**2022 Spring OOP Assignment Report**

과제 번호 : 3

학번 : 20210661

이름 : 오승준

Povis ID : seungjunoh

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

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

1. **프로그램 개요**

이 프로그램은 미로를 입력받아, 시작점으로부터 키를 구하고 탈출구까지 걸리는 최단거리를 출력하는 프로그램이다.

프로그램을 실행시키기 위해서는 제출한 VS 프로젝트 파일(prob3\_20210661.sln)을 실행하여, f5 key를 누를 시 코드가 작동한다. 이때, prob3\_20210661 folder 안에 prob3\_20210661.cpp와 동일한 위치에 “input.txt” 파일이 있어야 정상적으로 프로그램이 실행된다.

프로그램 디렉토리 내부 파일들의 간단한 설명은 아래와 같다.

Prob3\_20210661.sln : 과제 3번 프로젝트 파일이다.

Prob3\_20210661.docx : 과제 3번 보고서 파일이다.

Prob3\_20210661 > prob3\_20210661.cpp : 과제 3번의 C++ 코드 파일이다. 이 파일과 같은 위치에 input.txt를 두어야 한다.

1. **프로그램의 구조 및 알고리즘**

이 프로그램을 제작하는데 있어, 경로탐색을 위해 BFS(너비 우선 탐색)를 활용하였다.

너비우선 탐색이란 시작점으로부터 가장 가까운 정점부터 시작하여, 가장 먼 정점까지 탐색하는 방식으로, 방문한 정점의 위치를 기억하기 위해 Queue를 이용한다.

프로그램의 진행과정은 다음과 같다.

1. 입력받을 데이터가 들어있는 파일(input.txt)을 열고, 결과값을 출력할 파일(output.txt)을 생성한다.

2. input.txt으로부터 미로를 읽어내 map1에 저장하고, 이를 이중 for문을 통해 map2에 동일하게 복사한다. 이 과정에, k, e의 위치에 대한 정보를 변수들(kx, ky, ex, ey)에 저장한다.

3. 우선 시작점으로부터 k까지의 최단거리를 구하기 위하여, map1[0][0]으로부터 시작하여, 더 이상 탐색할 곳이 없거나, k를 발견할 때까지 반복문을 통해 현재 칸에서 상하좌우 칸으로 이동가능한지 탐색하고, 이동가능한 경우 그 위치에 대한 정보를 x[cnt], y[cnt], l[cnt]에 저장한다. 이때, 이미 지나간 칸을 다시 이동하지 않도록, 탐색을 완료한 위치의 값을 1로 변경한다. K를 찾은 경우, l[pos-1]의 값이 최단거리의 값이므로 이를 stok에 저장한다.

l[pos-1]에 최단거리 값이 있는 이유는, 반복문을 탈출하는 과정에서, pos의 값이 1 올라간 이후 if문을 통해 k를 찾기 때문이다. l[pos]에는 null 값이 있다.

4. 3과 마찬가지 방식으로, k부터 e까지의 최단거리를 구한다. Map1은 탐색과정 중 값이 변형되었기 때문에, map2를 통해 확인한다. 이때, 탐색의 시작점은 map2[ky][kx]이다. 최단거리 값은 ktoe에 저장한다.

5. stok와 ktoe의 합을 output.txt에 출력한다.

6. input.txt, output.txt 파일을 닫고, 프로그램을 종료한다.

1. **토론 및 개선**

이번 과제를 통하여 탐색 알고리즘에 대해서 처음 접할 수 있게 되었다. 데이터를 탐색하는 과정에도 다양한 방법이 있음을 알 수 있었다.

이 프로그램을 조금 더 효율적으로 만들기 위해서는 BFS를 진행하는 과정을 사용자 정의 함수로 정리한다면 좋을 것으로 생각된다. 프로그램 내에서 BFS를 두번 진행하나, 이에대해 각각 코드를 작성하였다.

1. **참고 문헌**
   * + BFS를 이해하기 위해 아래 블로그의 내용을 읽었으며, 코드를 일부 인용하였습니다. <https://blog.hexabrain.net/269>