**과제 1**

2018062733 윤동빈

1. 함수 설명

def check(col, c, r)

: 현재 c번 열의 퀸을 r번 행에 놓으려 할 때, c번 이전의 열에 놓인 퀸들이 현재 퀸을 공격하는지 확인하는 함수로 bfs()에서 현재 퀸의 위치를 결정할 때 사용

col = 각 열에 있는 퀸들의 행 정보를 담고 있는 list  
c = 현재 퀸의 열  
r = 현재 퀸의 행

def getH(col, n, c, mode)

: c번 열 퀸이 이외 퀸들에게 공격받는 횟수를 반환하는 함수로 mode가 True이면 c번 이전 열에 위치한 퀸들에게 공격받는 횟수를, 그렇지 않으면 c번 이후 열에 위치한 퀸들에게 공격받는 횟수를 반환

hc()에서 h[r][c]를 r번 행 c번 열에 퀸이 위치했을 때 서로 공격할 수 있는 pairs of queens 개수로 설정했는데, 이 h[r][c]를 업데이트하기 위해 getH()가 쓰임

col = 각 열에 있는 퀸들의 행 정보를 담고 있는 list  
n = input으로 주어지는 n  
c = 현재 퀸의 열  
mode = 탐색 범위 결정 (bool)

def outOfBound(r, c, n)

: 주어진 사이즈가 n인 n by n 행렬일 때, (r, c)가 행렬을 벗어나는지 확인하는 함수

r = 행  
c = 열  
n = input으로 주어지는 n

def updateRemainCnt(selected, remainCnt, reducedCnt, r, c, n)

: csp()에서 쓰이는 remainCnt가 열 별로 퀸이 위치할 수 있는 장소 개수를 의미하는 list인데, 이 remainCnt를 업데이트하는 함수

Selected = 퀸의 자리가 선택됐는지 알려주는 list (True/False)  
remainCnt = 퀸이 위치할 수 있는 장소 개수를 의미하는 list  
reducedCnt = 탐색 실패 시 remainCnt 복구용

2. 알고리즘 설명

(1) def bfs(n)

,   
return 타입은 list : 정답을 찾는데 성공했다면 각 퀸의 row 정보를 담은 list를 반환하고, 그렇지 않으면 -1이 n개 담긴 list를 반환



  
큐에는 현재 열 번호(columnNumber)와 현재 각 퀸의 정보(states)를 담은 리스트를 원소로 가짐

  
columnNumber를 0부터 시작해 다음 열로 순차적으로 탐색하기 때문에 n에 도달한다면 종료  
종료 시 답을 찾은 상태이므로 각 퀸의 row 정보가 담긴 states를 리턴

텍스트이(가) 표시된 사진

자동 생성된 설명  
현재 열(columnNumber)이 정해진 상태에서 퀸이 최종적으로 위치할 수 있는 행(r)을 탐색  
check()를 통해 이전 열에 위치한 퀸들과 공격받는 위치인지 확인  
공격받지 않는다면 퀸을 놓을 수 있으므로 해당 위치에 퀸을 놓고 큐 삽입 후 탐색 진행  
(nextStates는 다음 상태의 각 퀸의 row 정보를 의미)

(2) def hc(col, h, n, curH)

텍스트이(가) 표시된 사진

자동 생성된 설명, 텍스트이(가) 표시된 사진

자동 생성된 설명  
return 타입은 bool로, 답을 찾았는지 여부에 따라 True or False 반환

텍스트이(가) 표시된 사진

자동 생성된 설명  
h는 list로,  
h[r][c]는 r번 행 c번 열에 퀸이 위치했을 때 서로 공격할 수 있는 pairs of queens 개수를 의미함  
curH는 현재 상태에서 마지막으로 이동한 퀸의 h[r][c] 값을 의미함

