실습 보고서

|  |  |
| --- | --- |
| 과목 | 응용소프트웨어실습 |
| 과제 | 기말고사 |
| 제출일 | 2020-06-26 |
| 소속` | 컴퓨터소프트웨어학부 |
| 학번 | 2015726012 |
| 이름 | 최세일 |

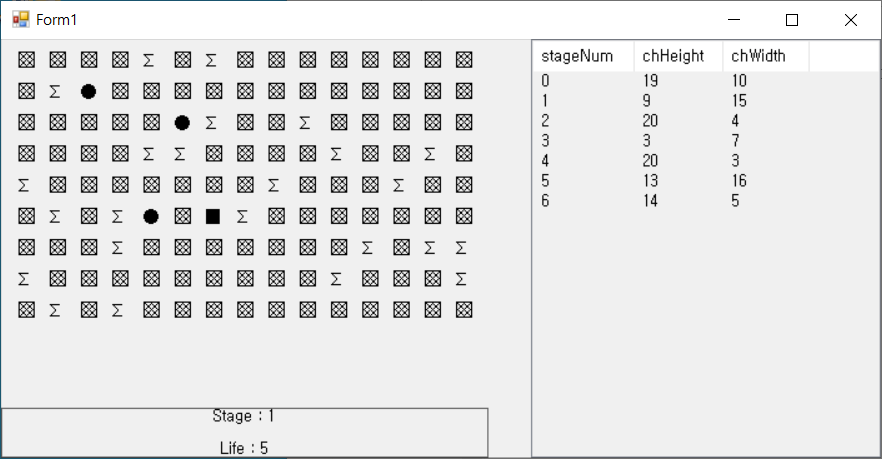


**목차**

1. **체크리스트**
2. **폼 작성**
3. **스테이지 구현**
4. **포탈 및 출구 구현**
5. **플레이어 구현**
6. **몬스터 구현**
7. **결과창 구현**
8. **고찰**
9. **체크리스트**

|  |  |  |
| --- | --- | --- |
| 내용 | | 구현 여부 |
| 1. 폼작성 | 1) 게임 판 작성 | O |
| 2) 결과 창 작성 | O |
| 1. 스테이지 구현 | 1) 스테이지 수는 최소 2개, 최대 10개. 매번 실행할 때마다 랜덤으로 결정된다. | O |
| 2) 스테이지의 너비는 최소 3칸, 최대 20칸이며, 높이는 최소 3칸, 최대 20칸. 각 스테이지의 너비와 높이는 랜덤으로 결정된다. | O |
| 3) 포탈을 타고 이동할 때 이전 스테이지의 정보가 남아있어야 한다. | O |
| 1. 포탈 및 출구 구현 | 1) 포탈의 위치는 스테이지 내 발판 위에서 무작위로 결정된다. 포탈의 수 또한 랜덤이며 최대 4개까지 생성될 수 있다 | O |
| 2) 포탈은 대응되는 스테이지로 플레이어를 이동시킬 수 있어야 하며, 플레이어가 이동되는 위치는 이동하는 스테이지에서 이전 스테이지로 이동할 수 있는 포탈의 전후좌우 위치 중 가능한 위치여야 한다 | O |
| 3) 출구는 무작위 스테이지에 생성되며, 출구까지 도착할 수 있는 포탈 경로가 반드시 하나 이상 존재해야 한다 | O |
| 4) 생성된 모든 스테이지에는 반드시 하나 이상의 포탈이 존재해야 한다 | O |
| 5) 자기 자신으로 이동하는 포탈은 없어야 하며, 2개 이상의 포탈이 같은 스테이지로 이동해서는 안 된다 | O |
| 1. 플레이어 구현 | 1) 플레이어의 초기 위치는 무작위다 | O |
| 2) 플레이어는 발판 위에서 상하좌우로 이동 가능하다 | O |
| 3) 플레이어가 포탈에 들어가면(키를 입력했을 때 포탈 위치에 해당하는 경우) 포탈이 가리키는 스테이지로 이동한다 | O |
| 4) 플레이어가 출구에 들어가면(키를 입력했을 때 출구 위치에 해당하는 경우) 플레이어는 게임에서 승리한다 | O |
| 5) 플레이어는 몬스터 위치로는 이동할 수 없다. | O |
| 5. 몬스터 구현 | 1) 몬스터는 사용자 입력이 들어온 후 행동한다 | O |
| 2) 몬스터는 상하좌우와 대각선 8개 방향 중 무작위 위치로 이동하며, 발판 위 무작위 위치에서 생성된다 | O |
| 3) 몬스터는 발판 위에서 이동 가능하며 포탈과 출구, 그리고 다른 몬스터 위치로는 이동할 수 없다. 행동이 불가능한 경우 자리를 유지한다 | O |
| 4) 몬스터가 이동했을 때 위치가 플레이어 위치라면 생명점수를 1 깎고 몬스터는 원래 위치를 유지한다 | O |
| 5) 몬스터의 최소 수는 0, 최대 수는 해당 스테이지 칸 수의 20%이다 | O |
| 6) 몬스터가 생명 점수를 0으로 만들었을 때 플레이어는 패배한다 | O |
| 6. 결과창 구현 | 1) 플레이어의 승패 여부에 따라 레이블을 다르게 표시해야 함 | O |
| 2) 결과창이 종료되기 전까지 게임판은 제어할 수 없음 | O |
| 3) 확인 버튼을 누르면 모든 창들이 꺼지고 프로그램을 종료해야 함 | O |

