When you operate the element in the table, notice that not to change the element in the old table you refer.

If use t2 = t to make element in table t2 equal to t1.

When you change the element in t2,then when you print(t) you will find the element in the t also changed.

local x=0

    local g = {}

    for i,v in ipairs(t) do

         table.insert(g,v);

    end

if you want to change this situation and you can use the way up this line to define t2.

-- the number of element in the table is not explicit ,and 0 can represent any number. To judge whether the content output is a [arithmetic progression](http://www.baidu.com/link?url=4HaKnUqzc7SNfz837AWeFAobSzSo-CYKqZ_-XAW4T0DtP6lXDZ0Aapv45j3i6s0sWMlDK79jHrbaP7qhBtdbO8pFV649LhXVsiZ8AXnfUAz78I674VTfPWaIQm_yc1Kc)

function istableorder(t,a,b)

local x=0

    local g = {}

    for i,v in ipairs(t) do

         table.insert(g,v);

    end

c={}

d={}

for i,v in pairs(t) do

if v~=0 then

x=x+1

table.insert(c,#c+1,i)     --note the not 0 number’s positon

table.insert(d,#d+1,v)     --and the value

end

end

 -- print(unpack(d))

 -- print(unpack(c))

if x==0 then

   if(#t<=(b-a+1)) then

      return true

   else

return false

   end

elseif x==1 then

    if c[1]==1 or c[1]==#t then

if ((d[1]+#t-1)>(b)) then

                    return false

                elseif((d[1]-#t+1)<a) then

                    return false

                end

                return true

    elseif(d[1]+(#t-c[1])<=b)and(d[1]-(c[1]-1)>=a) then            return true

    elseif

(d[1]-(#t-c[1])>=a)and(d[1]+(c[1]-1)<=b) then

                return true

            else

                return false

            end

        else

            local e

            e=(d[2]-d[1])/(c[2]-c[1])

            if c[1]==1 then

                for i=1,#t-1 do

                    if g[i+1]~=g[i]+e then

                        if g[i+1]~=0 then

                            return false

                        else

                            g[i+1]=g[i]+e

                            if g[i+1]>b or g[i+1]<a then

                                return false

                            end

                        end

                    end

                end

                return true

            elseif c[2]==#t then

                for i=#t,2,-1 do

                    if g[i-1]~=g[i]-e then

                        if g[i-1]~=0 then

                            return false

                        else

                            g[i-1]=g[i]-e

                            if g[i-1]>b or g[i-1]<a then

                                return false

                            end

                        end

                    end

                end

            return true

        --     else

        --         for i=c[1],2,-1 do

        --             if g[i-1]~=g[i]-e then

        --                 if g[i-1]~=0 then

        --                     return false

        --                 else

        --                     g[i-1]=g[i]-e

        --                     if g[i-1]>b or g[i-1]<a then

        --                         return false

        --                     end

        --                 end

        --             end

        --         end

        --         for i=c[1],#t-1 do

        --             if g[i+1]~=g[i]+e then

        --                 if g[i+1]~=0 then

        --                     return false

        --                 else

        --                     g[i+1]=g[i]+e

        --                     if g[i+1]>b or g[i+1]<a then

        --                         return false

        --                     end

        --                 end

        --             end

        --         end

        --         return true           --]]

            end

           end

end

-- a=6

-- b=7

-- t={5,0,0}

-- r={0,1,0,0,0,9}

-- s={0,0,0,0,5,0,0}

-- u={0,0,0,1,0,0,0,}

-- v={0,0,1,0,3}

-- w={0,0,3,0,1}

-- x={-1,0,0,1,0,3}

-- y={-1,0,1,2,0,0,5}

-- z={-2,0,0,0,0,0,0,}

-- sr={0,0,0,4,0,0,0,0,-1}

-- function shuchu( t )

--

print(unpack(t))

-- end

-- print(shuchu(t),istableorder(t,a,b))

-- print(shuchu(r),istableorder(r,a,b))

-- print(shuchu(s),istableorder(s,a,b))

-- print(shuchu(u),istableorder(u,a,b))

-- print(shuchu(v),istableorder(v,a,b))

-- print(shuchu(w),istableorder(w,a,b))

-- print(shuchu(x),istableorder(x,a,b))

-- print(shuchu(y),istableorder(y,a,b))

-- print(shuchu(z),istableorder(z,a,b))

-- print(shuchu(sr),istableorder(sr,a,b))

slot={

{1,2,3,4,5,6,7,8,0,0,0,0,0,0,0,0},

{1,2,3,4,5,6,7,7},

{-3,-1,1,3,5},

{-3,-1,1,3,4},

{1,0,2,5},

{1,0,3,4},

{0,0,0,1,3,5},

{0,0,1,2,0,4,5},

{0,0,1,2,0,0,4},

{0,0,1,2,0,0,5},

{0,0,0,1,0,0,0},

{0,0,0,0,0},

{1,5,6},

{1,5,9},

{5,4,3,2,1},

{2,0,0},

{0,0,0,0,0},

{1,0,0,0,0},

{0,0,0,0,1},

{0,0,1,0,0},

{0,1,0,3,0},

{1,2,0,0,0},

{0,0,0,1,2},

{1,2,3,4,5},

{2,4,6,8,10},

{3,5,7,0,11},

{-9,-7,0,0,-1},

{-9,0,0,0,-1},

{0,-7,-5,0,-1},

{0,0,0,-3,0},

{-9,0,0,0,0},

{0,0,0,0,-1},

{-9,0,-5,0,-1},

{0,-9,-7,0,0},

}

a=-1

b=3

for i=1,table.getn(slot) do

temp = slot[i];

if(istableorder(temp,a,b)) then

--print("ture|"..table.concat(temp,",").."}".." is order");

print("true".."  "..table.concat(temp,',').."  a="..a.." b="..b)

else

--print("false|"..table.concat(temp,",").."}".." is not order");

print("false".."  "..table.concat(temp,',').."  a="..a.." b="..b)

end

end

test={0,0,0,1,2}

istableorder(test,a,b)

print(unpack(test))

Baidu Button BEGIN