12주차 결과보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20191612 이름: 윤기웅

**1. (문제 및 제목은 작성하지 말 것. 답안만 작성하면 됩니다.)**

.................

readFile 함수 :

readFile 함수는 .maz 파일을 읽고 미로 데이터를 처리하는 역할을 합니다. 이 함수는 파일을 열고, 데이터를 읽어들여 미로 구조를 구성하며, 메모리를 동적으로 할당하여 미로 데이터를 저장하고, 인접 리스트를 생성합니다.

**함수의 시작 부분:** 파일 선택 다이얼로그를 열고 사용자가 파일을 선택하면 파일 이름과 경로를 저장합니다.파일이 .maz 확장자를 가지고 있는지 확인합니다.

**파일 존재 여부 확인:** 선택한 파일이 존재하는지 확인합니다. 존재하지 않으면 오류 메시지를 출력하고 false를 반환합니다. 파일이 존재하면 파일이 열렸음을 표시하는 isOpen 변수를 1로 설정합니다.

**파일 내용을 읽어 미로 크기 계산:** 파일의 각 라인을 읽어 미로의 행(hei)과 열(wid) 크기를 계산합니다. 미로의 실제 행(mazeRow)과 열(mazeColumn) 크기를 설정합니다. 동적으로 map과 input 배열을 할당합니다.

**메모리 할당 및 초기화:** input과 map 배열의 각 행을 동적으로 할당합니다. adjList와 visited 배열을 동적으로 할당하고 초기화합니다.

**파일 내용을 읽어 input과 map 배열에 저장:** 파일의 각 라인을 읽어 input 배열에 저장하고, 각 문자를 확인하여 map 배열에 1(길) 또는 0(벽)으로 저장합니다.

**인접 리스트 생성:** map 배열을 순회하며, 길(map[i][j] == 1)인 경우 각 위치에 해당하는 정점(vertex)을 생성합니다. 상하좌우 연결된 길이 있는지 확인하고, 인접 리스트에 노드를 추가합니다.

**디버깅용 출력 및 마무리:** 각 정점과 그 인접 리스트를 출력하여 디버깅합니다.

**결론:** 이 함수는 .maz 파일을 읽고 미로 데이터를 처리하여 input, map, adjList, visited 배열을 초기화합니다. 또한, 동적으로 할당된 메모리를 사용하여 데이터를 저장하고, 인접 리스트를 생성하여 미로의 연결 구조를 표현합니다.

Draw 함수 : `draw` 함수는 화면에 미로를 그리는 역할을 합니다. 함수의 시작 부분에서 배경색을 설정하고 색상과 선의 두께를 지정합니다. `weight` 변수는 미로의 각 셀의 크기를 나타냅니다. 이후 `ofNoFill` 함수를 호출하여 테두리만 그리도록 설정하고 `ofDrawRectangle` 함수를 사용하여 미로의 외곽을 그립니다. 미로의 벽을 그리기 위해 중첩된 while 문을 사용합니다. `mazeRows`와 `mazeCols` 변수의 값을 사용하여 각 셀을 확인합니다. 각 셀의 값이 '|' 또는 '-'인 경우, `ofDrawLine` 함수를 호출하여 해당 방향으로 선을 그립니다. 불필요한 문자에 대해서는 기본 case로 처리하고 아무 동작도 하지 않습니다. 이 과정을 통해 미로의 벽을 화면에 그립니다. 이후 dfs 알고리즘이 활성화된 경우, 색상과 선의 두께를 다시 설정하고, 파일이 열려 있는지 확인한 후 `dfsdraw` 함수를 호출합니다. 파일이 열려 있지 않으면 오류 메시지를 출력합니다. 마지막으로 `bShowInfo` 변수가 true인 경우, 화면 하단에 프로젝트 이름을 표시하는 문자열을 그립니다. `draw` 함수는 이렇게 화면에 미로를 그리는 모든 과정을 포함하고 있습니다.

................

freeMemory : freeMemory 함수는 할당된 메모리를 해제하는 역할을 합니다. 먼저 adjList의 각 노드를 순회하며 메모리를 해제합니다. i가 wid \* hei보다 작은 동안 while 루프가 실행되며, i를 3으로 나눈 나머지 값을 기준으로 switch 문이 실행됩니다. 각 case에서 현재 노드와 연결된 모든 노드를 해제합니다. 각 case 블록 내부에는 불필요한 계산이 포함되어 있지만, 이는 메모리 해제 과정에 영향을 미치지 않습니다. 노드를 모두 해제한 후 adjList[i]를 NULL로 설정합니다. 다음으로, mazeRow 크기만큼 for 루프를 실행하여 input과 map 배열의 메모리를 해제합니다. j를 2로 나눈 나머지 값을 기준으로 switch 문이 실행되며, input[j]와 map[j]를 번갈아가며 해제합니다. 마지막으로, visited 배열의 메모리를 해제합니다. 이 과정에서 불필요한 계산이 포함되어 있지만, 이는 메모리 해제 과정에 영향을 미치지 않습니다. freeMemory 함수는 이렇게 할당된 메모리를 해제하는 모든 과정을 포함하고 있습니다.

**2.**

노드를 만들어 인접리스트를 생성하면 시간 공간 복잡도가 다른 자료구조에 비해 더 적게 필요합니다. 특히 배열을 사용하는 것보다 더 적은 메모리 공간만 사용해서 미로의 그리기가 가능합니다.

.......................

.........................