프로그래밍 과제 II

※과제 제출마감 유의사항:
과제방 close 시간에 임박하여 과제물 제출을 시작하면 통신지연 등으로 업로드가 완료되지 못한채 과제방이 close될 수 있습니다.
과제방이 close되기 전에 업로드 완료된 제출만 정상 제출로 인정됩니다.
과제물을 충분한 시간 여유를 두고 제출하여 불이익을 피하기 바랍니다.

과제방 업로드 마감 및 과제방 close: 5월12일(수) 23시 59분

지연제출 감점 규정 및 제출마감 유의사항은 "과제 제출마감 유의사항" (3월17일자 eClass에 공지) 필독

교재 Program 3.12의 함수 path를 호출하여 경로찾기를 수행하면서 그 과정을 출력하는 maze 프로그램을 작성하시오. 단, 아래 내용을 반영하시오.
(a) 전역변수인 stack은 배열로 구현하지 않고 linked stack으로 구현한다.
(b) maze로는 입구좌표 (1,1)에 진입한 이후에 backtrack이 발생하지 않고, 입구에서 출구까지의 경로가 존재하는 것만 대상으로 한다.
(c) 현재 위치좌표 (row, col)의 값이 입구좌표로 진입 또는 legal move에 의해 변경될 때마다 현재 위치까지 찾은 경로를 maze 상에 *의 sequence로 표시하여 출력한다. **(아래 출력 포맷 및 예 설명 참조)**
(d) (c)의 출력은 GUI 도구 등을 사용하지 말고 단순히 printf 문으로 line 단위의 출력으로 수행한다. 이를 위해 현재까지 찾은 경로의 위치 좌표에 대한 정보를 linked list로 유지한다. **(아래 예 설명 참조)**
(e) 교재의 함수 path는 프로그램 종료 직전에 최종 찾은 경로를 구성하는 (i, j) 좌표 sequence를 출력하는데 본 과제에서는 이 출력을 생략한다.
(f) 대신 프로그램 종료 직전에 최종 찾은 경로를 입구부터 출구좌표까지 maze 상에 *의 sequence로 표시하여 출력한다.

maze 입력: 5 x 5 maze로 한다.

```
#define numRow 5
#define numCol 5
```

아래의 setup_maze() 함수를 호출하여 2차원 배열 maze[umRow+2][numCol+2]의 설정을 완료한다.

```
void setup_maze() {
        short int maze0[numRow][numCol] = { //여기서 줄 바꾸어 주세요(1).
                { 0,1,1,1,1 },
                { 1,0,1,0,1 },
                { 1,0,0,1,0 },
                { 0,0,1,0,1 },
                { 1,0,0,1,0 } //여기서 줄 바꾸어 주세요(2).
        };
        //2차원 배열 maze0[numRow][numCol]로부터
        //전역변수인 2차원 배열 maze[numRow+2][numCol+2]를 설정 완료하는 코드
}
```

유의: 채점 시 채점자는 위 코드에서 "//여기서 줄 바꾸어 주세요(1)." 이후부터 "//여기서 줄 바꾸어 주세요(2)." 이전까지의 다섯 line을 다른 maze로 값만 변경된 line들로 대체하고 컴파일 및 실행한다.

maze 및 현재 좌표까지 찾은 경로의 출력:
좌측과 상단에 row 및 col 번호를 출력한다. maze에서 막힌 좌표 (즉, maze[i][j]=1)는 x 출력, 열린 좌표 (즉, maze[i][j]=0)는 공백을 출력한다. 입구좌표 (1,1)부터 현재 위치 좌표 (row, col) 까지의 경로를 구성하는 좌표 위치마다 별 * 를 출력한다. maze와 경로의 출력이 완료될 때마다 "continue(Y/N)? :"로 프로그램의 계속 진행 여부를 묻고 답을 입력받는다.

예1: 위의 setup_maze() 함수에서 예로 든 maze의 경우, 입구좌표 (1,1)에 진입하였을 때 아래와 같이 출력한다.

```
row/col  1  2  3  4  5
   1     *  x  x  x  x
   2     x     x     x
   3     x        x
   4              x     x
   5     x        x
continue(Y/N)? :
```

예2: 예1의 상태에서 legal move에 의해 현재 위치를 (2,2)로 이동하였을 때 아래와 같이 출력한다.

```
row/col  1  2  3  4  5
   1     *  x  x  x  x
   2     x  *  x     x
   3     x        x
   4              x     x
   5     x        x
continue(Y/N)? :
```

Note:
예1에서 총 7 line이 이미 출력되었다. 예2의 출력은 예1의 출력 이후 line들에 7 line을 이어서 출력하는 것이다. 즉, 예1과 예2에 걸쳐 현재까지 줄 바꾸기를 제외하고 총 14 line이 출력된 것이다. GUI 도구 등을 사용하여 예1에서 수행한 출력 화면 상에서 (2,2) 위치에 * 표시를 추가하는 것이 아니다. 단순히 printf 문으로 line 단위의 출력을 수행한다.

