

Literals 與 Clauses:

實際上的 literal 是帶正負值(positive/negative)得，而 clauses 是好幾個 literals 組合而成的。如果我自訂一兩個 Class 來表示這兩個概念，會因為要自己補__eq__和__hash__而變得很麻煩，所以我直接用整數來表達，正數表示 positive literal，負數表示 negative literal，而 clauses 就是一個 set of literals。例如：{1, 2, 3}表示一個 clause，其中包含三個 literals，分別是 1, 2, 3。而因為 set 是 python 內建的資料結構，所以用起來會比自定義的 class 方便很多。而在這個作業裡面，我把 KB 定義為一個 list of clauses，例如：[{1, 2, 3}, {1, 2, 4}]，KB0 則因為都是 single literal，所以就是單純整數的 set，例如：{1, -2, -3, 4}，表示 1, 4 是 Mine，2, 3 是 Safe。

執行流程:

game_start(): 一直跑 play()這個 function，並在跑不下去(stuck)或是過關的時候幫忙確認玩家的答案是否正確。

generate_mine(): 產生地雷的位置，回傳一個 list of tuple，每個 tuple 是一個地雷的位置。

generate_board(): 根據是 player 還是 answer，產生對應的空白地圖。

assign_answer_board(): 根據地雷的位置，把地雷的位置填入地圖，並且計算每個格子周圍的地雷數量。

get_init_safe(): 隨機產生一個 safe 的位置，並且回傳。

print_board(): 印出 player 或是 answer 的地圖。

dig(): 根據玩家輸入的位置，挖掉該位置，如果挖到地雷就回傳-1，否則回傳該位置周圍的地雷數量(可以從 answer_board 得到)。

clean_KBs(): 把 KB 裡面的所有 clauses 都掃描一遍，如果裡面已經有確認過的 literals，就把該 clauses 刪掉，代表這個 clauses 已經沒有用了(已經恆成立了)，然後取 unique 和根據長度排序，這個 function 只是為了確保 KB 的資料是最精簡的狀態。

KB_to_KB0(): id 是傳入值去掉正負號，mine_or_safe = tar > 0，如果 mine_or_safe 是 True，代表是 Mine，否則是 Safe。因為不能一邊迭代一邊刪

除，所以先把要刪除的 index 記下來，最後再一次刪除。而如果 id 一樣，要看是不是同號，如果同號就把整個 clauses 從 KB 刪除，如果不同號就把該 literal 從 clauses 刪除。再來確認 KB 內有沒有 clause 是其他 clauses 的 subset，如果有就把 super set 刪除，直到沒有新的 subset 出現為止。

add_to_KB(): 如果 KB 裡面已經有這個 clauses 了，就不用加入了，如果沒有而且長度等於 1，就把這個 clauses 直接加入 KB。剩下的把負面敘述出現在 KB0 裡面的 literal 刪掉，同時做 subsumption 直到沒有新的 subset 出現為止。然後再把自己加到 KB 裡面。

play(): 先執行一次 clean_KBs()，確保 KB 和 KB0 的資料都是最精簡的狀態，先找一個在 KB 裡面的 single literal 當成目標，如果找不到就回傳-1，然後把這個目標透過 KB_to_KB0()加入 KB0，如果目標是 Mine，就把他 mark 起來，如果是 Safe，就 dig 它，如果回傳-1，代表挖到地雷，就回傳-1，否則看周圍那些座標(around)，如果 dig 的回傳是 0，所有周圍的地都是安全的，就直接把 around 裡面的座標做成 single negative literals 加到 KB 中，如果周圍全部都是地雷，則把 around 裡面的座標做成 single positive literals 加到 KB 中，如果周圍有地雷也有安全的，就利用排列組合的工具，把周圍的座標做成 all positive/negative clauses 加到 KB 中，然後再執行一次 clean_KBs()，確保 KB 和 KB0 的資料都是最精簡的狀態，然後就 return 回去，繼續下一輪的 play()，直到 KB 裡面沒有 clauses 了。

check_win(): 檢查玩家的地圖和答案是否一樣，如果除了地雷的位置的標誌不同以外，其他都一樣，就代表過關了，回傳 True，否則回傳 False。

一些實驗與發現:

我自己在測試的時候，發現 Hard 的場地其實基本上很難過，因為地雷的數量太多了，基本上快要是 Easy 的兩倍，這樣地雷的比例之下就算是給兩倍的多比例的 init_safe 也很難過關。還有不管在甚麼難度之下，只要有出現一整排的數字，會因為可行種類太多而卡住，然後就過不了，也可能有地雷直接圍成一圈，直接導致無法判斷，這些都是我在測試的時候發現的問題。下面幾張圖是一些範例:

