Lab 07 CSP

Name: Ahmed Kasteer

ROll number: 20F-0336

Section: 6C

#domains of all the regions with given choices of colors  
  
domains = {  
 'WA': ['red', 'green', 'blue'],  
 'NT': ['red', 'green', 'blue'],  
  
 'Q': ['red', 'green', 'blue'],  
 'NSW': ['red', 'green', 'blue'],  
  
 'V': ['red', 'green','blue'],  
 'SA': ['red', 'green','blue'],  
  
 'T': ['red', 'green','blue']  
}  
  
constraints = {  
  
  
  
  
 ('WA', 'NT'): lambda a, b: a != b,  
 ('NT', 'WA'): lambda b, a: b != a,  
  
 ('WA', 'SA'): lambda a, b: a != b,  
 ('SA', 'WA'): lambda b, a: b != a,  
  
 ('NT', 'Q'): lambda a, b: a != b,  
 ('Q', 'NT'): lambda b, a: b != a,  
  
 ('Q', 'NSW'): lambda a, b: a != b,  
 ('NSW', 'Q'): lambda b, a: b != a,  
  
 ('Q', 'SA'): lambda a, b: a != b,  
 ('SA', 'Q'): lambda b, a: b != a,  
  
 ('NSW', 'V'): lambda a, b: a != b,  
 ('V', 'NSW'): lambda b, a: b != a,  
  
 ('SA', 'V'): lambda a, b: a != b,  
 ('V', 'SA'): lambda b, a: b != a,  
  
 ('SA', 'NSW'): lambda a, b: a != b,  
 ('NSW', 'SA'): lambda b, a: b != a,  
  
 ('SA', 'NT'): lambda a, b: a != b,  
 ('NT', 'SA'): lambda b, a: b != a,  
  
}  
  
  
  
  
def revise(x, y):  
 *"""*  
 *Make variable `x` arc consistent with variable `y`.*  
 *To do so, remove values from `domains[x]` for which there is no*  
 *possible corresponding value for `y` in `domains[y]`.*  
 *Return True if a revision was made to the domain of `x`; return*  
 *False if no revision was made.*  
 *"""*  
revised = False  
  
 # Get x and y domains  
 x\_domain = domains[x]  
 y\_domain = domains[y]  
  
 # Get all arc (x, y) constraints  
 all\_constraints = [  
 constraint for constraint in constraints if constraint[0] == x and constraint[1] == y]  
  
 for x\_value in x\_domain:  
 satisfies = False  
 for y\_value in y\_domain:  
 for constraint in all\_constraints:  
 constraint\_func = constraints[constraint]  
 if constraint\_func(x\_value, y\_value):  
 satisfies = True  
 if not satisfies:  
 x\_domain.remove(x\_value)  
 revised = True  
  
  
 return revised  
  
  
def ac3(arcs):  
 *"""*  
 *Update `domains` such that each variable is arc consistent.*  
 *"""*  
# Add all the arcs to a queue.  
 queue = arcs[:]  
 # Repeat until the queue is empty  
 while queue:  
 # Take the first arc off the queue (dequeue)  
 (x, y) = queue.pop(0)  
  
 # Make x arc consistent with y  
  
 revised = revise(x, y)  
  
 # If the x domain has changed  
 if revised:  
 # Add all arcs of the form (k, x) to the queue (enqueue)  
 neighbors = [neighbor for neighbor in arcs if neighbor[1] == x]  
 queue = queue + neighbors  
 # Assign a single color to each region  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 arcs = [('WA', 'NT'), ('NT', 'WA'),  
 ('WA', 'SA'), ('SA', 'WA'),  
  
 ('NT', 'Q'), ('Q', 'NT'),  
 ('Q', 'NSW'), ('NSW', 'Q'),  
  
 ('Q', 'SA'), ('SA', 'Q'),  
 ('NSW', 'V'), ('V', 'NSW'),  
  
 ('SA', 'V'), ('V', 'SA'),  
 ('SA', 'NSW'), ('NSW', 'SA'),  
 ('SA', 'NT'), ('NT', 'SA')]  
 ac3(arcs)  
  
 print(domains)

