1. 목 적 : 미로 만들기

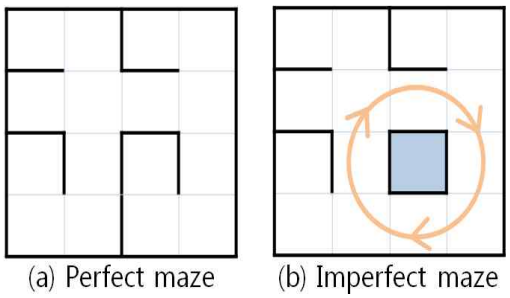
미로는 복잡한 길을 찾아 출발점부터 시작해 도착점까지 도달하는 퍼즐로 직사각형으로 배열된 방과 이러한 방들 사이를 막고 있는 벽으로 구성된다. 벽은 각 방의 상하좌우 사면에 존재할 수 있는데 인접한 두 방 사이에 벽이 없을 경우 두 방 사이를 자유로이 지날 수 있다. 본 프로젝트에서는 다음에 설명하는 완전미로를 만들 수 있는 여러 가지 방법을 알아보고 이를 직접 구현하기 위한 자료구조 설계 및 프로그래밍을 통해 미로를 생성해보도록 한다.

2. 프로그래밍 문제

완전 미로와 불완전 미로

완전미로(perfect maze)란 미로에서 임의로 서로 다른 출발점과 도착점을 설정할 경우, 두 지점을 연결하는 경로가 오로지 하나 존재하는 미로를 의미한다. 즉 완전미로에는 폐쇄된 공간이나 순환 경로가 존재하지 않는다. [그림1](a)에 나타난 미로는 폐쇄된 공간과 순환 경로를 모두 갖고 있지 않으므로 임의의 두 지점을 연결하는 경로가 오로지 하나 존재한다. 따라서 이 미로는 완전미로이다.

불완전미로(imperfect maze)란 완전미로가 아닌 미로이다. 즉 불완전 미로에는 폐쇄된 공간이나 순환경로가 존재하여 임의의 두 지점을 연결하는 경로가 하나 이상 존재한다. [그림1](b)의 3행3열의 방의 경우 4면이 벽으로 막혀있는 폐쇄 공간이고, 2행2열의 방부터 4행4열의 방까지 도달하는 경로가 2가지 이상 존재하는 순환경로이므로 이 미로는 불완전미로이다.



[그림 1] 불완전미로와 완전미로.

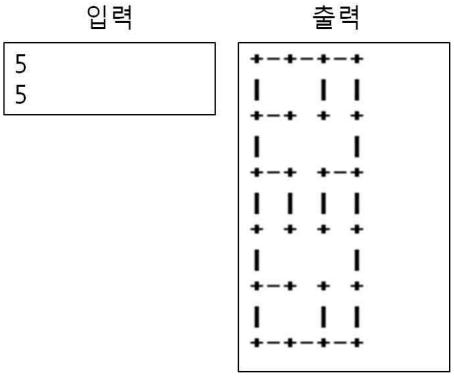
완전 미로 생성 문제

본 문제는 입력받은 너비와 높이를 갖고 예측 불가능한 완전미로를 출력하는 문제이다. 입력받는 너비는 미로가 가로로 가지고 있는 방의 개수이며, 세로는 미로가 세로로 가지고 있는 방의 개수이다. 따라서 [그림1](a)의 경우 가로, 세로 크기가 모두 4인 미로이다. 매 시행마다 예측 불가능한 미로를 만드는 문제이므로 입력하는 너비와 높이 값이 일정하더라도 출력되는 미로의 모양에는 차이가 있어야 한다.

입력형식: 입력은 표준입력을 이용한다. 입력 첫 줄에는 생성하려는 미로의 너비 N이 주어지고 이어지는 줄에는 미로의 높이 M이 주어진다.

