Group Activity 03

(3인 혹은 4인으로 팀을 구성하여 아래의 문제를 푼다. 팀 구성은 매 시간마다 달라져도 된다.)

팀원1: _____

팀원2: _____

팀원3: _____

팀원4: _____

1. C 혹은 C++로 지난 주 프로그래밍 과제를 수행할 때 다음의 2가지 자료구조를 고려할 수 있다.

```
struct item {
    char *word;
    char *explanation;
};
struct item dict[MAX];
```

혹은

```
struct item *dict[MAX];
```

즉, 배열에 객체 자체를 저장하거나 혹은 그 객체의 주소를 저장하는 것이다. (C++에서 `vector` 등을 사용하더라도 본질적으로는 마찬가지이다.) 이 두 가지 방법의 장단점을 C 혹은 C++의 관점에서 논하라. 논의를 지난 주 프로그래밍 과제에 한정하지 말고 가능한 한 일반화하라.

2. 다음은 출발점 (0,0)으로부터 출구 $(N-1, N-1)$까지 도달하는 가장 긴 경로의 길이를 구하여 반환하는 함수이다. 경로가 존재하지 않으면 −1을 반환한다. 미로에서 통로는 0, 벽은 1이라고 가정한다. 맨 처음 이 함수는 `MazePath(0, 0, 0)`으로 호출된다. 완성하라.

```
int MazePath(int x, int y, int len) {
   if (x<0 || y<0 || x>=N || y>=N || maze[x][y] != 0)
      return [        ];
   else if (x == N-1 && y == N-1) {
      return [        ];
   }
   else {
       maze[x][y] = 2;           // mark that (x, y) is visited









   }
}
```

3. 다음은 N-queen 문제를 푸는 순환함수이다. 이 함수를 변형하여 N-queen 문제의 서로 다른 해의 개수를 구하는 함수를 작성하라. `promising` 함수는 작성할 필요가 없다.

```
int cols[N+1];
bool queens( int level )     {
  if (!promising(level))
    return false;
  else if (level==N)
    return true;
  for (int i=1; i<=N; i++) {
    cols[level+1] = i;
    if (queens(level+1))
      return true;
  }
  return false;
}
```

4. 하나의 그래프와 사용할 색의 개수 m이 입력으로 주어진다. 그래프의 각각의 노드를 m개의 색 중 하나로 칠하면서 어떤 인접한 두 노드도 동일한 색으로 칠해지지 않도록 할 수 있는지 검사하는 문제이다. 예를 들어 아래의 그림은 $m = 3$개의 색으로(red, green, white) 칠한 결과이다. 되추적(backtracking) 기법으로 이 문제를 해결하려고 한다. 적절한 상태공간트리를 구상하고 알고리즘을 pseudocode의 형태로 기술하라.