Dec 17, 21 21:23

```
local the, help = {}, [[
ween.lua [OPTIONS]
ween (vb), archaic. To think or imagine.
  A small sample multi-objective optimizer / data miner. (c)2021 Tim Menzies <timm@ieee.org> unlicense.org
OPTIONS:
-best X Best end of the examples.
-debug X Run one test, show stack dumps on fail.
-file X Read data from files.
-h Show help.
-hints X How many to evaluate each iteration.
-p X Coefficient on distance calculation.
-seed X Random number seed.
-todo X Demos to run at start-up. 'all'=run all. = ing]
  local b4=(); for k,_ in pairs(_ENV) do b4[k]=k end
local function rogues()
  for k,v in pairs(_ENV) do if not b4[k] then print("?:",k,type(v)) end end end
   local pop.csv,fmt,map,keys,sort,copy,norm,push local color,first,firsts,coerce,second22,shuffle,bchop fmt = string.format function coerce(x) return tonumber(x) or x end function color(n,s) return fmt("\27|\mathbb{m}27|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{m}37|\mathbb{
                                                                                                         return tonumber(x) or x end
return fmt("\27[1m\27[%sm%s\27[0m",n,s) end
  local shout, out
function shout(x) print(out(x)) end
function out(t, u,key,val)
function out(t, u,key,val)
function (key(_,k) return string.format(".%s %s", k, out(t[k])) end
function val(_,v) return out(v) end
if type(t) -= "table" then return tostring(t) end
u = #t>0 and map(t, val) or map(keys(t), key)
return "("..table.concat(u,"")..")" end
  function bchop(t,val,policy, lo,hi,mid)
lt = lt or function(x,y) return x < y end
lo,hi = 1,#t
while lo <= hi do
mid = (lo+hi) // 2
if policy(t[mid],val) then lo=mid+1 else hi= mid-1 end end
return math.min(lo,#t) end</pre>
  local randi, rand, Seed -- remember to set seed before using this function randi(lo,hi) return math.floor(0.5 + rand(lo,hi)) end function rand(lo,hi)
lo, hi = lo or 0, hi or 1
Seed = (16807 * Seed) % 2147483647
return lo + (hi-lo) * Seed / 2147483647 end
   local slurp, sample, dist, ordered, hint, left_is_best function slurp( i) for eg in csv(the.file) do i=sample(i,eg) end; return i end
function slurp( i) for eg in csv(the.file) do i=sample(i,eg) end; return i en
function sample(i,eg)
local numeric,independent,dependent,head,data,datum
i = i or (xs=(),nys=0,ys=(),lo=(),hi=(),w={}),beads={},divs={});
function head (n,x)
function independent () i.lo[n]= math.huge; i.hi[n]= -i.lo[n] end
function dependent ()
i.w[n] = xifind"-" and -l or 1
i.ys[n] = xinys+1 end
i.nys = i.nys+1 end
if or xifind"" then
if xifind"-" or xifind"+" then dependent () else independent () end
if xifind"-" or xifind"+" then
function datum(n,x)
if x = """ then
if i.lo[n] = math.min(i.lo[n],x)
i.hi[n] = math.min(i.lo[n],x)
i.hi[n] = math.min(i.lo[n],x)
end
if fi.heads==0 then i.heads=map(eg,head) else push(i.egs,map(eg,datum)) end
function left is best(i.left right = x hlafts right)
   function left_is_best(i,left,right, a,b,lefts,rights)
lefts,rights=0,0
for n,_ in pairs(i,ys) do
    a = norm(i.lo[n], i.hi[n], left[n])
    b = norm(i.lo[n], i.hi[n], right[n])
lefts = lefts - 2.71828*(i.w[n] * (a-b)/i.nys)
rights = rights - 2.71828*(i.w[n] * (b-a)/i.nys)
end
return lefts/i.nys < rights/i.nys end</pre>
  function ordered(i,egs)
  return sort(egs or i.egs, function(a,b) return left_is_best(i,a,b) end) end
   function dist(i,eg1,eg2)
  local function dist1(lo,hi,a,b)
  if lo
          local d,n = 0,0
local a,b,inc
          local a,b,inc
for col,_ in pairs(i.xs) do
    a,b = egl[col], eg2[col]
    inc = a==""" and b==""" and 1 or dist1(i.lo[col],i.hi[col],a,b)
    d = d + inc^the.p
    n = n + 1 end
return (d/n)^(1/the.p) end
  return (d/n)^(l/the.p) end
function hint(i,egs)
local function hint(legs, all, min, evals,lvl)
local scoreds, nearest,best = {}
function nearest(_reg, tmp)
    return sort(map(scoreds, function(rank,scored)
    return sort(map(scoreds, function(rank,scored)
    return sort(map(scoreds, function(rank,scored)
    return sort(map(scoreds, function(rank,scored)
    return map(scoreds, function(rank,scored)
    return map(sort(rank) else,scored)/10^6, eg) end), firsts)[1] end
    then scoreds = egs
    then scoreds = egs
    then scoreds = egs
    else for j=1,the.hints do push(scoreds, pop(egs)) end
    best, scoreds = {}, ordered(1,scoreds)
    egs = sort(map(egs,nearest),firsts)
    for j=1,tegs)//2 do push(best, egs[j][2]) end
    return hint1(best, all, min, evals+the.hints,lvl..*|..") end
end
```

local go=() innction go_ing() return true end if function go_ing() return true end if function go_cav() in n=0; for eg in csv(the.file) do n=n+1; if n>390 then shout(eg) end end end if function go_ib() if t= (10,20,30,40) if t= (10,20,40,40) if t		
<pre>function go.ing() return true end function go.te() shout(the) end function go.csv( n)     n=0; for eq in cav(the.file) do n=n+1; if n&gt;390 then shout(eg) end end end function go.lib( u,t)     t= (20,20,40 (u)t)     assert(u[1] == t[1])     assert(3 == s.lo[1]) end     function go.ordered( _,i,egs)     i = slurp()     egs = ordered(i)     shout(i.heads)     for j=1,5 do shout(egs[j]) end     print("#")     for j=i=sgs=-5, i=sg do shout(egs[j]) end end  function dist1(_,eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     t=map(i.egs,dist1)     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("#")     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("#")     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end      i = slurp()     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end      i = function dist1(_,eg)     for i=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end      i = function dist1(_,eg)     for i=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end      i = function dist1(_,eg)     for i=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end      i = function dist1(_,eg)     i = function dist1(</pre>	148	
<pre>function go.the() shout(the) end function go.csv( n) n=0; for eq in csv(the.file) do n=n+1; if n&gt;390 then shout(eq) end end end function go.lib( u,t) t= [10,20,30,40) u=(10,20,30,40) u=(10,20,40,40) u=</pre>	149	local go={}
