인공지능 assignment1

2013012041 안효민

IT/BT 탈출하기

➔ 코드 동작

모든 층에서 동일한 알고리즘, 함수를 사용했기 때문에 각 층별로 실행절차는 같습니다. 먼저 파일을 열어서 2차원 리스트에 저장한 후에 딕셔너리와 리스트의 행,열값을 이용한 해시값으로 각각의 노드와 해당 노드의 상하좌우에 있는 노드와의 관계를 표현했습니다. 그리고 시작지점, 도착지점과 열쇠가 있는 노드를 알아내 따로 변수에 저장 후에 그리디 알고리즘을 구현한 함수 greedy에서 도착지점까지 최단 경로를 계산했습니다. 그 후에 모두 1로 구성된 2차원 리스트를 생성해서 지나간 경로와 시작지점, 도착지점을 각각 5,3 그리고 4로 바꾸고 결과파일을 생성했습니다.

➔ 사용 알고리즘

모든 층에서 그리디 알고리즘을 사용했습니다. BFS로 계산한 최단경로와 비교 결과 그리디 알고리즘을 사용해도 모든 층에서 최단경로를 도출해냈고 해당 층들의 정보를 이미 알고 있어 informed search algorithm을 사용하는 것이 합리적이라고 생각했습니다.

그리디 알고리즘은 heapq를 이용해서 구현했으며 heuristic 함수는 노드와 목표지점 또는 열쇠와의 행, 열의 차이의 제곱의 합을 사용했습니다

➔ 함수별 부가설명

- def output

~_floor_output.txt를 작성하기 위한 함수입니다. 크기와 최단경로 해시값의 리스트, 시작지점, 목표지점, 그리고 각 층의 함수에서 생성한 파일의 descriptor와 해당층의 최단경로 길이와 시간을 인자로 받습니다.

- def findNeighbor

입력된 노드의 상하좌우 이웃한 노드가 벽이 아닌지 검사한 후에 이웃노드로 설정하는 함수입니다. 파일에서 불러온 데이터와 노드의 좌표에 해당하는 리스트 인덱스, 그리고 미로의 가로 또는 세로길이와 노드간 관계를 표현한 딕셔너리를 인자로 받습니다.

- def bfs

최단경로가 맞는지 확인하기 위한 bfs알고리즘이 적용된 함수입니다.

- def greedy.

그리디 알고리즘이 적용된 함수입니다. 시작지점과 도착지점, 노드간 관계를 표현한 딕셔너리, 그리고 열쇠의 해시를 인자로 받습니다.

➔ 실험 결과