**تمرین 1: ربات فوتبالیست**

* **کارآیی** : گل زدن ، دفاع کردن ، برد بازی ، انجام حرکات مختلف ، خطا نکردن در طول بازی
* **محیط** : نوع زمین ، حضور توپ فوتبال ، حضور بازیکنان دیگر ، شرایط آب و هوایی
* **عملگر**: شوت کردن توپ ، پاس دادن توپ ، گرفتن توپ با دست توسط دوازه بان و ...
* **سنسور**: سنسور تشخیص توپ ، سنسور تشخیص فاصله ، سنسور تشخیص مرز زمین فوتبال ،سنسور تشخیص سرعت توپ و جهت آن ، سنسور آنالیز حرکات بازیکن و تشخیص بازیکنان ، سنسور عوامل محیطی

**تمرین 2: تمرین 8 وزیر**

def is\_safe(board, row, col):

    for i in range(col):

        if board[row][i] == 1:

            return False

    for i, j in zip(range(row, -1, -1), range(col, -1, -1)):

        if board[i][j] == 1:

            return False

    for i, j in zip(range(row, len(board), 1), range(col, -1, -1)):

        if board[i][j] == 1:

            return False

    return True

def solve\_n\_queens(board, col):

    if col >= len(board):def is\_safe(board, row, col):

    for i in range(col):

        if board[row][i] == 1:

            return False

    for i, j in zip(range(row, -1, -1), range(col, -1, -1)):

        if board[i][j] == 1:

            return False

    for i, j in zip(range(row, len(board), 1), range(col, -1, -1)):

        if board[i][j] == 1:

            return False

    return True

def solve\_n\_queens(board, col):

    if col >= len(board):

        return True

    for i in range(len(board)):

        if is\_safe(board, i, col):

            board[i][col] = 1

            if solve\_n\_queens(board, col + 1):

                return True

            board[i][col] = 0

    return False

def print\_solution(board):

    for row in board:

        print(row)

n = 8

board = [[0 for \_ in range(n)] for \_ in range(n)]

if solve\_n\_queens(board, 0):

    print\_solution(board)

else:

    print("No solution exists.")

        return True

    for i in range(len(board)):

        if is\_safe(board, i, col):

            board[i][col] = 1

            if solve\_n\_queens(board, col + 1):

                return True

            board[i][col] = 0

    return False

def print\_solution(board):

    for row in board:

        print(row)

n = 8

board = [[0 for \_ in range(n)] for \_ in range(n)]

if solve\_n\_queens(board, 0):

    print\_solution(board)

else:

    print("No solution exists.")

**مواجهه با مسائل غیر منطقی در هوش مصموعی:**

1. استفاده از شبیه سازی : استفاده از شبیه سازها و مدل سازی های پیچیده می تواند به شما کمک کند تا به تحلیل و حل مسائل غیرمنطقی بپردازید.با ایجاد محیط های شبیه سازی مناسب ، میتوان الگوریتم های خود را آزمایش کرده و به بهبود آن ها پرداخت.

2. استفاده از روش های تقویتی: مانند یادگیری تقویتی و یادگیری تقویتی عمیق می تواند به شما کمک کند تا با محیط های پویا و پیچیده سروکار داشته باشید و راه حل های منطقی و بهینه برای مسائل غیرمنطقی پیدا کنید.

3. استفاده از روش های پردازش زبان طبیعی : ر برخی موارد استفاده از این روش مانند تحلیل متن و استخراج اطلاعات ممکن است به شما کمک کند تا با داده های پیچیده و نامنطقی مواجه شده و به راه حل های مناسب برسید.

4.استفاده از روش های تصمیم گیری چند معیاره : در صورتی که با مسائل غیرمنطقی و پیچیده مواجه شدید استفاده از روش های تصمیم گیری چند معیاره مانند تحلیل سلسله مراتبی و تحلیل تصمیم چندشاخصه ممکن است به شما کمک کند تا تصمیمات منطقی و بهینه بگیرید.

زهرا ایزدی نسب

40016341054253