

# C++ 프로그래밍

## 프로젝트

프로젝트 명	<i>Snake Game</i>
팀 명	<i>팀 K</i>
문서 제목	결과보고서

Version	1.1
Date	2021-JUN-20

팀원	20192218 김 수빈
	20163144 이 성주

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

#### CONFIDENTIALITY/SECURITY WARNING

이 문서에 포함되어 있는 정보는 국민대학교 소프트웨어융합대학  
소프트웨어학부 및 소프트웨어학부 개설 교과목 C++프로그래밍 수강 학생 중  
프로젝트 "Snake Game"를 수행하는 팀 "K"의 팀원들의 자산입니다. 국민대학교  
소프트웨어학부 및 팀 "K"의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수  
없습니다.

### 문서 정보 / 수정 내역

<b>Filename</b>	최종보고서-Snake Game.doc
<b>원안작성자</b>	김수빈, 이성주
<b>수정작업자</b>	김수빈, 이성주

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2021-06-18	이성주	1.0	최초 작성	보고서의 전체적인 구성과 초안 작성
2021-06-19	김수빈	1.1	내용 수정	수정된 연구내용 추가

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 목 차

1	개요 .....	4
2	개발 내용 및 결과물 .....	5
	2.1 목표 .....	5
	2.2 개발 내용 및 결과물 .....	9
	2.2.1 개발내용 .....	9
	2.2.2 시스템 구조 및 설계 .....	21
	2.2.3 활용/개발된 기술 .....	34
	2.2.4 현실적 제한 요소 및 그 해결 방안 .....	34
	2.2.5 결과물 목록 .....	35
3	자기평가 .....	35
4	참고 문헌 .....	36
5	부록 .....	37
	5.1 사용자 매뉴얼 .....	37
	5.2 설치 방법 .....	42

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

# 1 개요

## 1.1 프로젝트 개요

2021 년 1 학기 C++프로그래밍 기말 프로젝트로 SnakeGame 을 제작한다. C++프로그래밍 언어로 ncurses 라이브러리를 이용하여 구현되도록 한다.

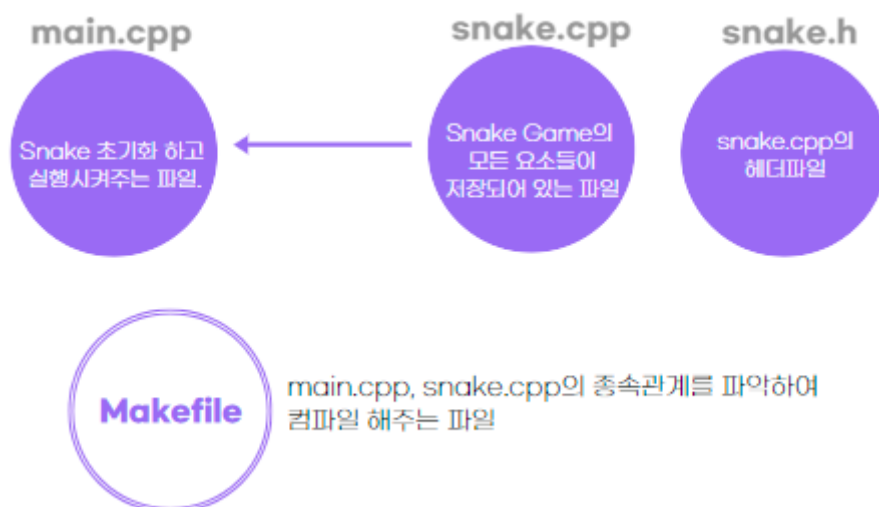
## 1.2 라이브러리 설치방법

Ncurses

```
<Mac>
brew install ncurses

<Ubuntu>
sudo apt-get update
sudo apt-get install libncurses5-dev libncursesw5-dev
```

## 1.3 프로젝트 전체적인 구조



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 2 개발 내용 및 결과물

### 2.1 목표

적용단계	내용	적용 여부
1 단계	Map 의 구현	적용
2 단계	Snake 표현 및 조작	적용
3 단계	Item 요소의 구현	적용
4 단계	Gate 요소의 구현	적용
5 단계	점수 요소의 구현	적용
6 단계	기말프로젝트 목표안에 포함되지 않은 다른 아이디어	적용

#### 2.1.1 프로젝트 목표

요구한 각 단계별 구현 완료 및 추가 단계 구현으로 완성도 있는 Snakegame을 만들고자 한다.

