**2022 Spring OOP Assignment Report**

과제 번호 : prob1\_3

학번 : 20210273

이름 : 하태혁

Povis ID : hth021002

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

I completed this programming task without the improper help of others.

프로그램을 하다 보면 결정해야 할 세부 사항이 많은데, 이러한 세부 사항을 처리한 방법과 이유를 보고서에 쓰십시오.

독창적인 아이디어와 추가 기능은 보너스 점수를 받을 수 있으므로, 보고서에 명확히 기재하십시오.

문제가 여러 개인 경우, 각 문제별로 정리해서 작성합니다.

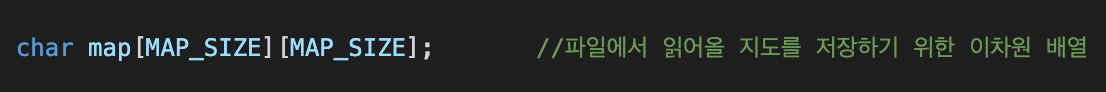
각 문항별 설명은 편의를 위한 것으로, 삭제하고 제출한다.

1. **프로그램 개요**

이 프로그램은 포스텍 학생들이 좀비를 피해 열쇠를 찾고 출구를 통해 탈출하는 최단경로를 찾기 위한 알고리즘으로 최단 경로를 찾기위해 BFS 알고리즘을 구현하여 사용했다. 10 X 10 지도를 각각 상하좌우가 연결된 그래프로 생각했으며 ‘0’, ‘k’, ‘e’를 그래프에서 방문할 수 있는 node로 해석했고 ‘1’은 이동할 수 없는 node로 해석했다. BFS가 진행되는 동안 방문한 node의 인접 node를 저장하기 위해서 자료구조 Queue를 직접 구현해 사용했다.

Input.txt에 저장된 지도를 불러오기 위해 <fstream>에 내장되어 있는 ifstream을 사용했고 최단 경로를 output.txt에 출력하기 위해서는 ofstream을 사용했다.

1. **프로그램의 구조 및 알고리즘**

텍스트, 장치이(가) 표시된 사진

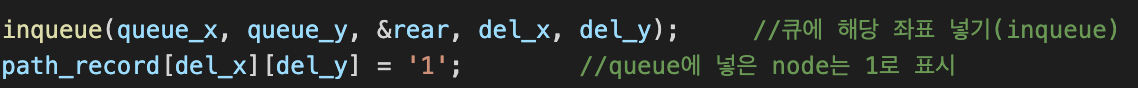
자동 생성된 설명

Char형 이차원 배열 map 을 선언하고 파일로부터 데이터를 읽어와 저장하였다. 이후 bfs 함수를 통해 시작점에서 key 까지의 최소거리와 key에서 exit까지의 최소거리를 받아와 최단거리를 계산했다.

텍스트이(가) 표시된 사진

자동 생성된 설명

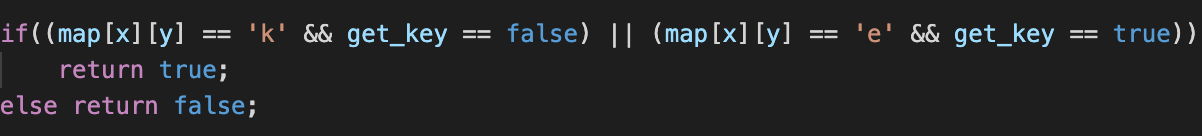
1. Bfs 내부에서 이차원 배열 두 개를 더 선언했다. Char형 이차원 배열 path\_record는 중복되는 node를 제거하기 위해 정의한 배열로 갈 수 없는 node는 ‘2’, 갈 수 있는 node는 ‘0’으로 저장했다. 이후 bfs를 진행하면서는 인접 node를 queue에 넣는 과정에서 이미 queue 존재하는 node가 중복되어 들어갈 수 있기 때문에, 이를 방지하고자 queue에 들어간 node는 1로 저장했다.



1. Int 형 이차원 배열 path\_record는 시작점과 해당 node 사이의 거리를 저장하기 위한 배열로 거리를 정수형으로 저장하기 위해 int형으로 선언했고 해당 node를 방문할 때마다 기존 node에 저장된 숫자에 1을 더했다.



1. get\_key는 열쇠의 획득 여부를 확인하기 위한 변수이다. bfs\_finish() 함수로 탐색 종료 여부를 확인할 때, 열쇠를 가지지 않은 상태로 ‘k’에 도달했을 때 종료, 열쇠를 가진 상태로 ‘e’에 도달했을 때 종료하기 위해 선언하였다. 만약 key를 가지지 않은 상태로 exit에 도달했을 때 프로그램이 종료되는 경우를 방지하기 위해 선언하였다.



Queue에 데이터를 삽입하고 삭제하는 과정은 사용자 정의 함수 inqueue와 dequeue로 구현하였으며 circle queue로 구현하였다. Front와 rear는 각각 queue의 가장 앞부분과 뒷부분의 위치를 저장하는 변수이다.

1. **토론 및 개선**

Test case를 더 만들고 프로그램을 실행해보면서 어떠한 예외 경우도 없이 프로그램이 실행될 수 있도록 여러 변수와 알고리즘을 추가하면서 BFS를 더 완벽하게 이해할 수 있었다.

키보드, 강판, 전자기기이(가) 표시된 사진

자동 생성된 설명 강판, 키보드이(가) 표시된 사진

자동 생성된 설명

위 사진과 같은 길이 넓은 map에서도 BFS알고리즘이 잘 작동되도록 프로그램을 제작하면서 node의 중복체크와 같은 개념도 이해하면서 알고리즘을 짤 수 있었다. 또한, 문제의 조건에 있었던 라이브러리를 쓸 수 있다면 훨씬 간편하고 편리하게 코드를 작성 할 수 있겠다는 생각을 해볼 수 있었다.

1. **참고 문헌**

<https://gmlwjd9405.github.io/2018/08/15/algorithm-bfs.html>