출력형식: 입력된 너비와 높이를 갖는 $N \times M$ 미로를 text 파일로 출력한다. 출력되는 text 파일의 확장자는 '.maz'이다. 미로는 방과 벽으로 이루어져 있는데, 각 방의 모서리는 '+'(plus sign), 가로 벽은 '-'(minus sign), 세로 벽은 '|'(vertical bar) 그리고 방은 ' ' (space)로 나타낸다(아래의 예 참조).

입출력 예:



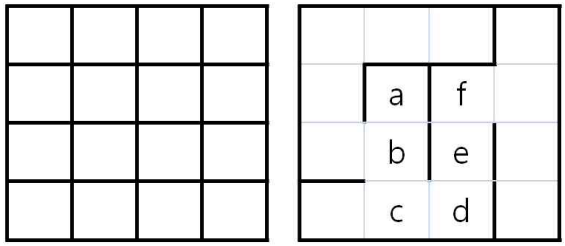
[그림 2] 입출력 예.

3. 실험방법

3-1 문제해결

미로 생성은 격자무늬로 배열된 방들에서부터 시작한다. $N \times M$ 미로는 N개의 열과 M개의 행으로 이루어진 NM개의 방을 갖고, 각각의 방들은 사면이 벽으로 막혀있다. 예를 들어 [그림 3]과 같이 4x4 미로를 만들기 위해 벽으로 둘러싸인 16개의 방들을 생성한다. 그 다음 인접한 방들 사이의 벽을 제거하여 두 방을 연결하는 통로를 만든다. 제거할 벽을 선택하는 과정이 본 문제의 관건인데, 이 과정에서 다음의 사항들을 모두 만족시켜야 한다.

1. 매 시행마다 예측 불가능한 미로를 만들기 위해서는 제거되는 벽들이 무작위로 선택되어야 한다.
2. 미로에서 임의의 두 지점을 연결하는 경로는 반드시 존재해야 하며 이 경로는 오로지 하나 존재해야 한다. 따라서 임의의 지점 X와 Y사이의 벽을 제거할지 결정할 때 두 지점을 연결하는 다른 경로가 이미 존재한다면 벽을 제거해서는 안 된다. 예를 들어 [그림 2]에서 지점 a와 f를 연결하는 경로가 이미 존재하므로 a와 f 사이의 벽을 제거해서는 안 된다.



[그림 3] 4x4 미로의 생성.

완전미로를 만들 수 있는 알고리즘에는 recursive backtracker, Kruskal's algorithm, Prim's algorithm, Eller's algorithm[1,2] 등이 존재한다. 이 중 Eller's algorithm은 매우 효율적인 알고리즘으로 무한한 크기의 미로를 linear time에 생성할 수 있다. 이 알고리즘에 대하여 살펴보면 아래와 같다.

< Eller's Algorithm >

Eller's algorithm 은 차례로 한 줄 씩 미로를 생성해나가는 방법이다. 한 번에 한 줄 씩 미로를 생성하므로 크기가 $N \times M$인 미로를 만들 경우 크기가 N인 1차 배열을 사용할 수 있다. 예를 들어 5×3 미로를 생성한다면 [그림 4]와 같이 크기가 5인 1차 배열을 사용한다. 이 알고리즘의 수행 순서는 다음과 같다.

① 미로의 첫 번째 줄을 초기화한다. 첫 번째 줄에 속한 N개의 방은 N개의 서로 다른 집합에 속하게 된다. 5×3 미로의 첫 번째 줄은 [그림 4]와 같이 나타낼 수 있다. 방에 적힌 숫자는 각각의 방이 속한 집합을 나타낸다. 5개의 방이 서로 다른 집합에 속해있으므로 각각의 방에 적힌 숫자도 모두 다르다.

1	2	3	4	5

[그림 4] 첫 번째 줄의 초기화.