#### 2.1.2 [1단계] Map의 구현

Ncurses 라이브러리 함수들을 사용하여 2 차원 배열로 된 Snake Map 을 Game 화면으로 표시하는 프로그램을 완성한다.

#### 2.1.3 [2단계] Snake 표현 및 조작 구현

[1 단계]의 맵 위에 Snake 을 표시하고, 화살표를 입력 받아 Snake 가 움직이도록 프로그램을 완성한다. Snake 는 다음과 같은 규칙을 따른다.

- 진행방향과 반대방향으로 이동할 수 없다.
- 자신의 Body를 통과할 수 없다.
- Wall를 통과할 수 없다.
- Head 방향의 이동은 일정시간에 의해 이동한다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

#### 2.1.4 [3단계] Item 요소의 구현

[2 단계]의 프로그램에서 Map 위에 Growth 와 Poison Item 을 출현하도록 한다. 다음과 같은 규칙을 따른다.

- Snake가 Growth Item을 획득하면, 몸의 길이가 1 증가한다.
- Snake가 Poison item을 획득하면, 몸의 길이가 1 감소한다.
- 몸의 길이가 3보다 작아지면 게임이 종료된다.
- Item들은 Snake가 있지 않은 임의의 위치에 출현한다.
- Item들은 벽 위에 출현하지 않는다.
- Item들이 출현 후 일정시간이 지나면 사라지고 다른 위치에 나타난다.

#### 2.1.5 [4단계] Gate 요소의 구현

[3 단계]의 프로그램에서 Map 의 Wall 의 임의의 위치에 2 개의 Gate 를 출현시키고, 각 Gate 에서 Snake 가 통과할 수 있도록 한다. Gate 는 다음과 같은 규칙을 따른다.

- Gate는 한 번에 한 쌍(2개)만 나타나고 겹치지 않으며, 벽의 고정된 위치에 나타난다.
- Gate에 Snake가 진입하면 다른 Gate로 진출한다.
- Gate는 진출 방향이 따로 있다. 추후 개발내용에서 자세히 다룬다.
- **Gate는 Snake의 Growth item을 획득 시 출현한다.**

Wall 과 Immune Wall 은 Gate 로 변할 수 있다. Wall 과 Immune Wall 에 다음 규칙을 추가한다.

- 모든 Wall은 Snake가 통과할 수 없다.
- Snake와 충돌 시 게임이 종료되며 시작/ 종료 여부를 선택한다.

#### 2.1.6 [5단계] 점수 요소의 구현

[4 단계] 프로그램에서 우측에 게임 점수를 표시하는 화면을 구성한다. 게임 점수를 표시하는 화면은 다음과 같은 규칙을 따른다.

- 게임 중 몸의 최대길이를 계산하여 표시한다. :  $(\text{Current Length})/(\text{Max Length})$ 로 계산한다.
- 게임 중 획득한 Growth와 Poison Item의 수를 표시한다.
- 게임 중 Gate 사용 횟수를 표시한다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

Mission 의 달성 여부를 표시하는 화면도 구성한다. 위의 게임 점수를 표시하는 화면의 구성과 동일하며, 각 Map 마다 주어진 Mission 을 2 개 달성하면 다음 Map 으로 넘어간다.

게임 종료 시, 해당 try 에서의 최대 길이, 획득한 Growth item, Poison item, Gate 를 지난 횟수를 표시합니다.

### 2.1.7 [6단계] 추가 기능 구현

– 소리, 맵 옵션, 게임 통계 화면, 총 점수와 죽은 이유, 중간 종료기능

[6 단계] 추가 기능은 총 6 가지를 구현한다.