예3: 예2의 상태에서 legal move에 의해 현재 위치를 (3,3)으로 이동하였을 때 아래와 같이 출력한다.

```
row/col  1  2  3  4  5
   1     *  x  x  x  x
   2     x  *  x     x
   3     x     *  x
   4              x     x
   5     x        x
continue(Y/N)? :
```

예4: 예3의 상태에서 계속 경로찾기가 진행되다가 현재 위치를 (4,4)로 이동하였을 때 아래와 같이 출력한다. (예3 이후부터 예4 이전까지의 출력화면 예시는 생략: 현재 위치 legal move의 순서는 (3,3)->(2,4)->(3,5)->(4,4))

```
row/col  1  2  3  4  5
   1     *  x  x  x  x
   2     x  *  x  *  x
   3     x     *  x  *
   4           x  *  x
   5     x        x
continue(Y/N)? :
```

예5: 예4에서 경로찾기는 종료된다. 프로그램을 종료하기 전 최종 찾은 경로를 입구부터 출구까지 *의 sequence로 표시하기 위하여 출구좌표 (5,5)에도 *을 표시하여 아래와 같이 최종 출력한다.

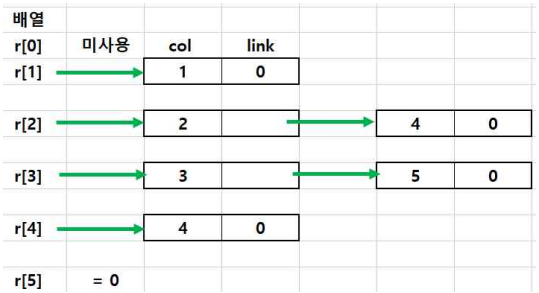
```
row/col  1  2  3  4  5
   1     *  x  x  x  x
   2     x  *  x  *  x
   3     x     *  x  *
   4           x  *  x
   5     x        x  *
continue(Y/N)? :
```

maze 및 경로 출력을 위한 linked lists:

예4의 출력의 경우를 보면, 경로의 좌표 순서는 (1,1)->(2,2)->(3,3)->(2,4)->(3,5)->(4,4) 이지만 출력은 line 단위로 진행되므로 위 line 부터 아래 line 순으로 출력해야 한다. * 표시가 출력되는 좌표의 순서는 (1,1)->(2,2)->(2,4)->(3,3)->(3,5)->(4,4)가 되어 경로 좌표 순서와 다르다.

이러한 출력을 위해 1번부터 5번까지 각 row 별로 * 표시된 위치좌표의 col 값의 linked list를 경로찾기가 동작하는 과정에서 계속 유지하여 출력 시 사용한다. 아래 그림의 우측은 예4의 출력을 수행할 때 이들 lists의 내용을 나타낸 것이다.

r[1], ..., r[5]는 NULL로 (즉, 노드가 하나도 없는 empty list로) 초기화한다. 이후, 입구좌표 진입 또는 legal move에 의해 현재 좌표값이 (i, j)로 변경될 때, 리스트 r[i]에 col 값이 j인 노드를 삽입한다. (부록의 예시 참조)

예4의 출력 화면	각 row 별 col 값의 linked list
row/col 1 2 3 4 5 1 * x x x x 2 x * x * x 3 x * x * 4 x * x 5 x x continue(Y/N)? :	

maze 및 경로 출력 구현의 제약사항: row i에서 *의 colum 위치 정보를 linked list r[i]에 유지하고 있다. 각 row의 출력에는 공백 또는 x 또는 * 문자가 사용된다. row i의 출력을 수행하는 여러 방법 중 다음은 사용하지 마세요: temporary 배열 row_char[1],..., row_char[5]를 모두 0으로 초기화한 후, 리스트 r[i]를 traverse하면서 노드에 기록된 col=j 일 때 row_char[j]=1 로 set한다. 이후 k=1부터 5까지 loop을 돌면서 row_char[k]=1 이면 *, row_char[k]=0 AND maze[i][k]=0 이면 공백, row_char[k]=0 AND maze[i][k]=1 이면 x를 출력한다. 미사용 이유: 배열 row_char[]의 할당에 O(numCol)의 공간 사용이 요구된다. O(numCol)의 공간 사용이 요구되지 않는 방법으로 출력을 코딩해 보세요.

프로그래밍 언어: C 언어로 한정

교재 C 코드 사용시 유의사항:
교재의 C 코드는 사용하는 컴파일러 및 버전에 따라 일부 수정이 필요할 수 있음에 유의 (예: scanf, 헤더파일 등). 이 경우 컴파일 및 실행에 아무 문제가 없도록 먼저 수정하는 작업이 필요합니다.

