13주차 예비보고서

전공: 경제학과 학년: 4학년 학번: 20180501 이름: 김연수

1.

Dfs 시간 복잡도는 미로의 크기가 n\*m이라고 가정했을 때 최악의 경우 O(n \* m)의 시간 복잡도를 가진다. 최악의 경우 미로를 전부 탐색하기 때문이다. Dfs는 깊이 우선 탐색이다 따라서 stack 자료구조를 이용해 구현할 수 있다. Stack은 특성상 마지막으로 들어온 원소가 첫 번째로 나가게 되는데, 이를 LIFO라고 부른다. 따라서 stack을 이용해 구현하는 경우, 막다른 길에 도달할 때까지 한 길로만 탐색하고, 막다른 길에 도달하게 되면, 다시 전으로 돌아가서 길을 탐색한다. 미로의 종착점이 dfs의 탐색 끝자락에 발견될 가능성이 존재하는 데, 이 때 시간 복잡도는 미로의 크기와 같다.

Bfs의 시간 복잡도 또한 최악의 경우 위 Dfs와 시간 복잡도가 같다. 다만 bfs의 경우 넓이 우선 탐색이기에, 최단거리 경로를 탐색하는 데에 유리하다. 최단 거리를 찾는 경우가 아니라면, dfs와 비교해서 크게 유리한 전략은 아니다.

그래프의 경우 아래와 같이 표현할 수 있다.

DFS: O(V + E)

BFS: O(V + E)

여기서 V는 정점의 수, E는 간선의 수를 의미한다.

2.

자료구조를 다음과 같이 구현한다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

x, y는 노드의 그래프 상의 좌표를 의미한다. A, b는 경로를 그리기 위한 화면상의 좌표를 의미한다. Value는 각 노드의 식별 변호이다. Left, up, right, down은 각각 인접한 노드를 가리킨다. 연결되어 있는 않다면 nullptr를 가리키다. Next와 prev는 dfs 또는 bfs로 찾은 미로의 경로 상에서 다음 경로와 전 경로를 의미한다. head노드부터 시작해 next노드를 따라가면 미로의 종착지의 도달할 수 있도록 설계한다.

dfs알고리즘을 iterative한 방법으로 구현하기 위해서 stack자료구조를 사용한다. cpp에서 지원하는 stack 자료구조와 관련 함수를 사용하여 구현할 수 있다. While loop를 돌아 stack에서 해당 원소가 존재하는 지 검사하고, 존재한다면, 원소를 빼내서 원소의 위치가 미로의 종착점인지를 확인한다. 만약 종착점에 도달했다면 while loop을 빠져나온다. 아니라면 left, up, right, down 중 이동 가능한 경로를 탐색한다. 이동 가능한 경로라면, visit표시를 해준뒤 stack에 해당 노드를 push해준다. 이와 같은 알고리즘으로 구현하면 재귀적으로 구현하지 않아도 dfs알고리즘을 설계할 수 있다.