(1) Poison Item 획득 시, 뱃소리가 난다.

(2) 처음 Game 실행 시, 맵 Random mode 와 Sequential mode 중 선택 가능하다.

(3) 해당 stage 를 die 또는 success 중간 게임 화면을 표시한다.

-표시 내용

#### ① 실패 시, 나오는 화면

1) 해당 Try 가 끝난 이유를 출력한다.

- Wrong Direction Key! : 현재 진행 방향과 반대방향 키를 입력할 시 출력한다.
- Colliding with Walls : 벽에 충돌 시 출력한다.
- Because Body is too short (Too many Poison) : 길이가 3 보다 작아지면 출력한다.

2) 해당 Stage를 얼마나 시도했는지, 해당 Try의 최대 길이, 획득한 Growth와 Poison Item의 수 그리고 Gate 사용 횟수를 표시한다.

#### ② 성공 시, 나오는 화면

해당 Stage 를 얼마나 시도했는지, 해당 Try 의 최대 길이, 획득한 Growth 와 Poison Item 의 수 그리고 Gate 사용 횟수를 표시한다.

성공한 Try의 기록은 최종 화면에 출력되는 Total Score에 반영된다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

### ③ 게임 종료 시, 나오는 최종 화면

1) Break 또는 Success 시, 게임을 계속 진행 할 지, 종료할 지 선택하게 된다.

종료를 선택할 시 최종 화면이 출력된다.

2) 또한, 4 라운드를 모두 진행 시 Total Score 를 출력한다.

- i. MapMode
- ii. 진행한 Round 의 갯수
- iii. Try 횟수
- iv. 게임 진행시 Max body 의 길이: Max Body 는 다른 변수와 다르게 실패한 게임에 대해서도 저장한다.
- v. 총 growth item 획득 횟수
- vi. 총 poison item 획득 횟수
- vii. 총 gate 진행 횟수

[출력화면]

```

<Total score>

Mode:                                     [Sequential Mode]

Round:

How many Play:                           16
How much is the longest body:             7
How much of an growth item the snake ate: 4
How much of an poison item the snake ate: 11
How much of an gate the snake passed:    5

```



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 2.2 개발 내용 및 결과물

### 2.2.1 개발 내용

#### 0. 기본 정의

Snake 객체를 화면에 출력할 시, 기본으로 정의하는 것을 설명한다.

##### (1) Color

- ◆ 사용 함수: **void colorInit();**
- ◆ 사용 변수: NONE, EMPTY, WALL, IMMUNE\_WALL, HEAD, BODY, GROWTH, POISON, GATE

Screen 에 표시 될 변수는 위와 같이 총 9 개이다. 각각의 변수들의 색을 **Ncurses 라이브러리** 내장함수 **init\_pair()**을 통해, 정한다.

##### (2) Window

- ◆ 사용 변수: **startWin, mapWin**

**Ncurses 라이브러리** 내장 변수인 **WINDOW \*win**을 통해, 두가지 Window를 선언하고 사용한다.

① **startWin** : 게임의 시작, 중간, 끝에서 사용하는 **Screen** 을 출력하는 Window 이다. 출력되는 내용에 대한 자세한 내용은 해당 함수에서 자세히 설명하도록 한다.

② **mapWin**: 게임 진행 시, 출력되는 **Window** 이다.

마찬가지로, 출력되는 내용에 대한 자세한 내용은 해당 함수에서 자세히 설명하도록 한다.

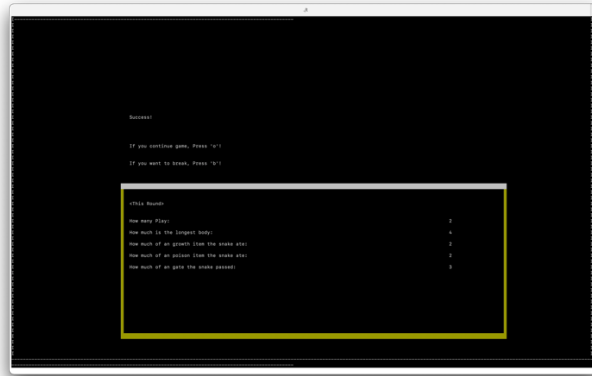
[출력화면] – 자세한 출력 내용도 해당 함수에서 자세히 보인다.