<pre>function go.the() shout(the) end function go.csv( n) n=0; for eq in csv(the.file) do n=n+1; if n&gt;390 then shout(eq) end end end function go.lib( u,t) t= [10,20,30,40) u=(10,20,30,40) u=(10,20,40,40) u=</pre>	450	function as ing() return true and
<pre>function go.csv( n)     n=0; for eq in cav(the.file) do n=n+1; if n&gt;390 then shout(eq) end end end function go.lib(</pre>		
n=0; for eg in csv(the.file) do n=n+1; if n>390 then shout(eg) end end end function go.lib( u,t)  t = (10,20,30,40)  u = copy(shuffle(t))  t[1]=100  assert(u[1] == t[1])  assert(u[1] == t[1])  assert(u[1] == t[0])  sesulurp()  function go.sample( s)  sesulurp()  sesulurp()  function go.ordered( _,i,egs)  i = slurp()  egs = ordered(i)  shout(i.heads)  for j=1,5 do shout(egs[j]) end  print('#")  for j=sess-5,*egs do shout(egs[j]) end end  function go.dist( i,dist1,t)  function dist1(_,eg) return (dist(i,i.egs[1],eg), eg) end  i = slurp()  t=map(i.egs,dist1)  for j=1,5 do print(',fmt("%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for j=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for j=isso-5,*egs do shout(egs[j]) end end  function go.dist( i,dist1,t)  function jo.dist( i,dist1,t)  function tist( i,sort,sort2,s,t), end  print('#")  for j=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for j=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for i=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for the distance in main (tap.go)  i = slurp()  sort!= ordered(i)  sor	151	function go.the() shout(the) end
n=0; for eg in csv(the.file) do n=n+1; if n>390 then shout(eg) end end end function go.lib( u,t)  t = (10,20,30,40)  u = copy(shuffle(t))  t[1]=100  assert(u[1] == t[1])  assert(u[1] == t[1])  assert(u[1] == t[0])  sesulurp()  function go.sample( s)  sesulurp()  sesulurp()  function go.ordered( _,i,egs)  i = slurp()  egs = ordered(i)  shout(i.heads)  for j=1,5 do shout(egs[j]) end  print('#")  for j=sess-5,*egs do shout(egs[j]) end end  function go.dist( i,dist1,t)  function dist1(_,eg) return (dist(i,i.egs[1],eg), eg) end  i = slurp()  t=map(i.egs,dist1)  for j=1,5 do print(',fmt("%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for j=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for j=isso-5,*egs do shout(egs[j]) end end  function go.dist( i,dist1,t)  function jo.dist( i,dist1,t)  function tist( i,sort,sort2,s,t), end  print('#")  for j=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for j=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for i=1,5 do print(',fmt('%5.3f",t[j][1]),out(t[j][2])) end  print('#")  for the distance in main (tap.go)  i = slurp()  sort!= ordered(i)  sor	152	function go.csv( n)
<pre>function go.lib( u,t)     t = (10,20,30,40)     u = copy(shuffle(t))     t[]=100     assert(u[1] -= t[1])     assert(u[1] -= t[1])     assert(u[1] -= 100) end  function go.sample( s)     s=sulurp()     assert(3= s.lo[1]) end  function go.ordered( _,i,egs)     i = slurp()     egs = ordered(i)     shout(i.heads)     for j=1,5 do shout(egs[j]) end     print("f)     function go.dist( i,distl,t)     function do.dist( i,distl,t)     function dosti(,eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     t=map(i.egs,distl)     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("f")     for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("f")     for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function do.hint( i,sort),sort2,s,lt)     function function lt(a,b) return left_is_best(i,a,b) end     i=slurp()     sort2 = int(i)     for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end     if m &gt; (*sort2)-20 then shout(eg) end end </pre>		
t= (10,20,30,40) u= copy(shuffle(t)) t[1]=100 assert(u[1] -= t[1]) asser		
<pre>u= copy(shuffle(t)) t[]=100 assert(u[1] ~= 1(1)) assert(u[1] ~= 100) end  function go.sample( s) s=slurp() assert(39s = #s.egs) assert(39s = #s.egs) assert(39s = *s.egs) assert(39s = *s.egs) assert(39s = st.egs) i = slurp() egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=egs-5,#egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist1(_,eg) return {dist(i,i.egs[1],eg), eg) end i = slurp() for j=t,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for i=slurp() sort= ordered(i) sort= hint(i) for m,eg in pairs(sort2) do i = if m &gt; (*tort2)-20 then shout(eg) end end i = function in an in the, ego) </pre>	154	function go.lib( u,t)
<pre>u= copy(shuffle(t)) t[]=100 assert(u[1] ~= 1(1)) assert(u[1] ~= 100) end  function go.sample( s) s=slurp() assert(39s = #s.egs) assert(39s = #s.egs) assert(39s = *s.egs) assert(39s = *s.egs) assert(39s = st.egs) i = slurp() egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=egs-5,#egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist1(_,eg) return {dist(i,i.egs[1],eg), eg) end i = slurp() for j=t,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for i=slurp() sort= ordered(i) sort= hint(i) for m,eg in pairs(sort2) do i = if m &gt; (*tort2)-20 then shout(eg) end end i = function in an in the, ego) </pre>	155	t= {10,20,30,40}
t[1]=100 assert(u[1] ~= t[1])		
<pre>assert(u[1] ~= [1]) assert(u[1] ~= 100) end  function go.sample( s) s=slurp() assert(398 == #s.egs) assert(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,</pre>		
<pre>assert(u[i] ~= 100) end  function go.sample( s)     s=slurp()  assert(398 == #s.egs)     assert(3 == s.lo[i]) end  function go.ordered( _,i,egs)     i = slurp()     egs = ordered(i)     shout(i.heads)     for j=1,5 do shout(egs[j]) end     print("#")  for j=esg-5, #egs do shout(egs[j]) end end  function go.dist( i,dist1,t)  function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end  fi = slurp()     function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end  for j=(#t)-5,#t do print(j,fmt("%5.31",t[j][1]),out(t[j][2])) end  print("#")  for j=(#t)-5,#t do print(j,fmt("%5.31",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt)  function lt(a,b) return left_is_best(i,a,b) end     i=slurp()  sort2= hint(i)  for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end  if m &gt; (#sort2)-20 then shout(eg) end end end  - Return to the operating system then number of failing demos.  local function main(the,eg)  local no,defaults = 0,()  for k,v in pairs(the) do defaults[k]=v end  local function reset(x)  for k,v in pairs(the) do defaults[k]=v end  seed = the.seed or 10019 end  reset()  local function reset(x)  for k,v in pairs(the.todo=="all" and keys(go) or (the.todo)) do     if type(go[it]) == "function" then return print("NOFUN:",it) end  reset()  local ok,msg = peall( go[it])  if ok then print(color(32, "PASS")it)  else print(color(31, "FAIL")it,msg); no=no+1 end end  rogues()  os.exit(no) end  - Make 'the' options array from help string and any updates from command line.  (help or ") rigsub("^*OPTIONS:", "") rigsub("Mss-("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms- /"Mss("Ms-</pre>	157	t[1]=100
