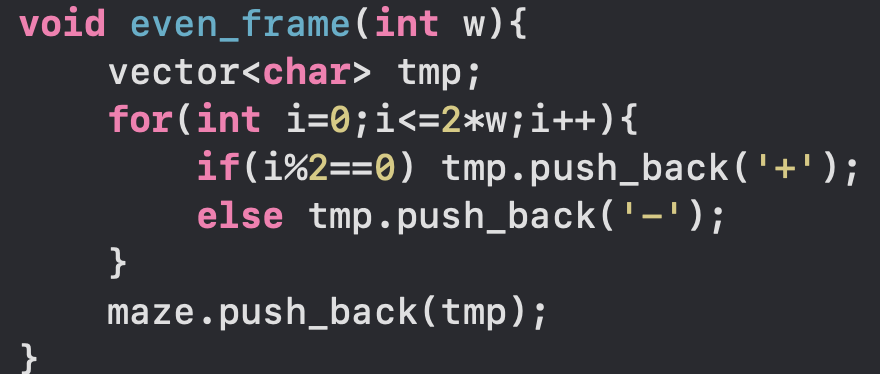
11주차 결과보고서

전공: 아트앤테크놀로지학과 학년: 3학년 학번: 20201116 이름: 이수빈

1. maze의 기본 틀 만들기
   * **even frame**

even\_frame 함수는 maze에서 짝수 row에 해당하는 부분을 만드는 함수이다. 이 함수는 위, 아래로 통하는 벽을 만든다. ‘+’와 ‘-‘을 반복으로 그리면 된다.



* + **odd\_frame**

함수는 maze에서 홀수row에 해당하는 부분을 만드는 함수이다. 이 함수는 양옆으로 로 통하는 벽을 만든다. ‘|’와 ‘ ‘(빈칸)을 반복으로 그리면 된다. 빈칸은 cell 하나를 의미한다.

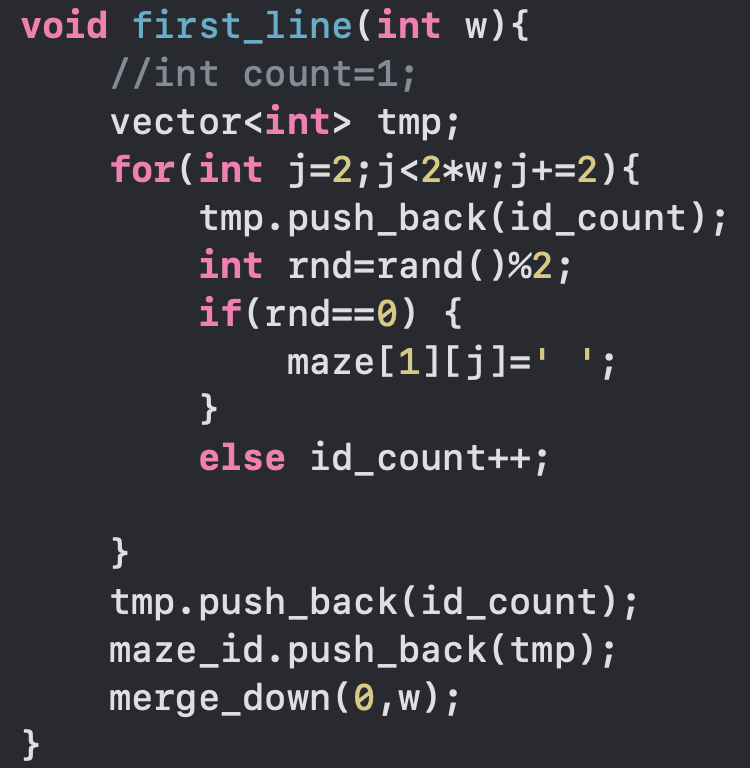
Text

Description automatically generated

1. 미로 만들기
   * **first\_line**

첫 번째 줄은 다른 줄과 다른 방법으로 생성하므로 따로 함수를 정의해 주었다.

각 column의 cell 마다 다른 maze\_id를 부여하는 것을 원칙으로 한다. 단, 랜덤으로 이웃하는 cell사이의 벽의 병합 여부를 정한 다음, 병합 시에는 maze\_id를 같게 한다. 모든 column에 대하여 위 과정을 반복한 후 다른 row에서도 똑같이 쓰이는 merge\_down 함수를 호출한다. 이 함수는 뒤에 설명을 하겠다.