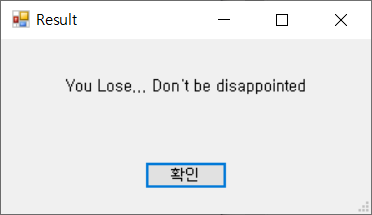
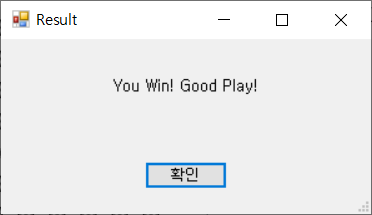
1. **폼 작성**
   1. 게임판 작성



* 오른쪽에서 현재 생성된 스테이지와 스테이지의 크기를 확인할 수 있도록 추가하였다.

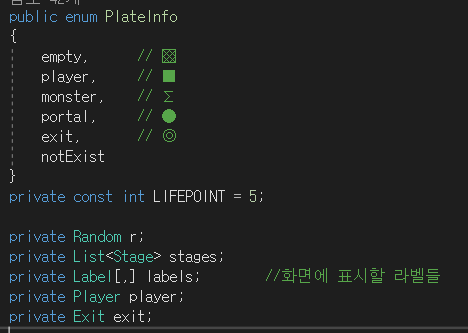
Form 의 크기는 현재 생성된 스테이지의 크기에 따라 조절되도록 하였다.

* 1. 결과창 작성



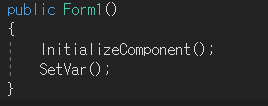
1. **스테이지 구현**

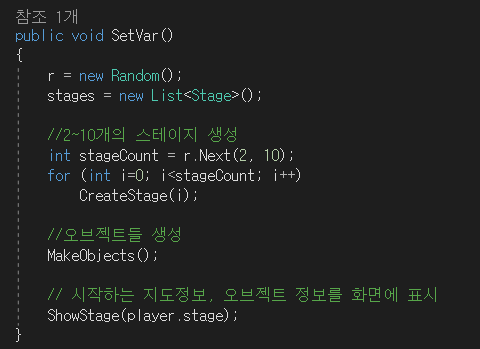
**1.** 랜덤 생성 **2.** 크기 지정 **3.** 스테이지 정보 저장



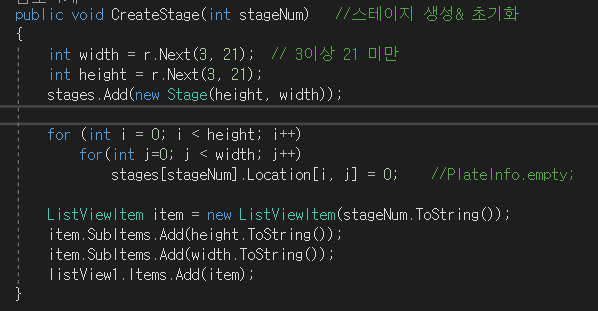
* 우선 스테이지의 오브젝트를 구분하기 위해 enum 을 선언해 주었고 필요한 변수들을 선언해 주었다.

그리고 별도로 Player, Stage, Portal, Monster 클래스를 생성해 주었다.

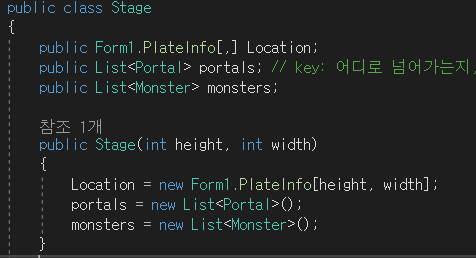




* 프로그램을 실행하게 되면 먼저 2~10개의 랜덤한 스테이지를 생성해 준다.