<pre>assert(u[i] ~= 100) end  function go.sample( s)     s=slurp()  assert(398 == #s.egs)     assert(3 == s.lo[i]) end  function go.ordered( _,i,egs)     i = slurp()     egs = ordered(i)     shout(i.heads)     for j=1,5 do shout(egs[j]) end     print("#")  for j=esg-5, #egs do shout(egs[j]) end end  function go.dist( i,dist1,t)  function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end  fi = slurp()     function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end  for j=(#t)-5,#t do print(j,fmt("%5.31",t[j][1]),out(t[j][2])) end  print("#")  for j=(#t)-5,#t do print(j,fmt("%5.31",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt)  function lt(a,b) return left_is_best(i,a,b) end     i=slurp()  sort2= hint(i)  for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end  if m &gt; (#sort2)-20 then shout(eg) end end end  - Return to the operating system then number of failing demos.  local function main(the,eg)  local no,defaults = 0,()  for k,v in pairs(the) do defaults[k]=v end  local function reset(x)  for k,v in pairs(the) do defaults[k]=v end  seed = the.seed or 10019 end  reset()  local function reset(x)  for k,v in pairs(the.todo=="all" and keys(go) or (the.todo)) do     if type(go[it]) == "function" then return print("NOFUN:",it) end  reset()  local ok,msg = peall( go[it])  if ok then print(color(32, "PASS")it)  else print(color(31, "FAIL")it,msg); no=no+1 end end  rogues()  os.exit(no) end  - Make 'the' options array from help string and any updates from command line.  (help or ") rigsub("^*OPTIONS:", "") rigsub("Mss-("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms+ /"Mss("Ms- /"Mss("Ms-</pre>	158	assert (u[1] ~= t[1])
<pre>function go.sample( s)     s=slurp()     assert(398 == \$s.egs)     assert(398 == \$s.egs)     assert(398 == \$s.egs)     is alurp()     egs = ordered(i)     shout(i.beads)     shout(i.beads)     shout(i.beads)     function go.ordered( _,i,egs)     is = slurp()     shout(i.beads)     shout(i.beads)     shout(i.beads)     function go.dist( i,distl,t)     function distl(_,eg) return {dist(i,i.egs[1],eg), eg) end     is = slurp()     function distl(_,eg) return {dist(i,i.egs[1],eg), eg) end     is = slurp()     function distl(_,eg) return {dist(i,i.egs[1],eg), eg) end     is = slurp()     function file(i)     in slurp()     function file(i)     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("#")     for j=(\$t\)-5,\$t\ do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("#")     for j=(\$t\)-5,\$t\ do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     ind     function go.hint( i,sort1,sort2,s,lt)     function lt(a,b) return left_is_best(i,a,b) end     is=slurp()     sort1= ordered(i)     sort2= hint(i)     for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end     if m &lt; (\$20\$ then shout(eg) end end end </pre>		
<pre>function go.sample( s) s=sulrp() assert(398 == #s.egs) assert(3 == s.lo[1]) end  function go.ordered( _,i,egs) i = slurp() egs = ordered() function go.ordered( _,i,egs) i = slurp() egs = ordered() for j=l,5 do shout(egs[j]) end print('#") for j=egs-5, #egs do shout(egs[j]) end end  function go.dist( i,distl,t) function distl(_,eg) return (dist(i,i.egs[1],eg), eg) end i = slurp() t=map(i.egs,distl) for j=l,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print('#") for j=t,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print('#") for j=t,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function lt(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sortl= ordered(i) sortl= ordered(i) sortl= ordered(i) sortl= if m &lt; 20 then shout(eg) end if m &gt; (#sort2)-20 then shout(eg) end end end </pre>		assert (u[1] 100) end
<pre>s=slurp() assert(39 == \$s.egs) assert(39 == \$s.egs) assert(39 == \$s.egs) function go.ordered( _,i,egs) i = slurp() egs = ordered(i) shout(i.heads) shout(i.heads) for j=fegs-5, #egs do shout(egs[j]) end print('#") for j=fegs-5, #egs do shout(egs[j]) end end function go.dist( i,distl,t) function distl(_,eg) return (dist(i,i.egs[l],eg), eg) end i = slurp() function distl(_,eg) return (dist(i,i.egs[l],eg), eg) end i = slurp() for j=i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print('#") for j=i,5-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print('#") for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sort1= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &lt; 20 then shout(eg) end if m &gt; (*sort2)-20 then shout(eg) end end end </pre>	160	
<pre>s=slurp() assert(39 == \$s.egs) assert(39 == \$s.egs) assert(39 == \$s.egs) function go.ordered( _,i,egs) i = slurp() egs = ordered(i) shout(i.heads) shout(i.heads) for j=fegs-5, #egs do shout(egs[j]) end print('#") for j=fegs-5, #egs do shout(egs[j]) end end function go.dist( i,distl,t) function distl(_,eg) return (dist(i,i.egs[l],eg), eg) end i = slurp() function distl(_,eg) return (dist(i,i.egs[l],eg), eg) end i = slurp() for j=i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print('#") for j=i,5-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print('#") for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sort1= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &lt; 20 then shout(eg) end if m &gt; (*sort2)-20 then shout(eg) end end end </pre>	161	function go.sample( s)
