**程序报告**

学号： 2113997 姓名：齐明杰

1. **问题重述**

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

斑马问题： 5 个不同国家（英国、西班牙、日本、意大利、挪威）且工作各不相同（油漆工、摄影师、外交官、小提琴家、医生）的人分别住在一条街上的 5 所房子里，

每所房子的颜色不同（红色、白色、蓝色、黄色、绿色），每个人都有自己养的不同宠物（狗、蜗牛、斑马、马、狐狸），喜欢喝不同的饮料（矿泉水、牛奶、茶、橘子汁、咖啡）。

根据以下提示，你能告诉我哪所房子里的人养斑马，哪所房子里的人喜欢喝矿泉水吗？

英国人住在红色的房子里

西班牙人养了一条狗

日本人是一个油漆工

意大利人喜欢喝茶

挪威人住在左边的第一个房子里

绿房子在白房子的右边

摄影师养了一只蜗牛

外交官住在黄房子里

中间那个房子的人喜欢喝牛奶

喜欢喝咖啡的人住在绿房子里

挪威人住在蓝色的房子旁边

小提琴家喜欢喝橘子汁

养狐狸的人所住的房子与医生的房子相邻

养马的人所住的房子与外交官的房子相邻

1. **设计思想**

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

演绎推理（Deductive Reasoning）是由一般到特殊的推理方法。与“归纳法”相对。推论前提与结论之间的联系是必然的，是一种确实性推理。本题我们使用python逻辑编程kanren库，将题目中的条件形式化为一个个的表达式，加入约束到kanren的一个集合中，然后利用kanren内置的run求解即可得到问题的解。

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):

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

def next(q, p, list): # p q 相邻意味着要么p在q左边，要么q在p左边

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

class Agent:

"""

推理智能体.

"""

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

"""

智能体初始化.

"""

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), # self.units共包含五个unit成员，即每一个unit对应的var都指代一座房子(国家，工作，饮料，宠物，颜色)

# 各个unit房子又包含五个成员属性: (国家，工作，饮料，宠物，颜色)

(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

(left, # 绿房子在白房子的右边

(var(), var(), var(), var(), '绿色'),

(var(), var(), var(), var(), '白色'),

self.units),

(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), # 中间那个房子的人喜欢喝牛奶

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

(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。

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

(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),

(membero, (var(), var(), var(), '斑马', var()), self.units), # 有人养斑马

(membero, (var(), var(), '矿泉水', var(), var()), self.units) # 有人喝水

)

def solve(self):

"""

规则求解器(请勿修改此函数).

return: 斑马规则求解器给出的答案，共包含五条匹配信息，解唯一.

"""

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**

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

在实验代码中填入相应的逻辑语句后点击运行，结果如下：

绿色房子里的人养斑马

黄色房子里的人喜欢喝矿泉水

('挪威人', '外交官', '矿泉水', '狐狸', '黄色')

('意大利人', '医生', '茶', '马', '蓝色')

('英国人', '摄影师', '牛奶', '蜗牛', '红色')

('日本人', '油漆工', '咖啡', '斑马', '绿色')

('西班牙人', '小提琴家', '橘子汁', '狗', '白色')

1. **总结**

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

通过这次实验，我基本掌握逻辑编程的思想，了解了逻辑编程与命令式编程的区别，同时能够依据给定的事实以及规则编写代码，解决逻辑约束问题（CLP），也熟悉了 Python 语言