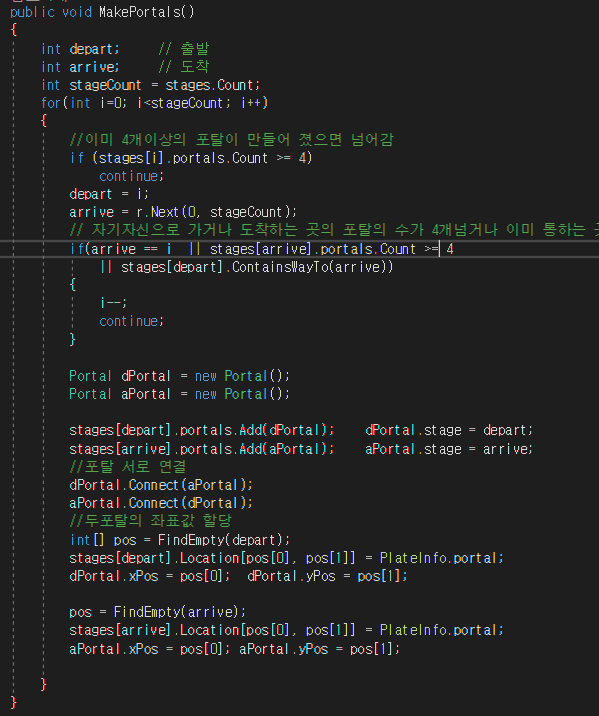


* 위의 SetVar() 메소드 안에서 루프를 돌면서 각 스테이지를 생성해준다.



* 생성된 스테이지는 그 스테이지만의 랜덤으로 정해진 높이와 너비, 몬스터, 포탈을 갖게 된다.

1. **포탈 및 출구 구현**

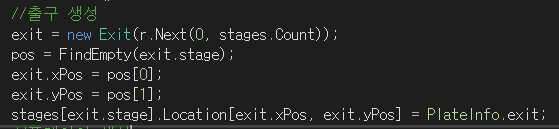
****

* 위의 MakeObject() 메소드 에서 MakePortals() 를 호출한다.

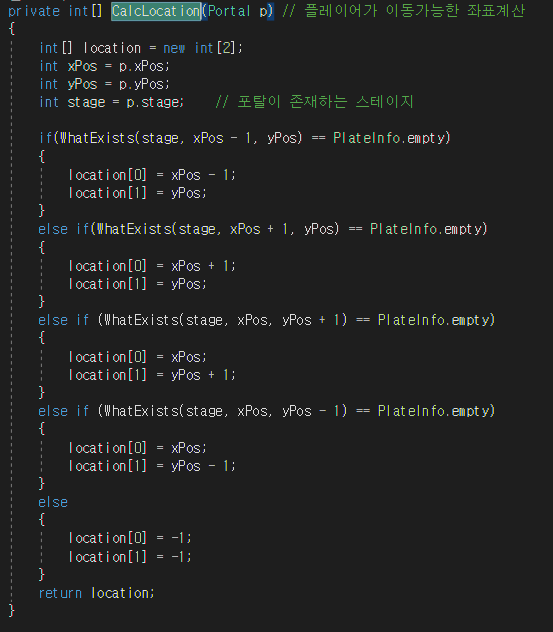
**1 랜덤 4개 생성**

위의 함수에서 포탈은 랜덤으로 생성된다. 생성될 때 해당 스테이지에 포탈의 개수를 파악하여 4개 미만일 때만 생성하게 된다.

**2. 포탈 위치 이동은 아래의 플레이어 이동에서 처리해 주었다.**



* 무작위 위치에 출구를 생성한다.



* 이동할 스테이지에 있는 포탈의 전후좌후 위치중 비어있는 자리를 찾아 리턴하도록 하였다.

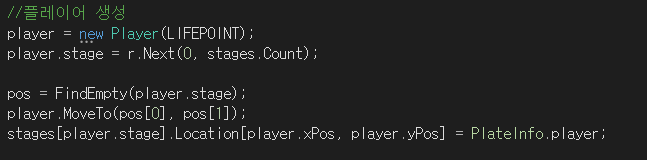
**3&4 출구로의 포탈 경로**

각 스테이지 별로 최소 하나의 포탈을 갖게 함으로써 출구까지의 경로를 확보하였다.