* + **merge\_right**

|  |  |  |
| --- | --- | --- |
| 1 | | 2 |
|  | 3 | 4 |

첫 번째 줄을 제외한 나머지 줄에 대해서는 매 row마다 merge\_right함수를 호출한다. 이 함수서는 몇 가지 규칙을 따르며 벽의 병합 여부를 랜덤으로 결정한다. rand()함수를 호출하여 만약 짝수이면 병합을 하고 홀수이면 벽을 유지하는 방식으로 진행한다. 지켜야 하는 규칙은 왼쪽과 바로 위 cell의 maze\_id가 같다면 병합하지 않는 것이다.



위와 같은 상황에서 주황색 벽을 기준으로 왼쪽 cell과 오른쪽 위의 cell의 maze\_id는 같이 때문에 병합하면 안된다. 따라서 만약, rand 함수가 짝수 난수를 생성하여 병합을 해야 하더하도 왼쪽과 위의 maze\_id 같지 않을 때만 병합을 진행하게 한다. 또한 통일성을 위해 병합 시, maze\_id가 작은 수로 maze\_id로 통일한다.

Text

Description automatically generated

그리고 규칙을 지키며 함수를 실행하기 위해서는 현재 실행 중인 row와 그 위 row의 maze\_id도 업데이트 해주어야 한다.

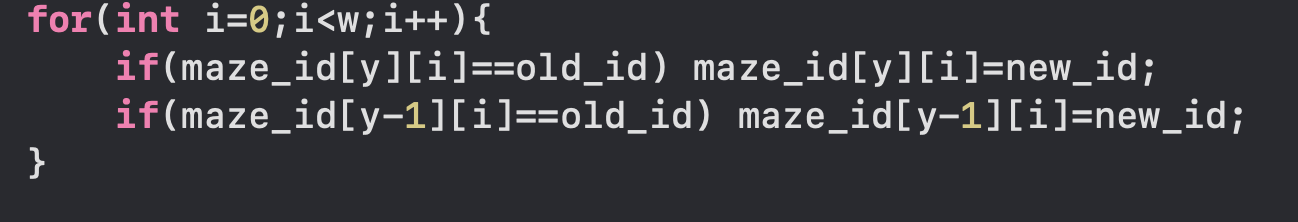
|  |  |  |  |
| --- | --- | --- | --- |
| 1  1 | 2 2 | | 2  2 |
| 2 | 4 |

예를 들면 위의 주황색에 해당하는 벽을 병합하려 한다 하자. 그러면 아래와 같이 해당 줄의 maze\_id 를 업데이트 해준다. 이는 모든 줄을 훑으며 병합 되는 두 cell의 maze\_id 중 더 큰 수와 같은 maze\_id를 갖은 cell을 병합 되는 두 cell의 maze\_id 중 더 작은 수로 업데이트 해주면 된다.

|  |  |  |  |
| --- | --- | --- | --- |
| 1  1 | 1 2 | | 2  1 |
| 1 | 4 |

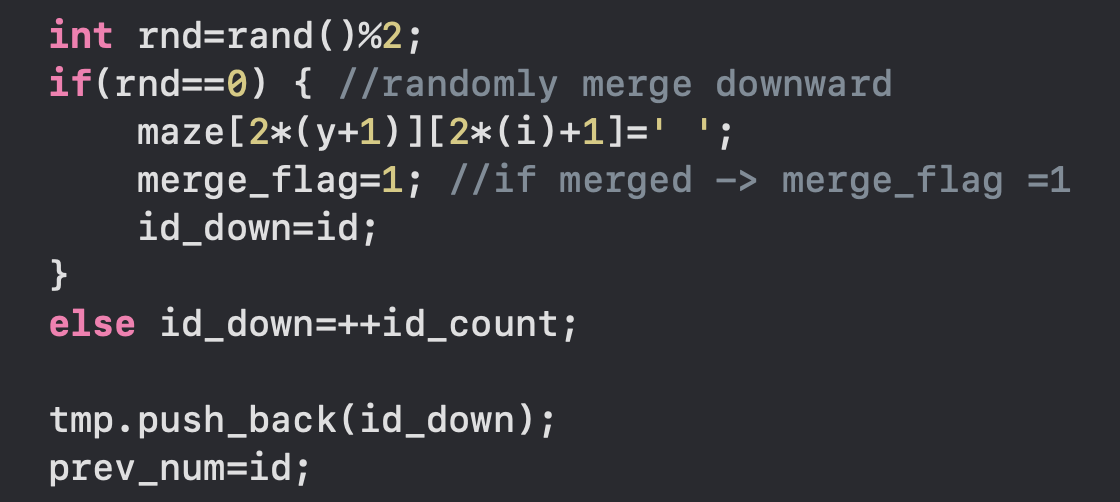
그 후 그 위의 줄도 업데이트 해준다. 이도 마찬가지로 모든 줄을 훑으며 병합 되는 두 cell의 maze\_id 중 더 큰 수와 같은 maze\_id를 갖은 cell을 병합 되는 두 cell의 maze\_id 중 더 작은 수로 업데이트 해주면 된다.