##### i. mapWin



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

ii. startWin



### (3) Default Value

◆ 사용 함수: `setDefault();`

◆ 사용 변수: `private`으로 선언된 클래스 내의 거의 모든 변수

게임 Play 시작 시, 기본적으로 설정되는 값들이 존재한다.

새롭게 게임이 시작될 때마다, `setDefault()` 함수를 호출하여, 변수를 초기화시킨다.

-Default Value

- ① growth item, poison item, gate 는 기본적으로 시작 시에는 존재하지 않으므로 (-1,-1)으로 설정한다.
- ② snake 의 시작 위치는 항상 고정하며, 길이도 3 으로 고정한다.
- ③ **dir**: 진행방향의 시작 시 방향은 오른쪽(d)이다.
- ④ **isPoison, isGrowth** 는 growth item, poison item 의 등장여부를 체크하는 변수이므로 시작 시, false 로 설정한다.
- ⑤ play 의 진행 상태를 저장하는 **score** 배열의 값을 초기화한다.
- ⑥ mission 달성 여부를 판단하는 **mission** 배열은 기본적으로 시작 시, false 로 설정한다.
- ⑦ game 의 진행여부를 판단하는 game 변수는 시작 시, true 로 세팅한다.

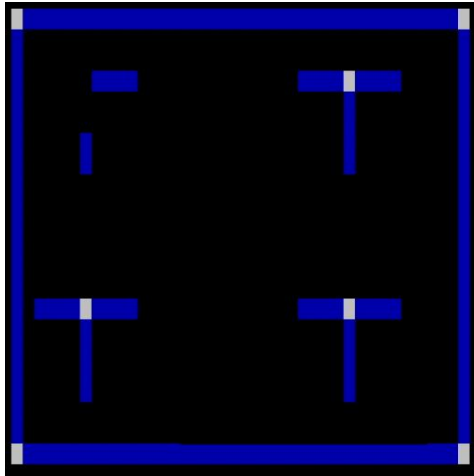
### 1. Map

Map 크기는 40X22 이고, Map 의 개수 총 4 개이다. txt 파일로 Map 의 데이터를 저장했고, 각 txt 파일은 Map 의 모양을 결정하는 2 차원 배열로 이루어져 있다. Map 좌표 값에서 0 은 빈 공간, 1 은 wall, 2 은 immune wall 를 뜻한다. Window

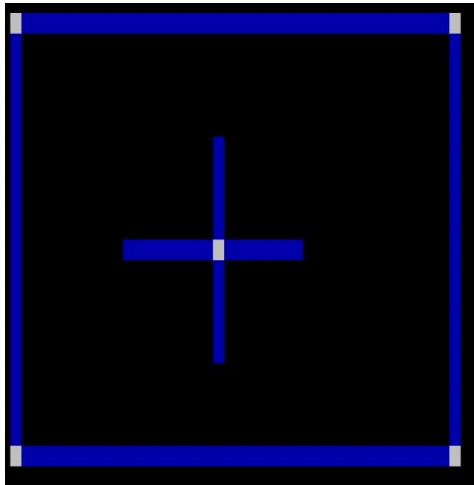


 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

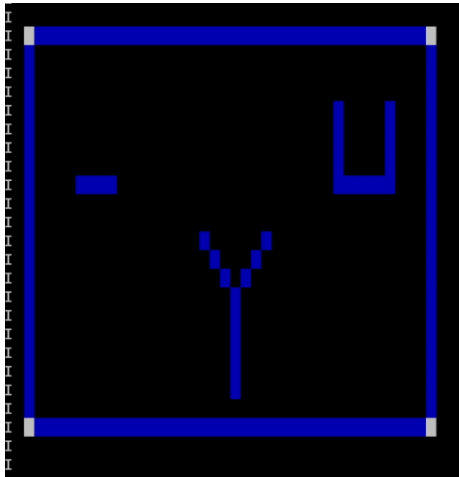
■ Map2



■ Map3



■ Map4



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

Map구현과 관련된 함수는

- void setMap(int count): txt파일로부터 MAP 배열을 채우는 함수
- void createMap(): MAP 배열을 통해 스크린을 구성하는 함수이다.