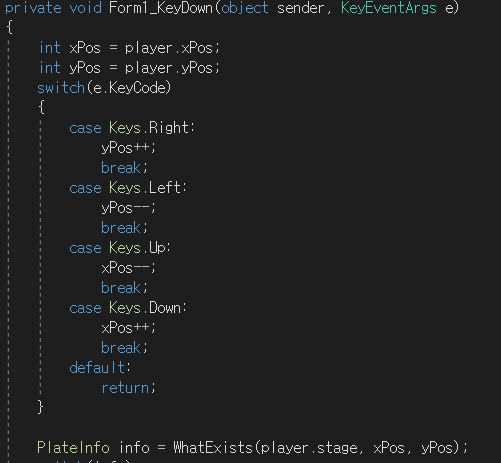
**5.**

포탈 생성시 랜덤으로 설정된 스테이지가 자기자신이거나 이미 통하는 경로라면 for문을 다시 돌도록 하였다.

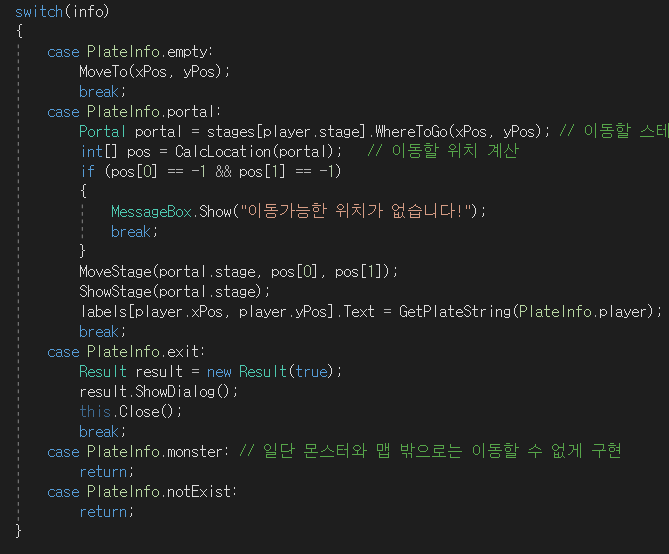
1. **플레이어 구현**



* **1. 플레이어 초기 위치 무작위로 생성**



* **플레이어의 이동을 위해 키값을 받아서 이동할 위치의 좌표를 계산하도록 하였다.**



* 플레이어 이동 처리.

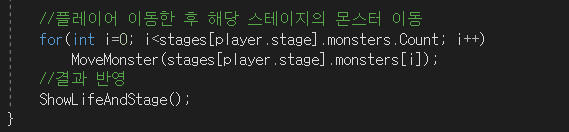
**2.** 만약 이동하려 한곳이 빈곳이면 이동한다.