|  |  |  |  |
| --- | --- | --- | --- |
| 1  1 | 1 1 | | 1  1 |
| 1 | 4 |

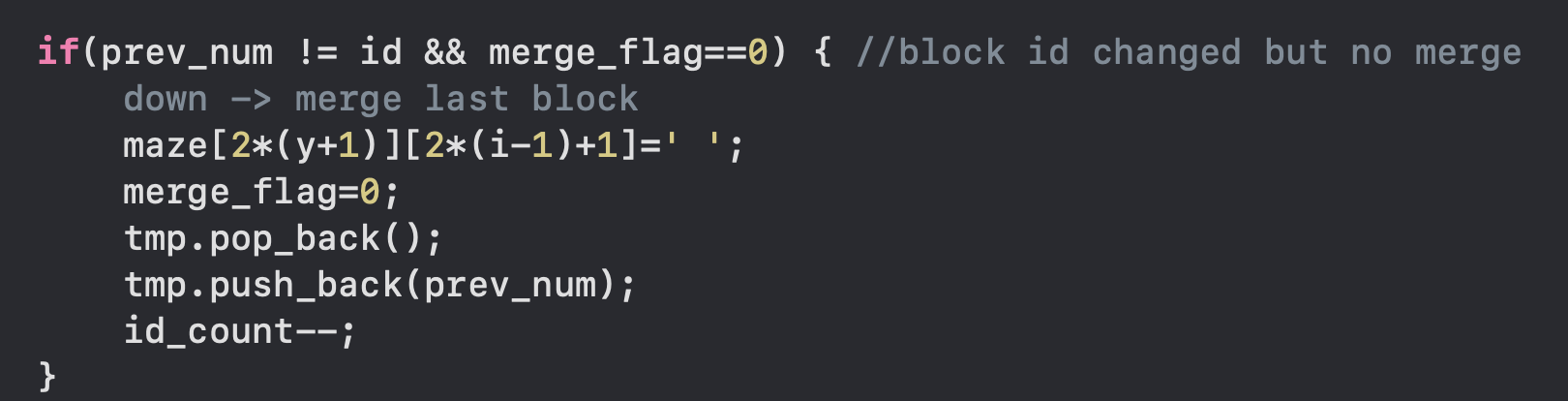


이 과정을 거쳐 maze\_id를 업데이트 해준다. 해당 row의 모든 cell에 대하여 위 과정을 완료하였다면 merge\_down 함수를 호출한다.

* + **merge down**

이 함수는 위, 아래 cell의 병합 여부를 결정해 주는 함수이다. 이도 마찬가지로 rand함수를 이용하여 랜덤하게 짝수이면 병합, 홀수이면 병합하지 않는 방식으로 진행한다. 

단, 같은 maze\_id를 갖은cell 중 적어도 하나의 cell은 아래로의 경로를 생성해야 한다. 이를 구현하기 위해 merge\_flag라는 변수를 설정하여 만약, 해당 maze\_id 그룹에서 적어도 하나의 cell이 아래로의 수직 경로를 설정하였다면 merge\_flag을 1로 설정한다. 만약, 해당 merge\_id 그룹의 마지막 cell까지 merge를 진행하였음에도 아직 merge\_flag가 0이라면, 즉 아무 cell도 아래와의 수직 경로를 설정하지 않았다면 강제로 아래로의 수직 경로를 설정해준다.