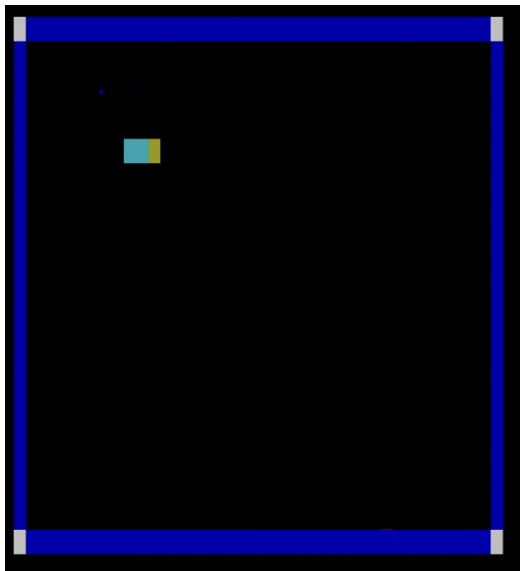
Startm() 함수를 통해 mode 선택 시, 사용자가 Sequential mode를 선택하면

24가지의 경우의 수 중 rand()함수를 통해 랜덤하게 map의 순서를 선택한다. 그 후 setMap()과 createMap()을 통해, 스크린에 Map을 구성한다.

## 2. Snake

Snake 의 Head 색은 노란색, Body 색은 밝은 청록색으로 표현한다. Snake 가 이동할 때 이동 방향 4 개를 구분하여 각각의 경우에 x, y 값을 변경하는 방식으로 구현한다. 이동 방향은 변수 선언을 하여 저장하고 키보드 입력에 따라 바꾸어 준다.

[실행화면]

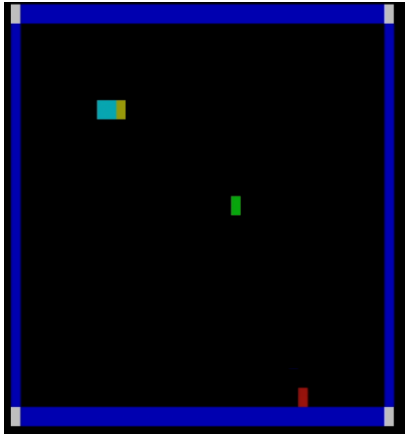


 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

### 3. Item

Growth와 Poison를 랜덤함수로 Snake의 위치를 제외한 무작위 빈 공간에 나타나도록 한다. 사용자가 Growth Item을 먹으면 Snake의 길이를 늘리고, Poison Item을 먹으면 길이를 줄인다. Growth의 색깔은 초록색으로, Poison의 색깔은 빨간색으로 표현한다.

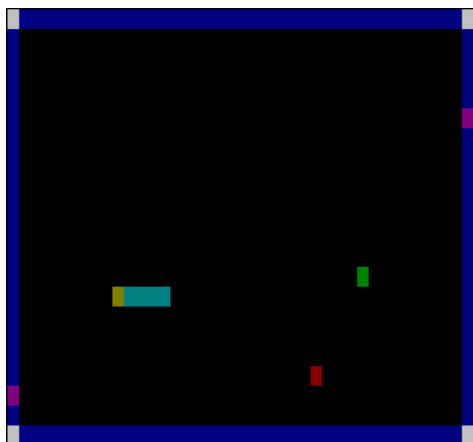
[실행화면]