**3.** 포털의 경우 플레이어가 있는 스테이지의 무슨포탈인지 구하고 그 포털이 가리키는 스테이지로 이동한다.

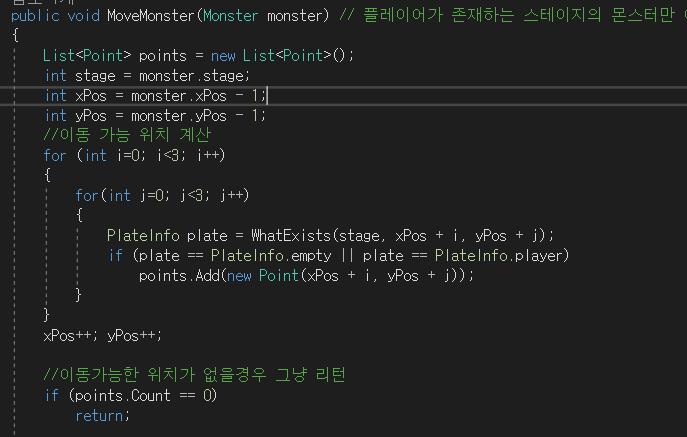
**4.** 출구의 경우 게임에서 승리하고 결과창을 띄운다

**5.** 몬스터 위치로는 이동할 수 없다.

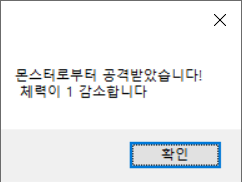
1. **몬스터 구현**

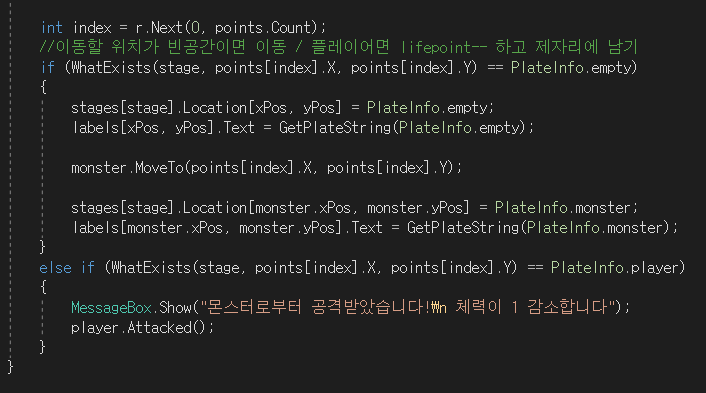


* 1. 위에서 플레이어가 이동한 이후 몬스터가 이동하도록 하였다.

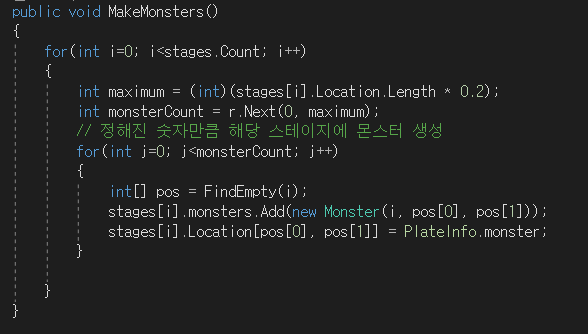


* 2. 몬스터의 이동처리 부분. 주위 8개 방향중 비어있거나 플레이어있는 위치의 좌표를 우선 받아오도록 하엿다. 이후에 그중에서 랜덤으로 이동할 위치를 정하게 된다. 만약 이동할 수 있는 위치가 없다면 제자리에 있는다.

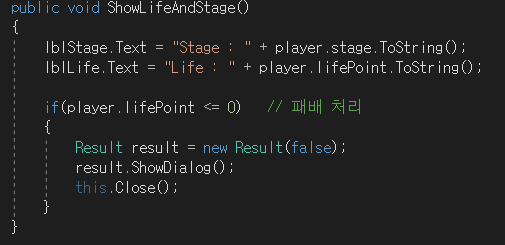




* 만약 플레이어가 있는 위치로 이동했을 경우 플레이어의 생명력을 1 깎는다.

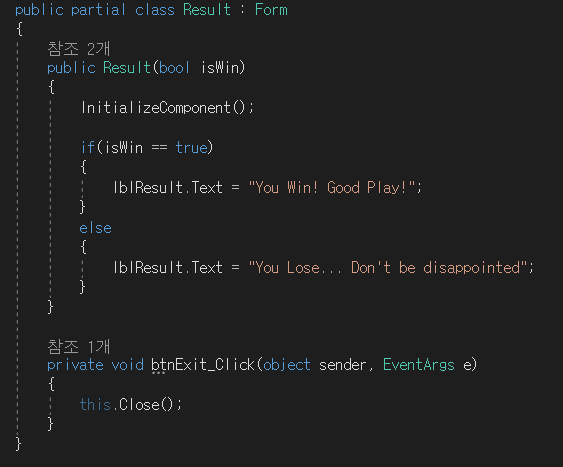


* 몬스터 랜덤 생성 부분. 위의 MakeObjects() 메소드에서 실행 된다.



* 이동시에 계속 호출되는 함수. 현재 스테이지와 남은 생명력을 표시한다. 만약 몬스터에게 공격받아 생명력이 0이 되었을 경우 결과창을 출력한다.

