**Connecting.c에 대한 instruction**

2017314875조규원

1.main 함수

1. Component의 x, y 좌표를 저장할 block 타입 배열을 선언, read와 write에 fopen

2. n, m 값을 입력받고 nxm만큼 for문을 돌려서 주어진 map을 입력받음

3 fscanf로 임시변수 char temp에 %c로 문자를 받은 후, temp가 공백 문자나 개행문자일경우 무시하고 temp가 \*인 경우 map을 저장하는 maze[i][j] 에 -1을 저장

4. temp가 0이거나 어떤 숫자일 경우 temp-48을 maze[i][j] 에 저장

5. 탐색 알고리즘인 search를 호출

6. 찾은 경로를 주어진 포맷에 맞게 출력

2. search 함수

1. 현재 x, y좌표, 진행방향인 int vec을 인자로 받음

2. 주어진 조건을 만족(flag=1)일 때 return 1;

3. 지나온 component의 개수(num)가 총 component의 개수(k)와 일치할 때 flag=1로 설정하고 return 1;

4. 만약 chk[x][y]가 -1 또는 -2인 경우 이미 지나온 경로거나 \*인 블록임으로 지나갈 수 없어서 return 0;

5. 지나갈 수 있는 경로라면 chk[x][y]을 -1로 변경하고 만일 그 자리에 0이 아닌 다른 수가 있다면 component를 지난 것임으로 num++와 root\_num[index\_root]에 그 component의 값을 저장한다. 이때 num=k라면 return 1한다.

6. 기본적으로 진행방향을 유지하는 것으로 한다. 즉 아래쪽으로 진행중이었다면 아래 블록을, 그 후 좌우 블록을, 마지막으로 위 블록을 탐색한다. 다만 현재 블록의 인접한 블록의 값을 확인해서 그곳에 component가 있다면 그곳으로 이동한다.

7. count 값에 up, down, left, right 함수들을 호출하고 return 된 값을 더해준다.

8. 만약 count가 0이라면 모든 탐색이 실패한 것임으로 옳지 않은 경로임으로 현재 블록을 경로 스택에서 제거한다.

3. 이동 함수(up, down, left, right)

1. 임시 변수를 선언하고 index\_root를 1 증가시킨다.

2. 경로 스택에 움직이고자 하는 방향의 블록의 좌표를 저장하고, 그 블록의 좌표를 인자로 search 함수를 호출하고, 그 return 값을 임시변수에 저장한다.

3. 임시변수의 값을 return한다.

performance analysis

크기의 map이 주어졌을 때 최선의 경우 두개의 component가 주어지고 이들이 인접한 경우이다. 이때 이 알고리즘은 의 시간 복잡도를 가진다. 반면 최악의 경우 모든 경로를 탐색하는 경우인데 이 경우 현재 블록에서 선택 가능한 방법은 4가지이고, 이것이 번 반복됨으로 이다.