### 4. Gate

Gate는 조건이 맞으면 두 쌍으로 출현한다. 고정된 값으로 Gate를 출현시킨다. If-else 문으로 Map의 종류에 따라오는 Gate의 위치를 바꾸어 준다. Snake가 Gate를 지날 때 항상 Map의 안쪽 방향으로 진출할 수 있도록 한다. Gate가 있는 wall이 가장자리에 있는 경우 option을 주어서 상단 벽일 때는 아래 방향으로, 하단 벽일 때는 위 방향으로, 좌측 벽일 때는 오른쪽 방향으로, 우측 벽일 때는 왼쪽 방향으로 가도록 한다. 또한, Gate가 있는 wall이 Map의 가운데에 있는 경우에도 option을 주어서 방향 별로의 조건을 만족시키도록 한다.

[실행화면]



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 5. 점수, mode, try

- 관련 함수: `void createScore()`, `void setScore()`, `void createMission()`, `void setMission()`,

`void setStage()`

`void startm()`

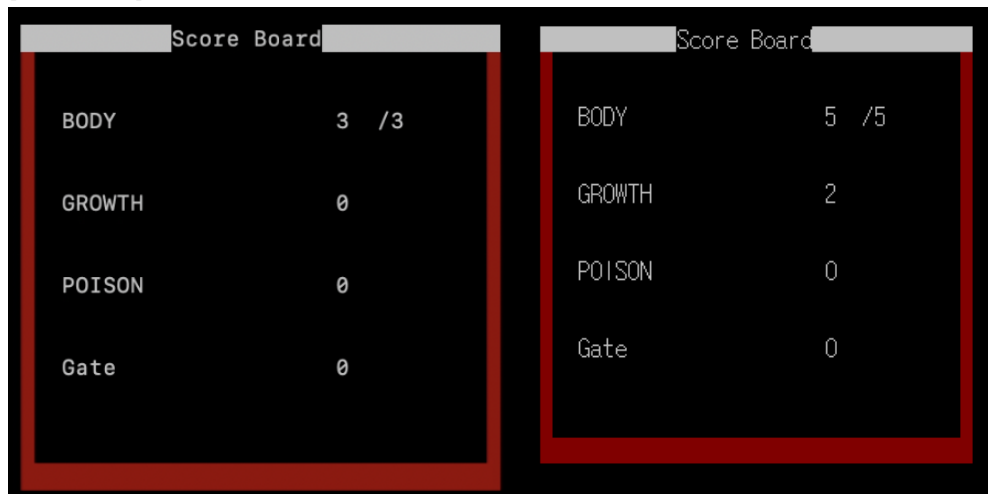
- 관련 변수: `int score[4]`, `bool mission[4]`, `Window *mapwin`

(1) `createScore`, `setScore`: 점수를 표시하는 화면을 위한 함수이다.

`createScore`함수를 통해 Snake의 최대 몸길이, Growth Item의 수, Poison Item의 수, Gate 사용 횟수가 들어있는 배열을 구성하고 `setScore`함수를 이용해 화면에 띄운다.

각 try마다 `setDefault()`함수를 이용하여 초기값으로 초기화된다.

[실행화면]



(2) `createMission`, `setMission`: 각 항목들의 조건에 Mission의 성공 여부가 달려있다.