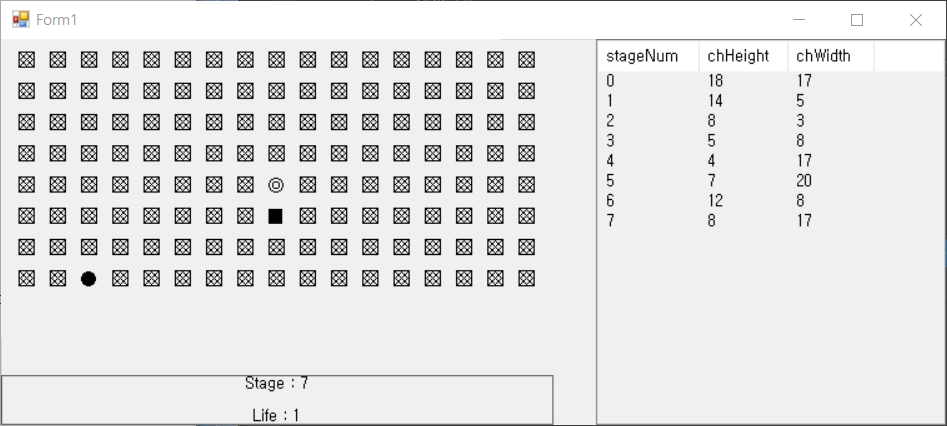
1. **결과창 구현**

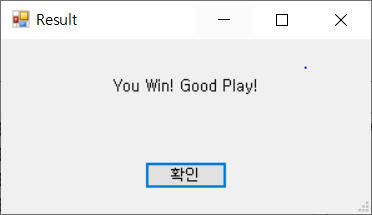


* 결과창 코드. 플레이어의 승리에 따라 다른 인자를 넣어주어 폼을 실행하도록 하였다.

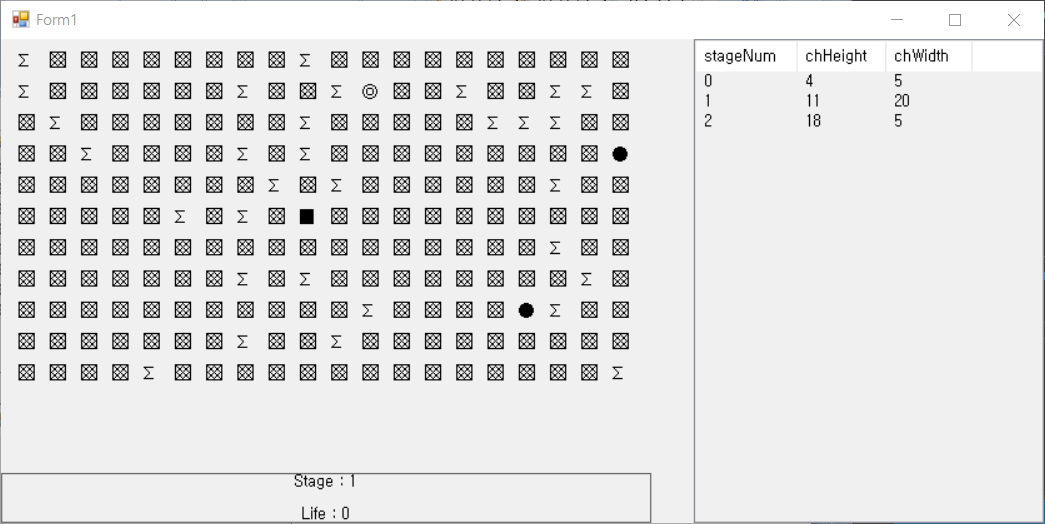
결과창이 종료되기 전에 게임판은 제어할 수 없도록 modal 창으로 띄워주었고

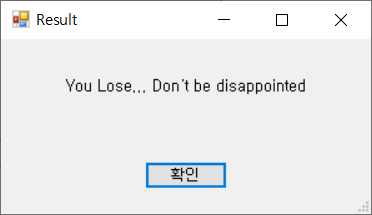
확인버튼을 누르게 되면 모든 form 을 종료하게 된다.

****

****

* 해당 위치에서 게임을 승리하는 모습

****

****

* 몬스터의 공격을 받아 게임오버 되는 모습

1. **고찰**

* 게임 실행중 현재 스테이지의 몬스터 수가 많다면 다소 딜레이가 발생하는 문제가 있다. 이점은 해당 스테이지의 모든 몬스터가 랜덤값을 계산하여 이동하느라 발생하는 듯 한데 나중에 더 해결해 봐야할듯
* 또 스테이지 이동시 스테이지를 표시할 때 새로운 라벨을 생성해서 화면에 표시해 주게 되는데 이때 다소 딜레이가 있다. 딜레이를 해결하기 위해 미리 라벨을 만들어 두려 했으나 볼륨이 큰 게임일 경우 시스템 부하가 많을 것 같다는 생각이 듬
* 이번에 구현에는 스테이지의 정보를 enum을 이용한 이중배열을 이용하여 해당 위치의 무엇이 있는지 표시하도록 하였는데 이 대신 object 형 이중배열을 선언하여 각 노드별로 비어있다면 null, 무언가 있다면 해당하는 오브젝트를 가리키도록 지정해 주는 것이 좀더 구현이 간편하지 않았을까 함