자료구조 미로찾기

201411802 응용통계학과 전민규

우선 처음엔 이것을 자료구조로 어떻게 활용할지 보다 미로를 실제 탐색한다면 어떤식으로 할 지 고민해 보았습니다. 그렇게해서 몇가지 생각해 보았고 이것을 바탕으로 자료구조에 접목할 수 있을지 생각해 보았습니다.

1. 전체적인 미로의 지도를 볼 수 있을 경우

1) 우선 3면이 막혀있는 곳을 모두 막습니다. 3면이 벽으로 막혀있다면 그곳은 더 이상 어디로 갈 수 없는 막다른 곳이기 때문입니다.

2) 그 후 각 막혀있는 모서리에서 갈래길이 나올때까지(즉 한길만 따라 오다가 갈라지는 길이 있을 때 까지) 그 길을 따라가며 모두 지웁니다.





3) 그렇게 되면 단 하나의 길, 즉 출발 지점에서 도착지점까지 이르는 하나의 길만 남게되며 그림에서 보면 파란색 길입니다. (미로 그림은 인터넷에서 가져왔고 빨간, 파란 표시는 직접 그렸습니다.)

-> 이를 구현한다면 전체적 지도를 보았을 때 삼면이 벽으로 둘러쌓인 곳의 좌표를 배열에 넣고 그 위치에서 시작했을 때 단 하나의 길이 있을 테니 그 길을 따라가며 길을 지우면 됩니다.

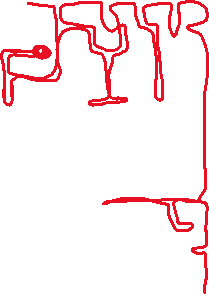
즉 벽을 1, 길을 0이라 했을 때 이미 지난길들을 2로 채워 더 이상 갈 수 없는 길로 만들면 이것이 지우는 방법이 될 것입니다.

이렇게 배열에 있는 모든 수들로부터 길을 지우고 남은 길만 길로 설정하면 됩니다. 이는 그저 배열정도를 필요로 합니다. 또한 보통의 경우 미로의 지도가 주어지지 않고 시행착오를 겪으며 찾아가는 것을 가정하기에 이 방법은 그렇게 많이 쓰일 것 같지는 않습니다.

2. 벽집고 가기

미로의 입구부터 출구까지 한 벽을 손으로 집고 출구가 나올 때 까지 쭉 가다보면 언젠가는 출구가 나온다는 말입니다. 막다른 길이 나와도 무시하고 벽을 집고 따라가다 보면 밑의 그림과 같이 결국 출구에 도달합니다.





이를 구현하기 위해 ‘한쪽면’에 대해 고려할 때 우선 아래 그림은 ‘왼쪽면’을 기준으로 하였습니다. 한방향으로 쭉 가다가 벽을 만나거나 길이 끊어지면 길이 있는 방향으로 꺾으면서 기준도 그에따라 바꿔줍니다.

이 때 원래 ‘왼쪽면’을 기준으로 잡고 가기로 했으므로 꺾을때는 꺾었을 때 왼쪽에 벽이 있는 방향으로 꺾습니다. 또한 길이 끊어진 경우(파란색 동그라미 부분) 항상 왼쪽(기준)으로 돌면 됩니다. 이 방법 역시 되돌아 올 필요 없이 앞만보고 쭉 가면 되므로 스택의 개념도 필요없고 그저 왼쪽, 오른쪽 중 어느벽을 타고 갈건지와(어느 벽이든 일정하기만 하다면 상관은 없습니다) 현재 ‘앞’이 뭔지를 식별할 수 있는 변수 정도면 충분할 것입니다.

3. 시행착오-1

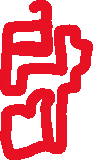
다음은 아무런 정보없이 하나하나 시행착오를 겪으며 가는 방법입니다.

1) 하나의 경로를 임의로 선택하여 가보고, 아래 그림처럼 막히면 다시 돌아와 다른 경로로 시도하는 방법입니다.

2) 실패하여 이전 것으로 다시 돌아갔을 때 가능한 경로들을 고려하기 위해 그 경로들이 어딘가에 저장되어 있어야합니다.

3) 이전 것으로 돌아갈 때는 가장 최근의 것으로 가야 하므로 이에 가장 적합한 자료구조로 stack이 있습니다.





다음은 stack으로 구현하기 위한 몇가지 과정들입니다

1) 우선 현재 위치에서 갈 수 있는 칸들의 좌표를 구조체로 정의하고 그 좌표를 스택에 넣습니다. 이 때 갈림길 없이 한 방향으로만 간다면 스택에 넣지 않습니다.

2) 지나갈 때마다 지나온 것들은 다른 숫자로 채워넣어 이미 들린 길을 다시 가지 않도록 합니다. (즉 한번 지나간 칸을 다시 가면 안됩니다)

3) 막다른 길을 만나면 가장 가깝고 아직 가보지 않은 칸, 즉 스택의 맨 위에 있는 칸으로 돌아가서 그 위치를 현재 위치로 재설정 한 후 새로운 경로를 찾습니다. 되돌아 갈때는 pop으로 넣었던 것들을 빼주면 됩니다.

4) 이러한 과정을 현재의 위치가 출구와 같거나 모든 위치를 다 검사할때까지 반복합니다.

조금 더 구조적으로 말하면

- 2차원 배열로 map을 만들고 배열값이 0이면 갈 수 있고 1이면 벽인 것으로 정함

- 출구는 E라 표현, 현재 위치는 X라 표현

- 이미 지나간 자리는 2로 채워넣음

- 스택에 저장되는 데이터는 행,열 좌표이며 이는 구조체로 처리함

-필요한 함수로는 우선 stack의 기본적 함수들인 init, push, pop 등에 push함수는 벽(1)이 아니거나 이미 지나온(2)가 아닐 경우 갈래길이 있다면 push하는 기능을 합니다.

4. 시행착오-2 (모든 경로 탐색)

다음은 또다른 시행착오중 하나인 길을 따라 꺾지않고 쭉가면서 확인하고 출구가 없을 경우 다시 그대로 돌아와서 다른 루트로 탐색하는 방식입니다.

갈림길이 나타날 때마다 다른 길이 있다는 정보만 기록하면서 지나간 길은 위의 방법처럼 지워나갑니다. 그렇게 쭉 가다 막다른 곳에 도달하면 직전 갈림길까지 돌아가며 방금 갔던 길은 아니라는 표시를 남깁니다. 이런식으로 모든 갈림길을 순차적으로 탐색해 가는 방법이 있을 수 있습니다.