**ITBT탈출** 18-2 인공지능 과제 1

2013011082 이상옥

**1.1. 코드 설명 – 전체 구조**

각 층에 대해 BFS, (개선된)DFS, (개선된)Greedy Best Search, A\* 중 가장 적절한 알고리즘을 사용.

코드구현은 아래와 같은 형식으로 되어 있음.



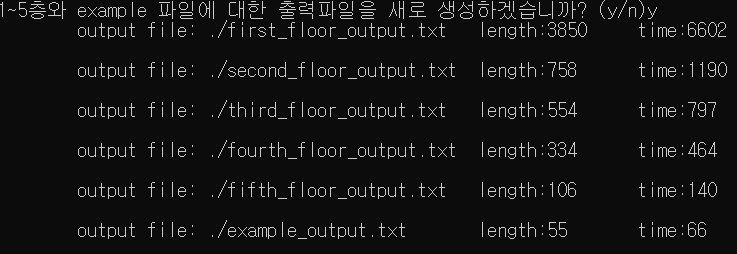
탐색 비용 count 는 CQ. count를 이용해 측정함. 상세 내용은 코드 내 주석 참조.

**1.2. 코드 설명 – 채점 시 참고사항**

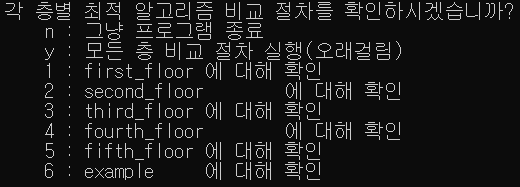
first\_floor(), second\_floor(), third\_floor(), fourth\_floor(), fifth\_floor()는 **파라미터 없음**.

각 함수는 과제 요구조건대로 5개 함수를 실행 시 해당되는 출력파일을 생성함

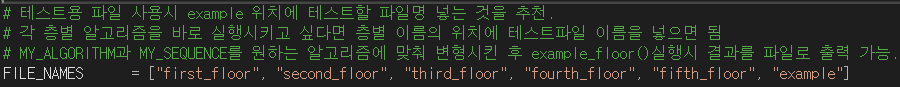
파일을 실행시 자동으로 실행되는 FindBestAnswers()에 의해 아래 창이 뜬다.



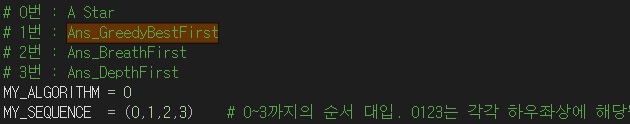
각 층별로 알고리즘간 비교하고 싶으면 그 다음 선택지에서 해당되는 층을 누르면 됨



다음은 주요 global 변수임 (파일 최상단에 있음)



**n번째 층에 적용한 알고리즘으로 새로운 미로를 확인하고 싶다면** File Names에서 해당되는 층의 이름을 새로운 미로의 이름으로 변경한 후, 해당 층을 실행하면 됨.



example의 위치에 새로운 미로의 이름을 넣으면 기본적으로 A\*알고리즘에 하우좌상 우선순위로 실행함. 이를 바꿔서 원하는 알고리즘으로 결과 확인 가능

**2.1. 층별 사용 알고리즘**

first\_floor AStar [0, 1, 2, 3] len:3850 time6602

second\_floor DepthFirst [2, 3, 0, 1] len: 758 time: 1190

third\_floor AStar [0, 1, 2, 3] len: 554 time: 797

fourth\_floor DepthFirst [1, 2, 3, 0] len: 334 time: 464

fifth\_floor AStar [0, 1, 2, 3] len: 106 time: 140

0123은 각각 하우좌상에 해당됨.

**단, Greedy 알고리즘은 본 과제 내에서 약간의 개량이 이뤄진 상태**임. (DepthFirst, Greedy Besty Search)

Greedy Algorithm이 Optimal 한 답을 찾으려면 모든 경우의 수를 확인해 봐야 하는데, 어차피 current best answer 보다 더 비싼 비용의 optimal answer가 나올 리가 없으므로

current best answer 보다 비용이 비싸지는 경우 자동으로 차단해버림.

**2.1. 각 층별 알고리즘 선정 이유**

A\*, Ans\_GreedyBestFirst, Ans\_BreathFirst, Ans\_DepthFirst 에 대해

**상하좌우 조합 4!가지 경우의 수를 모두 비교해 가장 저렴한 방법을 선정**함.

이는 모든 층 알고리즘 비교 선택지를 고르면 시각적으로 확인가능.