<pre>assert(398 == fs.egs) assert(3=s.lo[1]) end  function go.ordered( _,i,egs) i = slurp() egs = ordered(i) shout (l.heads) for j=fegs-5, fegs do shout(egs[j]) end print('#") for j=fegs-5, fegs do shout(egs[j]) end end  function go.dist( i,distl,t) function distl(_,eg) return (dist(i,i.egs[l],eg), eg) end i = slurp() t=map(l.egs,distl) for j=1,5 do print(j,fmt("%5.3f",t[j][l]),out(t[j][2])) end print('#") for j=(t)-5, ft do print(j,fmt("%5.3f",t[j][l]),out(t[j][2])) end print('#") for j=(t)-5, ft do print(j,fmt("%5.3f",t[j][l]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sortl= ordered(i) sortl= ordered(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (#sort2)-20 then shout(eg) end end end </pre>		
<pre>assext(3 == s.lo[1]) end function go.ordered( _,i,egs)     i = slurp()     egs = ordered(i)     shout(i.heads)     for j=1,5 do shout(egs[j]) end     print("#")     for j=tegs-5, fegs do shout(egs[j]) end end  function go.dist( i,dist1,t)     function dist1(_,eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     for j=tegs, first)     for j=tegs, f=tegs, f=teg</pre>		
<pre>function go.ordered( _,i,egs) i = slurp() egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=egs-5,#egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist1(_,eg) return (dist(i,i.egs[1],eg), eg) end it = mgo[i.egs,dist1) for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sortl= nint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (%sort2)-20 then shout(eg) end end end</pre>	163	assert (398 == #s.egs)
<pre>function go.ordered( _,i,egs) i = slurp() egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=egs-5,#egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist1(_,eg) return (dist(i,i.egs[1],eg), eg) end it = mgo[i.egs,dist1) for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(i,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sortl= nint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (%sort2)-20 then shout(eg) end end end</pre>	164	assert(3 == s.lo[1]) end
<pre>function go.ordered( _,i,egs) i = slurp() egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print('##') for j=egs-5,#egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist(_,eg) return (dist(i,i.egs[1],eg), eg) end  function funct</pre>		
<pre>i = slurp() egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=egs=5,#egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist1(_,eg) return [dist(i,i.egs[1],eg), eg) end i = slurp() for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=0,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for j=(#)-5,#t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end for j=(#)-5,#t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sortl= nint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (#sort2)-20 then shout(eg) end end end </pre>		
<pre>egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=4,5 do shout(egs[j]) end end  print("#") for j=4egs-5,#egs do shout(egs[j]) end end  function dist(i,distl,t) function dist(i,distl,t) function dist(i,eg) return (dist(i,i.egs[1],eg), eg) end i = slurp()  for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for (#") function lt(a,b) return left_is_best(i,a,b) end i=3lurp() sortl= ordered(i) sor</pre>	166	function go.ordered( _,i,egs)
<pre>egs = ordered(i) shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=4,5 do shout(egs[j]) end end  print("#") for j=4egs-5,#egs do shout(egs[j]) end end  function dist(i,distl,t) function dist(i,distl,t) function dist(i,eg) return (dist(i,i.egs[1],eg), eg) end i = slurp()  for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for (#") function lt(a,b) return left_is_best(i,a,b) end i=3lurp() sortl= ordered(i) sor</pre>	167	i = slurp()
shout(i.heads) for j=1,5 do shout(egs[j]) end print("#") for j=6gs-5,#egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist1(_,eg) return {dist(i,i.egs[1],eg), eg} end i = slurp() for j=1,5 do print(j,fmt("\$5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("\$5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("\$5.3f",t[j][1]),out(t[j][2])) end print("#") function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sort1= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m < 20 then shout(eg) end if m > (*gort2)-20 then shout(eg) end end end	400	age = ordered(i)
<pre>for j=1,5 do shout(egs[j]) end print('#") for j=#egs-5, #egs do shout(egs[j]) end end  function go.dist( i,dist1,t) function dist1(_,eg) return (dist(i,i.egs[1],eg), eg) end i = slurp() i = slu</pre>		
<pre>print("#")     for j=egas-5, #egs do shout(egs[j]) end end  function go.dist( i,dist1,t)     function dist1(_,eg) return {dist(i,i.egs[1],eg), eg} end     i = slurp()     t=map(1.egs,dist1)     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("#")     for j=(\$t)-5,#t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt)     function lt(a,b) return left_is_best(i,a,b) end     i=slurp()     sort1= ordered(i)     sort2= hint(i)     for m,eg in pairs(sort2) do         if m &lt; 20 then shout(eg) end         if m &gt; (%50rt2)-20 then shout(eg) end end end </pre>	169	
<pre>print("#")     for j=egas-5, #egs do shout(egs[j]) end end  function go.dist( i,dist1,t)     function dist1(_,eg) return {dist(i,i.egs[1],eg), eg} end     i = slurp()     t=map(1.egs,dist1)     for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("#")     for j=(\$t)-5,#t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt)     function lt(a,b) return left_is_best(i,a,b) end     i=slurp()     sort1= ordered(i)     sort2= hint(i)     for m,eg in pairs(sort2) do         if m &lt; 20 then shout(eg) end         if m &gt; (%50rt2)-20 then shout(eg) end end end </pre>	170	for i=1.5 do shout(eqs[i]) end
