به نام خدا

مسیح تنورساز 40006133

تکلیف سوم هوش مصنوعی

سوال 1:

```
backtracking(board, col)
```

تابع اصلی برای حل این سوال با استفاده از روش عقبگرد تابع backtracking است. این تابع به صورت بازگشتی عمل میکند و در هربار فراخوانی جدول با یک ستون بیشتر را در نظر میگیرد. زمانی که به ستون نهایی برسیم متوقف میشود و باز میگردد.

```
canplace(board, row, col)
```

این تابع به ما میگوید آیا میتوانیم در سطر و ستون مشخص شده وزیری را قرار دهیم یا خیر. در صورتی که میتوانستیم وزیری قرار دهیم True و اگر نمیتوانستیم False باز میگرداند. چک کردن این خانه ها سه حالت دارد: وزیر ها در یک ردیف باشند،وزیر ها در قطر سمت چپی و بالا باشند،وزیر ها در قطر سمت چپی و پایین باشند. دلیل چک نکردن قطر سمت راست آن است که ما هربار با یک واحد افزایش برای ستون، backtracking را فراخوانی میکنیم. انگار که نیمه ی سمت راست را نداریم و در نظر نگرفته ایم.

با فراخوانی این تابع درون backtracking ما متوجه میشویم که وزیر را در سطر و ستون مشخص شده قرار دهیم یا خیر. اگر میتوانستیم، وزیر را قرار میدهیم و تابع backtracking را با یک واحد افزایش برای ستون فراخوانی میکنیم.

برای شروع باید تابع backtracking با ستون 0 فراخوانی کنیم، تا از ستون صفر شروع کند.

اگر زمانی به بن بست رسیدیم به شاخه قبلی باز میگردیم و شاخه جدیدی را امتحان میکنیم. این همان الگوریتم عقبگرد است. توجه شود که در این سوال از مکاشفه ای استفاده نشده است، در صورتی که از مکاشفه ای استفاده کنیم انتظار داریم الگوریتم بهینه تر عمل کند، مثلا تعداد بازگشت به عقب کمتری داشته باشد و یکسری از شاخه ها را اصلا وارد نشود.

سوال 2:

```
local_search()
```

تابع اصلی برای حل این سوال تابع local_search است. این تابع با فراخوانی get_neighbors لیست همسایه ها را میگیرد. یکی را به تصادف انتخاب میکند. چک میکند که آیا این همسایه تعداد conflict کمتری دارد یا خیر. اگر کمتر بود، به این همسایه تغییر حالت میدهد.

روند ذکر شده را به تعداد MAX_ITERATIONS انجام میدهد، اگر به جواب نهایی رسیدیم True و در غیر این صورت False باز میگرداند.

```
get_neighbors(board)
```

این تابع جدول الان ما را دریافت میکند و با جابجا کردن وزیر ها 72 حالت مختلف را تولید میکند. در نهایت لیست این 72 همسایه را برای ما بازمیگرداند.

```
count_conflicts(board)
```

حالت حال حاضر را دریافت کرده و تعداد تهدید های موجود در زمین بازی را بازمیگرداند.

```
create_initial_board()
```

جدول اولیه را تعریف میکند. به صورت تصادفی یک وزیر در هر ستون میچیند.

طبق فرض، در هر ستون حداکثر یک وزیر داریم و لازم نیست نگران قرار گرفتن چند وزیر در یک ستون باشیم.

```
print_board(board)
```

زمین بازی را برای ما چاپ میکند. برای وزیر ها Q را پرینت میکند.