**程序报告**

学号： 2113662 姓名：张丛

1. **问题重述**

（斑马问题）

====================================================================

五个不同国家且工作各不相同的人分别住在一条街上的五所房子里，每所房子颜色不同，每个人都有自己养的不同的宠物，喜欢喝不同的饮料。根据提示，判断哪所房子里的人养斑马，哪所房子里的人喜欢喝矿泉水。

逻辑约束求解问题，给出的14条文字提示就是14个逻辑约束与限制，当给出的逻辑足够多的时候，便可以根据逻辑推理将最后的逻辑唯一确定，从而回答问题。

1. **设计思想**

====================================================================

使用python逻辑编程kanren库,将题目的条件化为一个个表达式，加入约束到kanren的一个集合中，最后利用kanren内置的run求解即可

# 等价关系格式一: eq(var(), value) / eq(var(), var())

x = var() # 变量声明，kanren的推理基于变量var进行

z = var()

run(0, x, eq(x, z), eq(z, 3)) # 规则求解器，kanren的推理通过run函数进行

# 格式要求为: run(n, var(), rules,[rules, ...])

# 求解指定规则下符合的变量结果

# 等价关系格式二: (eq, var(), value) / (eq, var(), var())

x = var()

z = var()

run(0, x, (eq, x, z), (eq, z, 3))

# 属于关系格式 membero(var(), list / tuple)

x = var()

run(0, x, membero(x, (1, 2, 3)), # x is a member of (1, 2, 3) #x是（1,2,3）的成员之一

membero(x, (2, 3, 4))) # x is a member of (2, 3, 4) #x是（2,3,4）的成员之一

# 逻辑和关系格式 conde((rules, rules))

x = var()

run(0, x, conde((membero(x, (1, 2, 3)), membero(x, (2, 3, 4)))))

# 逻辑或关系格式 conde([rules], [rules]))

x = var()

run(0, x, conde([membero(x, (1, 2, 3))], [membero(x, (2, 3, 4))]))

# 调用lall包定义规则集合, lall(rules, [rules, ...])

x = var()

z = var()

rules = lall(

eq(x, z),

eq(z, 3)

)

run(0, x, rules)

# 调用lall包定义规则集合, lall(rules, [rules, ...])

x = var()

z = var()

rules = lall(

(eq, x, z),

(eq, z, 3)

)

run(0, x, rules)

====================================================================

1. **代码内容**

====================================================================

from kanren import run, eq, membero, var, conde # kanren一个描述性Python逻辑编程系统

from kanren.core import lall # lall包用于定义规则

import time

def left(q, p, list): #表示在关系集合list中q在p的左侧

return membero((q, p), zip(list, list[1:]))

def next(q, p, list): #表示在关系集合中p和q相邻

return conde([left(q, p, list)], [left(p, q, list)])

class Agent:

def \_\_init\_\_(self):

self.units = var() # 单个unit变量指代一座房子的信息(国家，工作，饮料，宠物，颜色)

# 例如('英国人', '油漆工', '茶', '狗', '红色')即为正确格式，但不是本题答案

# 请基于给定的逻辑提示求解五条正确的答案

self.rules\_zebraproblem = None # 用lall包定义逻辑规则

self.solutions = None # 存储结果

def define\_rules(self):

"""

定义逻辑规则.

"""

self.rules\_zebraproblem = lall(

(eq, (var(), var(), var(), var(), var()), self.units),

(membero, ('英国人', var(), var(), var(), '红色'), self.units), # 英国人住在红房子里

(membero, ('西班牙人', var(), var(), '狗', var()), self.units), # 西班牙人养了一条狗

(membero, ('日本人', '油漆工', var(), var(), var()), self.units), # 日本人是一个油漆工

(membero, ('意大利人', var(), '茶', var(), var()), self.units), # 意大利人喝茶。

(eq, (('挪威人', var(), var(), var(), var()), var(), var(), var(), var()), self.units), # 挪威人住在左边的第一个房子里

(membero, (var(), '摄影师', var(), '蜗牛', var()), self.units), # 摄影师养了一只蜗牛

(membero, (var(), '外交官', var(), var(), '黄色'), self.units), # 外交官住在黄房子里

(eq, (var(), var(),(var(), var(), '牛奶', var(), var()),var(), var()),self.units),

(membero, (var(), var(), '咖啡', var(), '绿色'), self.units), # 喜欢喝咖啡的人住在绿房子里

(membero, (var(), '小提琴家', '橘子汁', var(), var()), self.units), # 小提琴家喜欢喝橘子汁

(membero,(var(), var(), var(), '斑马', var()), self.units),

(membero,(var(), var(), '矿泉水', var(), var()), self.units),

(left, (var(), var(), var(), var(), '白色'), (var(), var(), var(), var(), '绿色'),self.units), # 绿房子在白房子的右边

(next, ('挪威人', var(), var(), var(), var()),

(var(), var(), var(), var(), '蓝色'), self.units), # 挪威人住在蓝房子旁边。

(next, (var(), '医生', var(), var(), var()), # 养狐狸的人所住的房子与医生的房子相邻

(var(), var(), var(), '狐狸', var()), self.units),

(next, (var(), '外交官', var(), var(), var()), # 养马的人所住的房子与外交官的房子相邻

(var(), var(), var(), '马', var()), self.units)

)

def solve(self):

self.define\_rules()

self.solutions = run(0, self.units, self.rules\_zebraproblem)

return self.solutions

agent = Agent()

solutions = agent.solve()

# 提取解释器的输出

output = [house for house in solutions[0] if '斑马' in house][0][4]

print ('\n{}房子里的人养斑马'.format(output))

output = [house for house in solutions[0] if '矿泉水' in house][0][4]

print ('{}房子里的人喜欢喝矿泉水'.format(output))

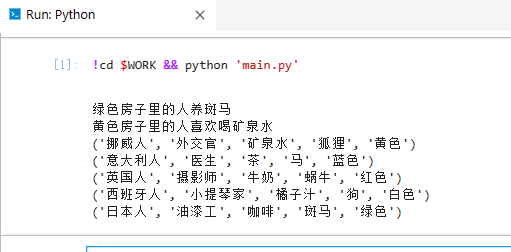
# 解释器的输出结果展示

for i in solutions[0]:

print(i)

1. **实验结果**

====================================================================





1. **总结**

====================================================================

实验达到了预期。

基本掌握了逻辑编程的思想，进一步学习了python语言。

实验遇到的困难有：之前没系统学习过python语言，需要学习python的基本语法、结构等；编写left和next函数表示左边和邻近关系；14条逻辑约束，在编写时容易出现细节上的错误。

改进方向：14条（甚至更多）逻辑的输入是否可以更简？是否可更高效实现逻辑约束？