斑　马　问　题

实验一

Zebra Problem

实验时间：2024年3月18日

截止时间：2024年4月1日

CONTENT

目　录

问题描述

Problem description

问题描述
Problem description

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？

1. 英国人住在红色的房子里
2. 西班牙人养了一条狗
3. 日本人是一个油漆工
4. 意大利人喜欢喝茶
5. 挪威人住在左边的第一个房子里
6. 绿房子在白房子的右边
7. 摄影师养了一只蜗牛
8. 外交官住在黄房子里
9. 中间那个房子的人喜欢喝牛奶
10. 喜欢喝咖啡的人住在绿房子里
11. 挪威人住在蓝色的房子旁边
12. 小提琴家喜欢喝橘子汁
13. 养狐狸的人所住的房子与医生的房子相邻
14. 养马的人所住的房子与外交官的房子相邻

参考资料：斑马难题-百度百科
(https://baike.baidu.com/item/斑马难题/3709972?fr=aladdin)

2

实验基础

Experimental Basis

实验基础
Experimental Basis

- **kanren：**(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

README License

kanren

build unknown

Logic Programming in Python

Examples

kanren enables the expression of relations and the search for values which satisfy them. The following code is the "Hello, world!" of logic programming. It asks for `1` number, `x`, such that `x == 5`

```
>>> from kanren import run, eq, membero, var, conde
>>> x = var()
>>> run(1, x, eq(x, 5))
(5,)
```

实验基础

Experimental Basis

- **kanren**：(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

- 变量声明：var()
- 规则求解器：run(n, var(), rules,[rules, ...])
- 等价关系表达式：eq(x, y)，其意即为变量x等价于变量y。
  - 等价关系格式一：eq(var(), value) / eq(var(), var())

```
# 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
#-------------------------------------
Output: (3,)
```

实验基础

Experimental Basis

- **kanren**：(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

- 变量声明：var()
- 规则求解器：run(n, var(), rules,[rules, ...])
- 等价关系表达式：eq(x, y)，其意即为变量x等价于变量y。
  - 等价关系格式二： (eq, var(), value) / (eq, var(), var())

```
# 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))
#-------------------------------------
Output: (3,)
```

实验基础

Experimental Basis

- **kanren**：(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

- 成员关系表达式：membero(var(), list / tuple)

```
# 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
#-------------------------------------
Output: (2, 3)
```

实验基础

Experimental Basis

- **kanren**：(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

- **逻辑和/或的目标构造函数：**

  逻辑和关系格式 conde(**(rules, rules)**)

```
# 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)))))
#-------------------------------------
Output: (2, 3)
```

实验基础

Experimental Basis

- **kanren**：(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

- **逻辑和/或的目标构造函数：**

  逻辑或关系格式 conde([rules], [rules])

```
# 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))]))
#-------------------------------------
Output: (1, 2, 3, 4)
```

实验基础

Experimental Basis

- **kanren**：(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

- 使用lall包定义规则集合： lall(rules, [rules, ...])

等价关系格式一:

```
# 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)
#-------------------------------------
Output: (3,)
```

实验基础
Experimental Basis

- **kanren**：(https://github.com/logpy/logpy) 是 Python 的一个逻辑编程包。
  - 可选择该逻辑编程包或自定义逻辑语法

```
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
```

- 使用lall包定义规则集合： lall(rules, [rules, ...])

等价关系格式二:

```
# 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)
#----------------------------------
Output: (3,)
```

3

代码实现

Code implementation

代码实现
Code implementation

TODO: 邻近规则如何定义，

e.g., 6. 绿房子在白房子的右边

11. 挪威人住在蓝色的房子旁边

自定义规则：左邻近规则left(), 右邻近规则right(),邻近规则next()

```
#############################################################################
####                可在此处定义自己所需要用到的自定义函数(可选)                  ####
####    提示：定义左邻近规则left(), 定义右邻近规则right(),定义邻近规则next()    ####
#############################################################################
#                                                                           #


#                                                                           #
#############################################################################
#################                  非必要性工作                  ################
#############################################################################
```

代码实现

Code implementation

TODO: 添加题给的14条逻辑规则

```
houses = var()
rules_zebraproblem = lall(
            (eq, (var(), var(), var(), var(), var()), houses),
            # houses共包含五个house成员，一个house对应的var都指代一座房子(国家，工作，饮料，宠物，颜色) ，
            # 各个house又包含五个成员属性:(国家，工作，饮料，宠物，颜色)

            # 示例：基于问题信息可以提炼出，有人养斑马，有人喜欢和矿泉水等信息
            (membero,(var(), var(), var(), '斑马', var()), houses),
            (membero,(var(), var(), '矿泉水', var(), var()), houses),

        )
```

4

结果输出

Result output

结果输出

Result output

完成 Agent Cell 后，在左侧**提交结果**的标签中，把整个 Agent Cell 转化为 main.py 文件进行**系统测试**。能通过测试就可以**提交结果**。

通过solve函数（勿修改）进行输出，输出格式要求如下：
（一）提取解算器的输出：哪所房子里的人养斑马，哪所房子里的人喜欢喝矿泉水？

（二）解算器的输出结果展示
1. 输出正确的五条匹配信息；
2. 每条匹配信息依次包含(国家，工作，饮料，宠物，颜色)五个元素；
3. 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。

谢谢