② 미로의 현재 행에서 첫 번째 방부터 시작하여 마지막 방까지 서로 인접해 있는 방들이 어떤 집합에 속하는지 비교한다. 인접해 있는 두 방이 서로 다른 집합에 속해있다면 두 방 사이의 벽을 남겨둘지 제거할지 임의로 선택한다. 벽을 제거하면 인접한 방들을 연결하는 통로가 생기는데 이렇게 해서 연결된 방들은 모두 같은 집합으로 합쳐진다. 예를 들어 [그림 5](a)의 첫 번째 방과 두 번째 방은 각각 서로 다른 집합 1과 집합 2에 속하므로 두 방 사이의 벽을 제거하거나 남겨둘 수 있다. 임의로 두 방 사이의 벽을 제거하면 [그림 5](b)와 같이 첫 번째 방과 두 번째 방을 연결하는 통로가 생기고 두 방이 같은 집합 1에 속하게 된다. 이와 같은 방식으로 [그림 5](b),(c)를 거쳐 마지막 방까지 도달하면 [그림 5](d)와 같은 모양을 얻을 수 있다.

1	2	3	4	5

(a)

1	1	3	4	5

(b)

1	1	1	4	5

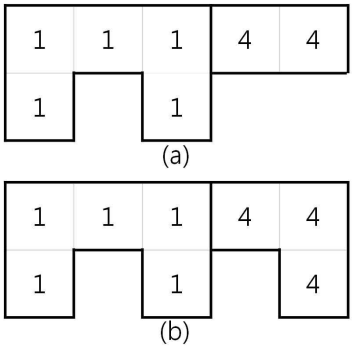
(c)

1	1	1	4	4

(d)

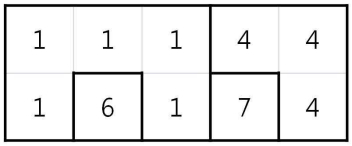
[그림 5] 첫 번째 줄의 벽 제거.

③ 현재 줄과 다음 줄 사이의 벽을 제거하고 두 방 사이의 수직 경로를 만든다. 현재 줄의 방 중 하단의 벽을 제거하고 다음 줄의 방과 연결될 방은 각 집합마다 하나 이상을 임의로 선택하여야 한다. 두 방 사이에 수직 경로가 만들어지면 이러한 두 방이 같은 집합에 속하게 되므로 첫 번째 줄의 방과 같은 집합에 속한 방들이 두 번째 줄에 나타나게 된다. 예를 들어 [그림 5](d)에서 집합 1에 속하는 첫 번째, 두 번째, 세 번째 방 중 첫 번째 방의 하단 벽을 제거하고 이 방과 인접해 있는 두 번째 줄의 첫 번째 방과 연결하면 [그림 6](a)와 같이 1행1열의 방과 2행2열의 방이 서로 연결되어 같은 집합 1에 속하게 된다. 이로써 집합 1은 수직 경로를 하나 생성하였으므로 나머지 두 방에 대해서는 수직 경로를 더 이상 생성하지 않아도 무방하다. 하지만 수직 경로를 생성하는 방은 무작위로 선택되므로 [그림 6](a)의 경우 1행1열의 방과 더불어 1행3열의 방 또한 하단의 벽이 제거되었다. [그림 6](a)에서는 집합 4에 속한 네 번째, 다섯 번째 방 모두 하단의 벽이 제거되지 않은 상태로 이렇게 되면 완전미로를 생성할 수 없다. 따라서 집합 4에 속한 네 번째, 다섯 번째 방 중 적어도 하나 이상을 선택하여 하단의 벽을 제거하고 두 번째 방과 연결한다. [그림6](b)는 다섯 번째 방의 하단 벽을 제거하고 인접한 1행5열의 방과 2행5열의 방을 연결한 상태이다.



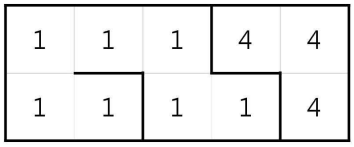
[그림 6] 두 줄 사이의 수직 경로 생성.

④ ③에서 완성하지 않은 다음 줄의 나머지 방들을 구체화한다. 바로 전 줄과 수직의 경로로 연결된 방들 이외에 나머지 방들은 각각 서로 다른 집합에 속하게 한다. 예를 들어 [그림 7]과 같이 [그림 6]에서 구체화되지 않은 두 번째 줄의 2,4번 째 방은 각각 새로운 집합 6,7에 속하게 된다.