<pre>for j=#egs-5, #egs do shout(egs[j]) end end  function go.dist( i,dist1,t)     function dist1[_eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     t=map(1.egs,dist1)     function dist1[_eg) return (dist(i,i.egs[1],eg), eg) end     i = slurp()     t=map(1.egs,dist1)     function dist1[_eg) return (dist(i,i.egs[1],eg), eg) end     print("#")     for j=(#t)-5,#t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end     print("#")     for j=(#t)-5,#t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt)     function lt(a,b) return left_is_best(i,a,b) end     i=slurp()     sort1= ordered(i)     sort2= hint(i)     for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end     if m &gt; (#sort2)-20 then shout(eg) end end end  - Return to the operating system then number of failing demos.  - Return to the operating system then number of failing demos.  local function main(the,eg)     local no,defaults = 0,()     for k,v in pairs(the) do defaults[k]=v end     local function reset(x)     for k,v in pairs(the) do defaults[k]=v end     seed = the.seed or 10019 end     reset()     for k,v in pairs(the.todo=="all" and keys(go) or (the.todo)) do     if type(go[it]) == "function" then return print("NOFUN:",it) end     reset()     local ok,msg = peall( go[it])     if ok then print(color(32, "PASS")it)     else print(color(32, "PASS")it)     else print(color(31, "FAIL")it,msg); no=no+1 end end     roques()     os.exit(no) end  - Make 'the' options array from help string and any updates from command line.     for n.word in ipairs (arg) do if word==("",ifag) then     x = x=="false" and "fune" or tonumber(arg(n+i)) or arg(n+i) end end     the[flag]=x end)</pre>		
<pre>function go.dist( i,dist1,t) function dist1(_,eg) return {dist(i,i.egs[1],eg), eg) end i = slurp() function dist1(_,eg) return {dist(i,i.egs[1],eg), eg) end i = slurp() for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(\$t)-5,\$t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt) function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (\$tort2)-20 then shout(eg) end end end </pre>		
<pre>function go.dist( i,distl,t) function dist(i,distl,t) function dist(i,distl,t) function dist(i,distl) for j=l,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=l,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(#)-5,#t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sortl= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (%sort2)-20 then shout(eg) end end end </pre>	172	<pre>ror j=#egs-5,#egs do shout(egs[j]) end end</pre>
<pre>function disti(_reg) return (dist(i,i.egs[1],eg), eg) end i = slurp() fumap(i.egs,dist1) for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end is sort1= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (\$80rt2)-20 then shout(eg) end end end </pre>	173	
<pre>function disti(_reg) return (dist(i,i.egs[1],eg), eg) end i = slurp() fumap(i.egs,dist1) for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(*t)-5,*t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end is sort1= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (\$80rt2)-20 then shout(eg) end end end </pre>		Constitution of Albert of Albert at
<pre>i = slurp() t=mp(i.egs,dist1) for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(#)-5, #t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt) function l(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sortl= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &lt; 20 then shout(eg) end if m &lt; 20 then shout(eg) end if m &lt; 0 then shout(eg) end if m &gt; (% sort2) - 20 then shout(eg) end if m &gt; (% sort2) - 20 then shout(eg) end local function main(the,go) local function main(the,go) local no,defaults = 0,() for k,v in pairs(the) do defaults(k]=v end local function reset(x) for k,v in pairs(the) do defaults(k]=v end seed = the.seed or 10019 end reset() go (the.debug ]() for _,it in pairs(the.todo="all" and keys(go) or (the.todo)) do if type(go[it]) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = peall( go[it) local ok,msg = peall( go[it]) if ok then print (color(32, "PASS")it) so color (seed print (color(32, "FASS")it) color (seed print (color(32, "FASS")it) for k,v in pairs(the) end color (seed print (color(32, "FASS")it) for all (point maint) end reset() for "") '(spub) ("A'OPTIONS:","") '(spub) ("lw%s"-(['%s]+)") "(%s(['%s]+)", function(flag,x) for n,word in ipairs(arg) do if word==("""flag) then x = x=="false" and "fure" or tonumber(arg(n+1)) or arg(n+1) end the[flag]=x end)</pre>		
<pre>t=map(i.egs,dist1) for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(\$t)-5,\$t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end is_lurp() sort!= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (\$tort2)-20 then shout(eg) end end end </pre>	175	function distl(_,eg) return {dist(i,i.egs[1],eg), eg} end
<pre>t=map(i.egs,dist1) for j=1,5 do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end print("#") for j=(\$t)-5,\$t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end is_lurp() sort!= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (\$tort2)-20 then shout(eg) end end end </pre>	176	i = slurp()
<pre>for j=1,5 do print(j,fmt("%53f",t[j][1]),out(t[j][2])) end print("#") for j=(#1)-5,#t do print(j,fmt("%53f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sortl= ordered(i) sortl= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (#sort2)-20 then shout(eg) end end end if m &gt; (#sort2)-20 then shout(eg) end end end  Run demos, each time resetting random seed and the global config options Return to the operating system then number of failing demos. local function main(the,go) local no,defaults = 0,[) for k,v in pairs(the) do defaults[k]=v end local function reset(x) for k,v in pairs(the) do defaults[k]=v end seed = the.seed or 10019 end reset() go (the.debug ]() for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[tl)) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it) local ok,msg = pcall( go[it]) if ok then print(color(32, "PASS")it) else print(color(32, "FASS")it) sol.exit(no) end Make 'the' options array from help string and any updates from command line. the fload, x) for ,word in ipairs(arg) do if word==("-"flag) then x = x=="false" and "fune" or tonumber(arg(n+i)) or arg(n+i) end if x=="false" then x=false elseif x=="rue" then x=true end the [flag]=x end)</pre>		
<pre>print("#")     for j=(#t)-5, #t do print(j,fmt("%5.3f",t[j][1]),out(t[j][2])) end end  function go.hint( i,sort1,sort2,s,lt)     function lt(a,b) return left_is_best(i,a,b) end     i=slurp()     sort1= ordered(i)     sort2= hint(i) sirs(sort2) do     if m &lt; 20 then shout(eg) end     if m &gt; (#sort2)-20 then shout(eg) end end end </pre>		