C 코드 제출 준수사항: 조교의 "프로그램 과제 제출 및 채점 안내" (3월17일자 eClass에 공지)에 따름

Visual Studio 버전: 조교의 "프로그램 과제 제출 및 채점 안내" (3월17일자 eClass에 공지)에 따름

제출물:
1. 레포트 파일 (파일 형식은 pdf)
2. 소스코드 파일: .c 또는 .txt 화일로 제출. (소스코드만 조교 컴퓨터의 VS 프로젝트로 복사되어 컴파일 및 실행 예정)

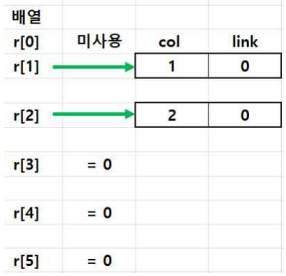
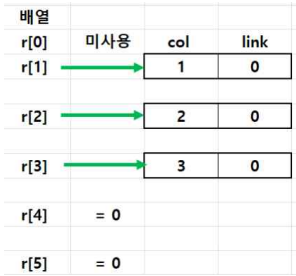
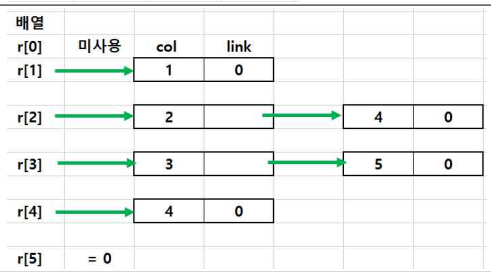
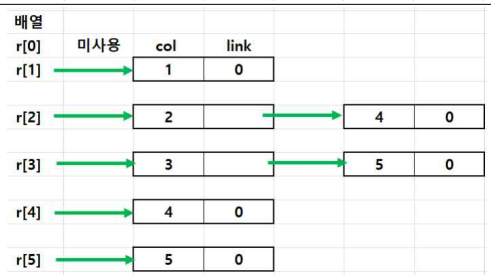
레포트 내용 구성:
1. 본 과제 내용과 관련하여 자신의 주요 C 코드에 대한 설명 (해당 부분 C 코드 캡처이미지를 삽입하고 설명)
특히, 교재의 함수 path()에서 수정되거나 새로 작성된 코드를 중심으로 설명
제출한 C 프로그램의 전체 코드를 설명할 필요는 없습니다.
2. 프로그램 실행결과 화면 캡처 및 설명: 서로 다른 maze 2개에 대한 실행 결과
3. 가정(assumption) 등 기타사항 (필요시)

기타 사항: 수업시간에 설명한 내용에 따름

제출처: eClass 과제방

제출마감 및 과제방 close: 5월12일(수) 23시59분
제출마감에 임박하여 과제물 업로드를 시도하면 통신지연 등으로 제출을 완료하지 못한채 과제방이 close될 수 있습니다.
과제방이 close되기 전에 업로드 완료된 제출만 정상 제출로 인정됩니다.
과제물을 충분한 시간 여유를 두고 제출 완료하여 불이익을 피하기 바랍니다.
지연제출 감점 규정 및 제출마감 유의사항은 "과제 제출마감 유의사항" (3월17일자 eClass에 공지) 필독

부록:

	출력 화면	각 row 별 col 값의 linked list	비고
초기화			r[i]=0 에서 0 은 NULL pointer
예1	row/col 1 2 3 4 5 1 * x x x x 2 x x x 3 x x 4 x x 5 x x continue(Y/N)? :		① 입구좌표 (1,1)에 진입 시, r[1]에 col=1 인 노드 삽입 ② 이후 화면 출력
예2	row/col 1 2 3 4 5 1 * x x x x 2 x * x x 3 x x 4 x x 5 x x continue(Y/N)? :		① 현재 위치를 (2,2)로 이동시, r[2]에 col=2 인 노드 삽입 ② 이후 화면 출력
예3	row/col 1 2 3 4 5 1 * x x x x 2 x * x x 3 x * x 4 x x 5 x x continue(Y/N)? :		① 현재 위치를 (3,3)으로 이동시, r[3]에 col=3 인 노드 삽입 ② 이후 화면 출력
예4	row/col 1 2 3 4 5 1 * x x x x 2 x * x * x 3 x * x * 4 x x * 5 x x continue(Y/N)? :		① 현재 위치를 (4,4)로 이동시, r[4]에 col=4 인 노드 삽입 ② 이후 화면 출력
예5	row/col 1 2 3 4 5 1 * x x x x 2 x * x * x 3 x * x * 4 x * x 5 x x * continue(Y/N)? :		① 프로그램 종료 전 r[5]에 col=5 인 노드 삽입 ② 이후 화면 최종 출력