Constraint programming

Outline for solution not perfect

Def bruteForce(pzl)

Return solved pzl, or empty string on failure

Make the pzl a string or ur dumb

If pzl is completely filled out then return ‘’ if puzzle is invalid otherwise pzl

Find setOfChoices that are collectively exhaustive #may come back to this

For each possible choice in setOfChoices:

Subpzl = pzl with possibleChoice applied #possible choice = put this label at this position, put this block here, etc

bruteForce(subpzl)

if bf: return bf # success

else return “” for failure

Def better(pzl)

Return solved pzl, or empty string on failure

Make the pzl a string or ur dumb

If pzl if puzzle is invalid return “”

If solved return pzl

Find setOfChoices that are collectively exhaustive #may come back to this

For each possible choice in setOfChoices:

Subpzl = pzl with possibleChoice applied #possible choice = put this label at this position, put this block here, etc

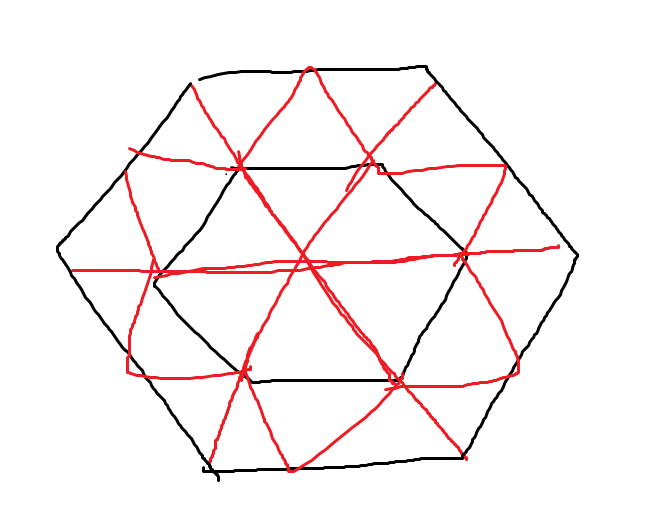
bruteForce(subpzl)

if bf: return bf # success

else return “” for failure

homework due Friday:

no more than 2 hours to solve else bring in and show code w known bug

space like w 6 labels – attach one of these labels to the little triangles so that none of the 7 complete hexagons has the label repeated within it

2: want to know if its possible so that w 7 labels its possible to have no common labels in the 7 hexagons or within the horizontal, pos diagonal, neg diagonal rows