<pre>for j=(#t)-5,#t do print(j,fmt("%5.3f",t[j[1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt)     function lt(a,b) return left_is_best(i,a,b) end     i=slurp()     sortl= ordered(i)     sort2= hint(i)     for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end     if m &gt; (%sort2)-20 then shout(eg) end end end  Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options</pre>	178	
<pre>for j=(#t)-5,#t do print(j,fmt("%5.3f",t[j[1]),out(t[j][2])) end end  function go.hint( i,sortl,sort2,s,lt)     function lt(a,b) return left_is_best(i,a,b) end     i=slurp()     sortl= ordered(i)     sort2= hint(i)     for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end     if m &gt; (%sort2)-20 then shout(eg) end end end  Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options</pre>	179	print("#")
function go.hint( i,sort1,sort2,s,lt)  function lt(a,b) return left_is_best(i,a,b) end  is=lurp()  sort1= ordered(i)  sort2= hint(i) are (sort2) do  if if m > (isort2)-20 then shout(eg) end  if m > (isort2)-20 then shout(eg) end end end		2 - 4 (84) 5 84 44-14 54 (805-268 + 14) 11) (4-14) 12) \ 4
<pre>function go.hint( i,sort1,sort2,s,lt) function lt(a,b) return left_is_best(i,a,b) end i=slurp() sort1= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &gt; (\$80rt2)-20 then shout(eg) end end end if m &gt; (\$80rt2)-20 then shout(eg) end end end </pre>		ror j=(#t)-5,#t do print(j,:mt("%3.31",t[j][1]),out(t[j][2])) end end
<pre>function lt(a,b) return left_is_best(i,a,b) end is_slurp() sortl= ordered(i) sort2 = int(i) for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end if m &gt; (#sort2) = 20 then shout(eg) end end end      if m &gt; (*sort2) = 20 then shout(eg) end end end      if m &gt; (#sort2) = 20 then shout(eg) end end end</pre>	181	
<pre>function lt(a,b) return left_is_best(i,a,b) end is_slurp() sortl= ordered(i) sort2 = int(i) for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end if m &gt; (#sort2) = 20 then shout(eg) end end end      if m &gt; (*sort2) = 20 then shout(eg) end end end      if m &gt; (#sort2) = 20 then shout(eg) end end end</pre>	182	function go.hint( i.sort1.sort2.s.lt)
<pre>i=slurp() sort1= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do     if m &lt; 20 then shout(eg) end     if m &gt; (\$20 then shout(eg) end end end     if m &gt; (\$60 tt2)-20 then shout(eg) end end end     if m &gt; (\$60 tt2)-20 then shout(eg) end end end     Run demos, each time resetting random seed and the global config options.     Return to the operating system then number of failing demos. local function main(the,eg) for k, v in pairs(the) do defaults(k]=v end local function reset(x) for k, v in pairs(the) do defaults(k]=v end seed = the.seed or 10019 end reset() go [the.debug ]() for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do     if type(go[tit) = ""inuction" then return print("NOFUN:",it) end     reset() local ok,msg = pcall( go[it]) local ok,msg = pcall( go[it]) local ok,msg = print(color(32, "PASS")it) else print(color(32, "PASS")it) solse print(color) os.exit(no) end local ok,msg = pcall( go[it]       if ok then print(color(31, "FAIL")it,msg); no=no+1 end end roques() roques() roques() rows in pairs(arg) do if word==("-", flag) then     x = x=="false" and "true" or tonumber(arg(n+1)) or arg(n+1) end end the [flag]=x end)</pre>		
<pre>sortl= ordered(i) sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end if m &lt; 20 then shout(eg) end if m &gt; (#sort2)-20 then shout(eg) end end end  Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options Run demos, each time resetting random seed and the global config options</pre>		
<pre>sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end end end if m &gt; (\$cort2)-20 then shout(eg) end end end </pre>	184	i=slurp()
<pre>sort2= hint(i) for m,eg in pairs(sort2) do if m &lt; 20 then shout(eg) end end end if m &gt; (\$cort2)-20 then shout(eg) end end end </pre>	185	sort1= ordered(i)
<pre>for m, eg in pairs(sort2) do if m &lt; 20 then shout(eg) end end end if m &gt; (#sort2)-20 then shout(eg) end end end if m &gt; (*sort2)-20 then shout(eg) end end end if m &gt; (#sort2)-20 then shout(eg) end end end </pre>		
<pre>if m &lt; 20 then shout(eg) end end end if m &gt; (%cort2)-20 then shout(eg) end end end  if m &gt; (%cort2)-20 then shout(eg) end end end </pre>		
<pre>if m &gt; (#sort2)-20 then shout(eg) end end end </pre>	187	for m,eg in pairs(sort2) do
<pre>if m &gt; (#sort2)-20 then shout(eg) end end end </pre>	188	if m < 20 then shout(eq) end
-Run demos, each time resetting random seed and the global config optionsReturn to the operating system then number of failing demos. local function main(the, go) local no, defaults = 0, {} for k, v in pairs(the) do defaults[k]=v end local function reset(x) for k, v in pairs(the) do defaults[k]=v end local function reset(x) for k, v in pairs(the the the the the the the the the the		
- Run demos, each time resetting random seed and the global config options.  - Return to the operating system then number of failing demos.  local function main(the,go)  local no,defaults = 0,()  for k,v in pairs(the) do defaults[k]=v end  local function reset(x)  for k,v in pairs(defaults) do the[k]=v end  seed = the.seed or 10019 end  reset()  go [the.debug ]()  for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do  if type(go[tl)) ~= "function" then return print("NOFUN:",it) end  reset()  local ok,msg = pcall( go[it)  for k,v in pairs(the.todo="all" and keys(go) or (the.todo)) do  if type(go[tl)) ~= "function" then return print("NOFUN:",it) end  reset()  local ok,msg = pcall( go[it])  sif ok then print(color(32,"PASS")it)  solse print(color(31,"FAIL")it,msg); no=no+1 end end  rogues()  - Make 'the' options array from help string and any updates from command line.  the promotion of the print(color(31,"FAIL")it,msg);  function(flag,x)  for ,word in ipairs(arg) do if word==("-"flag) then  x = x=="false" and "fure" or tonumber(arg(n+1)) or arg(n+1] end end  the [flag]=x end)		
Return to the operating system then number of failing demos.    local no, defaults = 0, ()	190	
Return to the operating system then number of failing demos.    local no, defaults = 0, ()	191	Run demos, each time resetting random seed and the global config options.