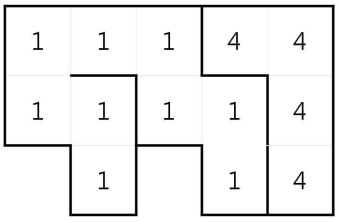
[그림 7] 두 번째 줄의 구체화.

⑤ 마지막 줄에 도달할 때까지 ②~④의 과정을 반복한다. [그림 8]은 [그림 7]의 두 번째 줄에 대하여 ②와 같이 서로 다른 집합에 속한 두 방 사이의 벽들 중 제거할 벽을 임의의 개수만큼 무작위로 골라 제거한 모습이다. ②의 과정에서 두 방 사이의 벽은 벽을 사이에 두고 인접한 두 방이 서로 다른 집합에 속했을 때만 제거할 수 있다. 따라서 [그림 8]의 2행2열의 방과 2행3열의 방은 인접해 있으나 서로 같은 집합 1에 속해있으므로 그 사이의 벽을 제거해서는 안 된다.



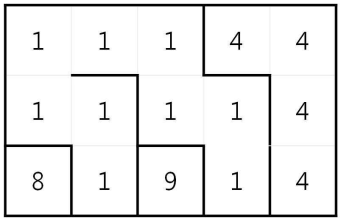
[그림 8] 두 번째 줄의 벽 제거.

[그림 9]는 [그림 8]에서 ③과 같이 미로의 다음 줄로 수직 경로를 생성한 뒤의 모습이다. [그림 9]에서 집합 4에 속한 방은 2행5열의 방 하나이므로 집합 4에서 하나 이상의 수직 경로를 만들기 위해 이 방은 반드시 인접한 3행5열의 방과 연결되어야 한다.



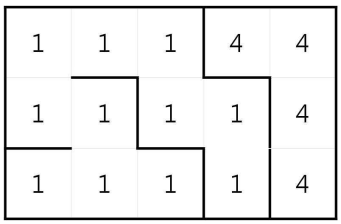
[그림 9] 두 번째 줄과 세 번째 줄 사이의 수직 경로 생성.

[그림 10]은 ④의 과정을 거쳐 5x3 미로의 마지막 줄까지 구체화시킨 모습이다.

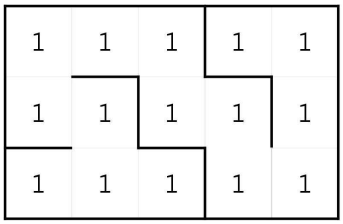


[그림 10] 마지막 줄의 벽 제거 이전 상태.

⑥ 미로의 마지막 줄에서는 인접해 있으며 서로 다른 집합에 속한 방들 사이의 모든 벽을 제거한다. 따라서 이 과정을 수행한 후 크기 $N \times M$인 미로의 모든 NM개의 방은 같은 집합에 속하게 된다. ⑥의 과정에서도 ②와 마찬가지로 두 방 사이의 벽은 벽을 사이에 두고 인접한 두 방이 서로 다른 집합에 속했을 때만 제거할 수 있다. 따라서 [그림 10]에 나타난 3행1열, 3행2열, 3행3열의 방은 모두 서로 다른 집합에 속해있으므로 그 사이의 벽들을 제거해야 하나 이렇게 해서 3행의 1,2,3열이 같은 집합 1로 합쳐질 경우 [그림 11]과 같이 3행3열과 3행4열은 같은 집합에 속하게 되므로 3행3열과 3행4열 사이의 벽은 제거할 수 없다. [그림 12]는 ⑥을 거쳐 완성된 5x3 완전 미로이다.



[그림 11] 3행의 1,2,3열의 방을 연결한 상태.



[그림 12] Eller's algorithm을 통해 완성된 5x3 미로.

3-2 프로젝트 수행

문제 해결에서 언급된 방법들 중 하나를 고르거나 자신이 새로운 방법을 고안하여 다음과 같은 단계를 거쳐 프로그램을 설계하고 이를 구현한다.