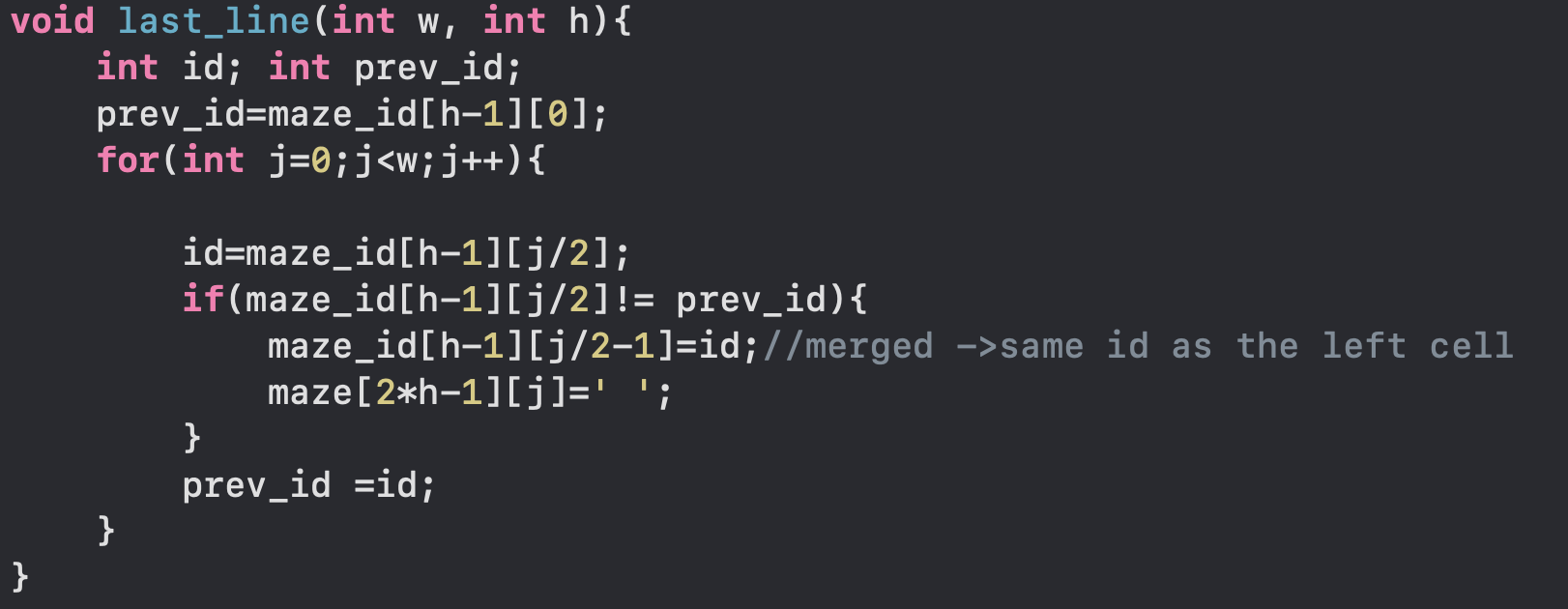


만약, merge\_id가 다음 그룹으로 넘어가였는데, merge\_flag 가 1이라면 다시 merge\_flag를 0으로 설정해 주어 다음 그룹에도 같은 방식으로 작동할 수 있게 해준다.



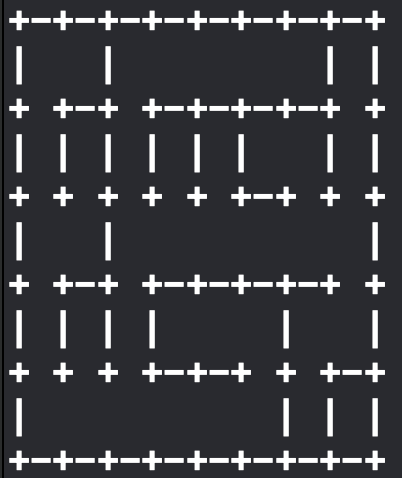
* + last\_line

마지막 줄에서는 다른 maze\_id를 갖고 있는 cell들을 무조건 병합해서 마지막 줄의 모든 cell의 maze\_id가 같도록 만들어 주어야 한다.



이를 위해 위와 같이 모든 cell을 검사하며 양옆의 maze\_id가 다르다면 병합한다.

위의 함수들을 이용하여 다음과 같은 미로를 생성할 수 있다. (입력값 : 8, 5)



예비 보고서에서 설명한 Kruskal 알고리즘과의 차이 점은 방법은 미로의 모든 칸을 한 트리의 노드로 생각하고 이의 MST를 찾는 방식이 아니라, 위와 같은 방식으로 미로를 생성한다는 것이다. 본 프로젝트에서 사용한 eller’s 알고리즘을 사용하면 시간 복잡도와 공간 복잡도는 다음과 같다.

시간 복잡도 : O(H/2\*W/2\*2) (모든 벽에 대해서 병합 여부를 결정하는 실행을 해야 하기 때문에 미로의 cell의 개수 \*2만큼의 시간 복잡도가 요구된다.)

공간 복잡도 : O(H\*W) (미로를 저장해야 하므로 미로의 크기 만큼의 공간이 필요하다.)