Hill-Climbing Search는 greedy algorithm이므로 현재 h값보다 작은 h값 중 최솟값인 위치로 이동하는데 그 위치의 좌표(r, c)와 최솟값(minn)을 얻는 과정

텍스트이(가) 표시된 사진

자동 생성된 설명  
바로 전 과정에서 더 작은 h[r][c] 값을 얻지 못했다면 stuck에 걸린 상태이므로 탐색 실패(False)

그렇지 않으면 탐색을 진행 (min != 987654321)  
현재 퀸을 바로 전 과정에서 얻은 좌표로 재설정 후 (col[c] = r)  
다음 탐색에서 쓰일 h를 getH()를 통해 업데이트 하는 과정  
hc()는 재귀를 통해 탐색을 진행 ( )

(3) def csp(col, selected, remainCnt, n, time)

텍스트이(가) 표시된 사진

자동 생성된 설명,   
return 타입은 bool로, 답을 찾았는지 여부에 따라 True or False 반환  
csp()는 재귀로 동작하면서 다음 단계로 넘어갈 때마다 time이 1씩 증가함  
즉, 서로 다른 퀸을 n개 설정하는데 성공했다면(time >= n) 정답 조건을 만족했으므로 True 리턴  
  
텍스트이(가) 표시된 사진

자동 생성된 설명  
next는 [선택되지 않은 퀸의 열, 해당 열에서 위치할 수 있는 장소 개수]를 원소로 하는 list  
위 코드에서 첫 번째 for loop는 next를 생성하는 과정  
  
**Most constrained variable**을 선택하기 위해 next를 legal value 개수가 적은 순으로 정렬  
  
두 번째 for loop는 **Least constraining value**를 선택하는 과정으로 위의 next를 활용  
퀸이 위치할 수 있는 행이 0~n-1이 있을 때, 중간 행에서 멀수록 다른 variable의 경우의 수에 영향을 덜 주게 되므로 양끝 행에서 하나씩 인덱스를 가져와 범위를 중간으로 좁혀가는 방향으로 진행  
(r1은 0에서, r2는 n-1에서 시작하므로 r1>r2일 때 탐색 종료)

텍스트이(가) 표시된 사진

자동 생성된 설명  
selectedCol은 이번에 고려할 퀸의 열을 의미 (위 next의 첫 번째 원소)  
reducedCnt는 csp()가 재귀를 통해 탐색을 진행하고 인자로 remainCnt가 계속 넘어가기 때문에 탐색 실패 시 원래 정보로 복구하기 위한 용도

이미 자리가 결정된 퀸들에 대해 현재 열(selectedCol)에서 퀸이 r1번 행에 놓일 수 있는지  
즉, 서로 공격하는 위치에 있는지 조건 확인 (2번째 if문)

만약 공격받는 위치이면 탐색을 진행할 수 없으므로 r1행에서 탐색 중단  
여기서 flag는 탐색 진행 가능 여부를 나타내며 True or False 값을 가짐

텍스트이(가) 표시된 사진

자동 생성된 설명  
탐색이 가능할 때,  
updateRemainCnt()를 통해 remainCnt 업데이트 및 퀸 정보 등록(col, selected)

재귀를 통해 탐색을 진행하는데 True가 반환됐다면 정답을 얻었다는 말이므로 True 리턴  
그렇지 않으면 변경된 정보(selected, remainCnt) 복구

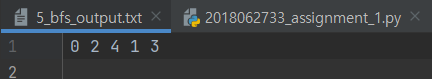
텍스트이(가) 표시된 사진

자동 생성된 설명  
r2에서도 위와 같은 방법으로 진행

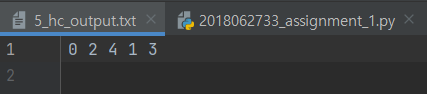
최종적으로 탐색 실패 시 False를 반환

3. 실험결과 (N=5)

(1) BFS 결과



(2) Hill-Climbing Search 결과



(3) CSP 결과