1. 간단한 예를 만들어 이에 자신이 설계한 자료구조를 적용하여 검토한다.
2. 잘못된 입력에 대해 적절한 에러 메시지를 출력해준다.
3. 주어진 예에 작성한 프로그램을 수행시켜보고 필요한 경우 디버깅 과정을 수행하여 오류를 수정한다.

4. 숙제 및 보고서 작성

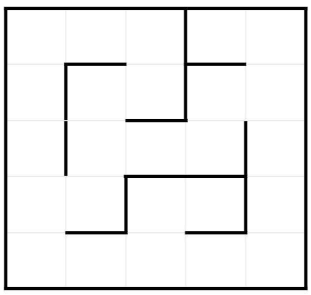
4-1 예비보고서

1. 문제 해결에서 언급한 미로 생성 알고리즘들에서 Eller's algorithm을 제외한 나머지 알고리즘 중 하나를 선택하여 이를 조사하고 이해한 후 그 방법을 기술한다.
2. 본 실험에서 완전 미로(perfect maze)를 만들기 위하여 선택한 알고리즘 구현에 필요한 자료구조를 설계하고 기술한다. 설계한 자료구조를 사용하였을 경우 선택한 알고리즘의 시간 및 공간 복잡도를 보인다.
3. 작성한 예비보고서는 실험 시작 전 제출한다.

4-2. 숙제

순환 경로가 존재하는 불완전 미로 생성 문제

실험 시간에 해결한 완전 미로 생성 문제를 바탕으로 순환 경로가 존재하는 불완전 미로를 출력하는 프로그램을 구현한다. 따라서 구현하고자 하는 미로에는 폐쇄된 공간은 존재하지 않으나 [그림 13]과 같이 어떠한 두 지점을 연결하는 경로가 하나 이상 존재한다. 너비 N, 높이 M인 $N \times M$ 불완전 미로를 생성할 경우 $N \times M$ 완전 미로에서 추가적으로 제거할 수 있는 벽의 개수는 $\lfloor MIN(N,M)/2 \rfloor$ 개여야 한다.



[그림 13] 순환 경로가 있는 불완전 미로.

입력형식: 입력은 표준입력을 이용한다. 입력 첫 줄에는 생성하려는 미로의 너비 N이 주어지고 이어지는 줄에는 미로의 높이 M이 주어진다.

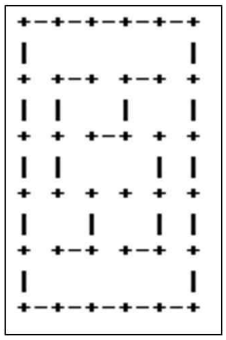
출력형식: 입력된 너비와 높이를 갖는 $N \times M$ 미로를 text 파일로 출력한다. 출력되는 text 파일의 확장자는 '.maz'이다. 미로는 방과 벽으로 이루어져 있는데, 각 방의 모서리는 '+'(plus sign), 가로 벽은 '-'(minus sign), 세로 벽은 '|'(vertical bar) 그리고 방은 ' '(space)로 나타낸다(아래의 예 참조).

입출력 예:

입력

```
5
5
```

출력



[그림 14] 숙제 입출력 예.

4.3 결과 보고서

아래 보인 사항을 작성하여 조교가 공지한 기간까지 제출한다.

1. 실험시간에 작성한 프로그램의 알고리즘과 자료구조를 요약하여 기술한다. 완성한 알고리즘의 시간 및 공간 복잡도를 보이고 실험 전에 생각한 방법과 어떻게 다른지 아울러 기술한다.
2. 숙제문제를 해결하기 위한 알고리즘 및 자료구조를 요약하여 기술하시오. 시간 및 공간 복잡도를 보인다.
3. 작성한 숙제 프로그램을 공지한 제출요령에 의거하여 기한 내 제출한다. 요청이 있을 경우 실험시간에 작성한 프로그램도 아울러 제출한다.

PRJ-2(1주차/3주) 예비보고서

전공: 학년: 학번: 이름

PRJ-2(1주차/3주) 결과보고서

전공: 학년: 학번: 이름