

function Col:adds(t)
  for _,v in pairs(t) do self:add(v) end; return self end

function Col:add(x,inc)
  if x ~= "?" then inc=inc or 1; self.n = self.n + inc; self:add1(x,inc) end
  return x end

function Col:merged(other,  out)
  out = self:merge(other)
  if out:div()*.95 <= (sellf.n*self:div() + other.n*other:div())/out.n then
    return out end end

---      ._     ._ _
---      | |  |_|  | | |

local Num = class("Num", Col)
function Num:new(at,name)
  self:super(at,name)
  self.w = self.name:find"-$" and -1 or 1
  self.ok, self.has  = true,{}
  self.max= 64
  self.lo,self.hi = math.huge,-math.huge end

function Num:add1(x,inc)
  self.hi = math.max(x, self.hi)
  self.lo = math.min(x, self.lo)
  local a = self.has
  if     #a  < self.max        then self.ok=false; a[1+#a]         =x
  elseif r() < self.max/self.n then self.ok=false; a[1+(r()*#a)//1] =x end end

function Num:all()
  if not self.ok then self.ok=true; table.sort(self.has) end
  return self.has end

function Num:mid()   return per(self:all(), .5) end
function Num:div(  a) a=self:has(); return (per(a,.9) - per(a..1))/2.54 end

function Num:same(x,y) return math.abs(x-y) <= self:div()*.35 end

function Num:merge(other,   out)
  out = Num(self.at, self.name)
  for _,n in pairs(self.has)  do out:add(n) end
  for _,n in pairs(other.has) do out:add(n) end
  return out end

---       _      ._ _
---      _>  \/  | | |
---          /

local Sym = class("Sym", Col)
function Sym:new(at,name)
  self:super(at,name)
  self.has = {}
  self.mode,self.most = nil,0 end

function Sym:add1(x,inc)
  self.has[x] = (self.has[x] or 0) + inc
  if self.has[x] > self.most then
    self.most, self.mode = self.has[x], x end end

function Sym:mid() return self.mode end
function Sym:div(  e,p)
  e=0; for _,v in pairs(self.has) do p=v/self.n; e = e - p*math.log(p,2) end
  return e end

function Sym:merge(other,   out)
  out = Sym(self.at, self.name)
  for x,n in pairs(self.has)  do out:add(x,n) end
  for x,n in pairs(other.has) do out:add(x,n) end
  return out end

print(Sym(23,"thing"):adds{"a","a","b"})
local n = Num(23,"thing")
for i=1,1000 do n:add(i) end

for i,x in pairs(n:all()) do io.write(x," ") end

for k,v in pairs(_ENV) do if not b4[k] then print("?",k,type(v)) end end
```