local function main(the,go)  local no,defaults = 0,()  for k,v in pairs(the) do defaults[k]=v end  local function reset(x)  for k,v in pairs(defaults) do the[k]=v end  local function reset(x)  for k,v in pairs(defaults) do the[k]=v end  seed = the.seed or 10019 end  reset()  go [the.debug ]()  for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do  if type(go[tl)) ~= "function" then return print("NOFUN:",it) end  reset()  local ok,msg = pcall( go[it])  for local ok,msg = pcall( go[it])  cogues() be print(color(32,"PASS")it)  cogues() be print(color(32,"FALL")it,msg); no=no+1 end end  reset()  result(no) end		
local no,defaults = 0,[]  for k,v in pairs(the) do defaults[k]=v end  local function reset(x)  for k,v in pairs(the) do defaults[k]=v end  local function reset(x)  for k,v in pairs(defaults) do the[k]=v end  Seed = the.seed or 10019 end  reset()  go[the.debug] ()  for type[go[it]) -= "function" then return print("NOFUN:",it) end  reset()  local ok,msg = peall( go[it])  if ok then print(color(32, "PASS")it)  clse print(color(32, "FASS")it)  selse print(color(31, "FAIL")it,msg); no=no+1 end end  roques()  os.exit(no) end		
<pre>for k,v in pairs(the) do defaults(k]=v end local function reset(x) for k,v vin pairs(defaults) do the[k]=v end Seed = the.seed or 10019 end reset() go (the.debug ]() for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[ith) /= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it]) for k then print(color(32, "PASS")it) coses() sprint(color(32, "PASS")it) so rogues() sprint(color(31, "FAIL")it,msg); no=no+1 end end rogues() so exit(no) end </pre>	193	
<pre>for k,v in pairs(the) do defaults(k]=v end local function reset(x) for k,v vin pairs(defaults) do the[k]=v end Seed = the.seed or 10019 end reset() go (the.debug ]() for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[ith) /= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it]) for k then print(color(32, "PASS")it) coses() sprint(color(32, "PASS")it) so rogues() sprint(color(31, "FAIL")it,msg); no=no+1 end end rogues() so exit(no) end </pre>	194	<pre>local no,defaults = 0,{}</pre>
<pre>local function reset(x) for k,v in pairs(defaults) do the[k]=v end Seed = the.seed or 10019 end reset() go[ the.debug ]() for _,t in pairs(the.todo=="all" and keys(go) or (the.todo)) do     if type(go[it]) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall ( go[it] )     if ok then print(color(32,"PASS")it)         else print(color(31,"FAIL")it,msg); no=no+1 end end roques()     os.exit(no) end </pre>	405	
for k,v in pairs(defaults) do the [k]=v end seed = the.seed or 10019 end reset() go [the.debug ]() for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[th) /= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it]) if ok then print(color(32, "PASS")it) else print(color(32, "PASS")it,msg); no=no+1 end end rogues() os.exit(no) end os.exit(no) end Make 'the' options array from help string and any updates from command line. (help or ""):gsub("A-OPTIONS:",""):gsub("hu%s"-[(^%s]+]/^n]*%s((^%s]+)", function(flag,x) for n, word in ipairs(arg) do if word==("-".flag) then x = x=="false" and "frue" or tonumber(arg[n+1]) or arg[n+1] end end the[flag]=x end)		
Seed = the.seed or 10019 end reset() go[ the.debug ]() for _,ti n pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[it]) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall (go[it]) if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end roques() os.exit(no) end	196	iocal runction reset(X)
Seed = the.seed or 10019 end reset() go[ the.debug ]() for _,ti n pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[it]) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall (go[it]) if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end roques() os.exit(no) end	197	for k,v in pairs (defaults) do the[k]=v end
reset() go [the.debug]() for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[th) -= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it]) if ok then print(color(32, "PASS")it) else print(color(31, "FALL")it,msg); no=no+1 end end rogues() os.exit(no) end Make 'the' options array from help string and any updates from command line. help or ""):gsubi("n^*OPTIONS:",""):gsub("n\ms*-[(^ms]+]\mn]\ms*(s(\ms*s)+)", function(flag,x) for n,word in ipairs(arg) do if word==("-".flag) then x = x=="false" and "frue" or tonumber(arg[n+1]) or arg[n+1] end end if x=="false" then x=false elseif x=="frue" then x=true end the[flag]=x end)	198	Seed = the seed or 10019 end
<pre>go[ the.debug ]() for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do if type(go[it]) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it]) if ok then print(color(32,"PASS")it) else print(color(31,"FAIL")it,msg); no=no+1 end end roques() os.exit(no) end </pre>		
<pre>for _,it in pairs(the.todo=="all" and keys(go) or (the.todo)) do    if type(go[tit]) == "(unction" then return print("NOFUN:",it) end    reset()  local ok,msg = pcall( go[it])  fok then print(color(32, "PASS")it)  slee print(color(32, "PASS")it)  sos.exit(no) end </pre>		
<pre>if type(go[it]) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it] ) if ok then print(color(32,"PASS")it) else print(color(31,"FAIL")it,msg); no=no+1 end end roques() roques() os.exit(no) end </pre>	200	go[ the.debug ]()
<pre>if type(go[it]) ~= "function" then return print("NOFUN:",it) end reset() local ok,msg = pcall( go[it] ) if ok then print(color(32,"PASS")it) else print(color(31,"FAIL")it,msg); no=no+1 end end roques() roques() os.exit(no) end </pre>	201	for it in pairs (the todo=="all" and keys (go) or (the todo)) do
reset()  local ok,msg = pcall(go[it])  local ok,msg = pcall(go[it]		
local ok_msg = pcall( go[it]) if ok then print(color(32, "PASS")it) cos else print(color(32, "PASS")it) roques() cos.exit(no) end		
if ok then print(color(32, "PASS")it)  else print(color(31, "FAIL")it,msg); no=no+1 end end  roques()  os.exit(no) end	203	
if ok then print(color(32, "PASS")it)  else print(color(31, "FAIL")it,msg); no=no+1 end end  roques()  os.exit(no) end	204	local ok meg = posli ( go[i+] )
else print(color(31, "FAIL ")it,msg); no=no+1 end end roques() os.exit(no) end	205	
rogues()  ose os.exit(no) end		
rogues()  ose os.exit(no) end		if ok then print (color(32, "PASS")it)
os.exit(no) end	206	<pre>if ok then print(color(32, "PASS")it)     else print(color(31, "FAIL")it, msg); no=no+1 end end</pre>
Make 'the' options array from help string and any updates from command line.  (help or ""):gsub("^A°OPTIONS:",""):gsub("\n%s*-[[^Ks]+)[^Mn]*%s([^Ks]+)",  for n, word in ipairs (arg) do if word=("-"flag) then  x = x=="flase" and "Tune" or tonumber(arg[n+1]) or arg[n+1] end end  if x=="flase" then x=false elseif x=="true" then x=true end  the[flag]=x end)	206	<pre>if ok then print(color(32, "PASS")it)     else print(color(31, "FAIL")it, msg); no=no+1 end end</pre>
<pre>200 Make 'the' options array from help string and any updates from command line. 211 (help or ""):gsub("n*OptiONS:",""):gsub("in%*-[(n*s -)]"\ni]"%s([n*s +)", 212 function(flag,x) 213 for n, word in ipairs(arg) do if word=-(""flag) then 214</pre>	206 207	<pre>if ok then print(color(32, "PASS")it)   else print(color(31, "FAIL")it,msg); no=no+1 end end rogues()</pre>
<pre>211 (help or ""):gsub("^.^OPTIONS:",""):gsub("\n%s*-([^%s]+)]^\n]\%s([^%s]+)", 212</pre>	206 207 208	<pre>if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end rogues() os.exit(no) end</pre>
<pre>211 (help or ""):gsub("^.^OPTIONS:",""):gsub("\n%s*-([^%s]+)]^\n]\%s([^%s]+)", 212</pre>	206 207 208 209	<pre>if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end rogues() os.exit(no) end</pre>
function(flag,x) for n,word in ipairs(arg) do if word==("-"flag) then for n,word in ipairs(arg) do if word==("-"flag) then x = x=="flaks" and "true" or tonumber(arg[n+1]) or arg[n+1] end end if x=="flaks" then x=false elseif x=="true" then x=true end the[flag]=x end)  the[flag]=x end)	206 207 208 209	<pre>if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end rogues() os.exit(no) end</pre>
for n,word in ipairs(arg) do if word==("-"flag) then  x = x=="flake" and "true" or tonumber(arg[n+1]) or arg[n+1] end end  if x=="flake" then x=false elseif x=="true" then x=true end  the[flag]=x end)	206 207 208 209 210	if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end roques() os.exit(no) end Make 'the' options array from help string and any updates from command line.
<pre>z= x == "false" and "true" or tonumber(arg[n+1]) or arg[n+1] end end z= if x == "false" then x=false elseif x== "true" then x=true end z= the[flag]=x end)</pre>	206 207 208 209 210 211	if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end rogues() os.exit(no) end  Make 'the' options array from help string and any updates from command line. (help or ""):gsub("^.OPTIONS:", ""):gsub("um%s"-([^%s]+) ^nu]*%s[(7%s]+)",
<pre>z= x == "false" and "true" or tonumber(arg[n+1]) or arg[n+1] end end z= if x == "false" then x=false elseif x== "true" then x=true end z= the[flag]=x end)</pre>	206 207 208 209 210 211 212	if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end roques() os.exit(no) end  Make 'the' options array from help string and any updates from command line. (help or ""):gsub("^.*OPTIONS:", ""):gsub("\n"\s**-([^\ark S]+)[^\angle \n"\s**([\ark S]+)", function(flag,x)
<pre>215    if x=="false" then x=false elseif x=="true" then x=true end 216    the[flag]=x end) 217</pre>	206 207 208 209 210 211 212	if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end roques() os.exit(no) end  Make 'the' options array from help string and any updates from command line. (help or ""):gsub("^.*OPTIONS:", ""):gsub("\n"\s**-([^\ark S]+)[^\angle \n"\s**([\ark S]+)", function(flag,x)
216 the[flag]=x <b>end</b> ) 217	206 207 208 209 210 211 212 213	<pre>if ok then print(color(32, "PASS")it)     else print(color(31, "FAIL")it,msg); no=no+1 end end rogues() os.exit(no) end Make 'the' options array from help string and any updates from command line. (help or ""):gsub("^."OPTIONS:", ""):gsub("\m%s"-([\%s]+)\\n])\%s([\%s]+)", function(flag,x)     for n,word in ipairs(arg) do if word==("-"flag) then</pre>
217	206 207 208 209 210 211 212 213 214	if ok then print(color(32, "PASS")it) else print(color(31, "FAIL")it,msg); no=no+1 end end roques() os.exit(no) end  Make 'the' options array from help string and any updates from command line. (help or ""):gsub("^.*OPTIONS:", ""):gsub("Mr.%s*-([-4%s+])"An]*%s([-%s]+)", function(flag,x) for n,word in ipairs(arg) do if word== ("-"flag) then x = x=="false" and "fure" or tonumber(arg[n+1]) or arg[n+1] end end
217	206 207 208 209 210 211 212 213 214 215	<pre>if ok then print(color(32, "PASS")it)</pre>
	206 207 208 209 210 211 212 213 214 215	<pre>if ok then print(color(32, "PASS")it)</pre>
218 II the.m then print(neip) eise main(the,go) end	206 207 208 209 210 211 212 213 214 215 216	<pre>if ok then print(color(32, "PASS")it)</pre>
	206 207 208 209 210 211 212 213 214 215 216 217	if ok then print(color(32, "PASS")it)  else print(color(31, "FAIL")it,msg); no=no+1 end end rogues() os.exit(no) end  Make 'the' options array from help string and any updates from command line.  (help or "").rgsub("."OPTIONS:","").rgsub("um%s"=(["%s]+)"\n] "%s[[%s]+)", for n,word in inpairs(arg) do if word= (""-"flag) then  x = x=="false" and "true" or tonumber(arg[n+1]) or arg[n+1] end end if x=="false" then x=false elseif x=="true" then x=true end  the[flag]=x end)