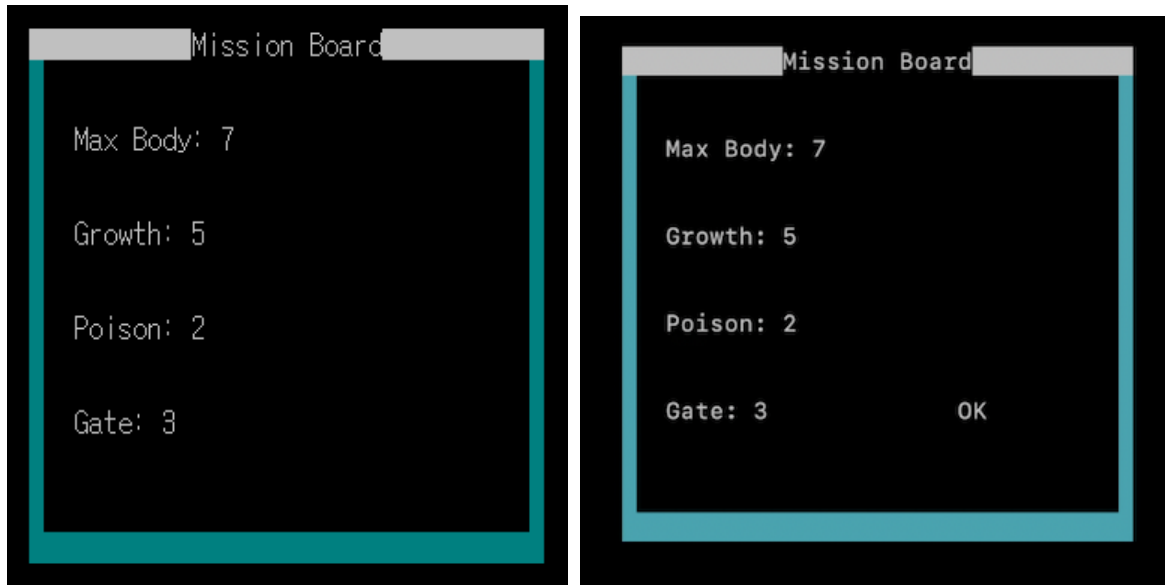
-Mission

- Max Body: Max Body가 7이상 달성 시, true
- Poison item: poison item을 2개 이상 획득 시, true
- Growth item: growth item 5개 이상 획득 시, true
- Gate: gate를 3번 이상 이용 시, true

Mission이 달성된다면, Mission Board에 ok를 출력한다.

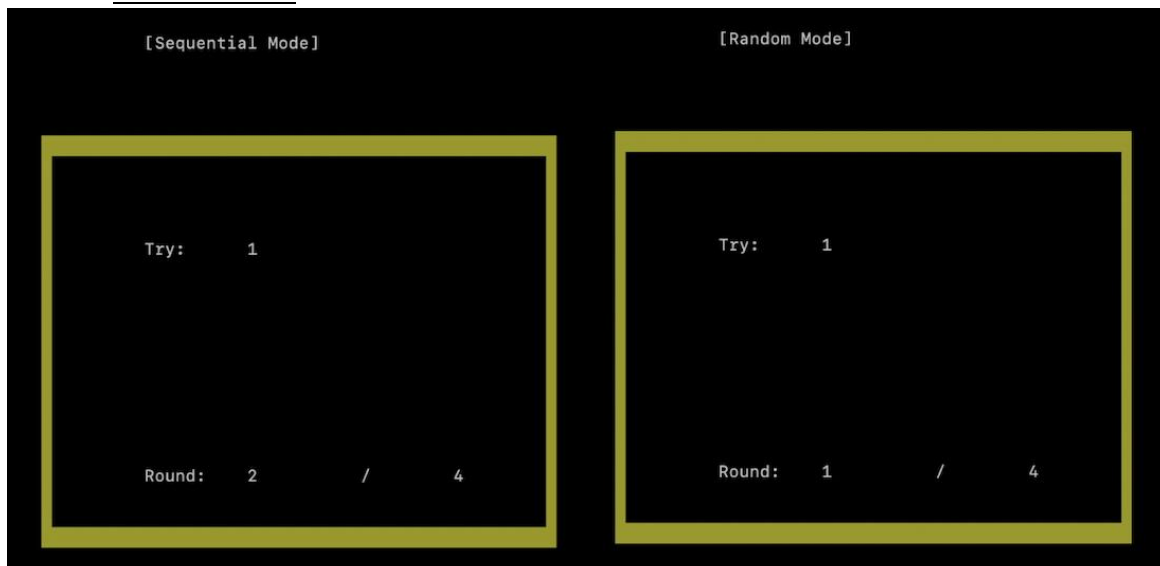
 <div> <b>국민대학교</b>  <b>컴퓨터공학부</b>  <b>C++ 프로그래밍</b> </div>	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

[실행 화면]



(3) try횟수, Mode 출력과 round 순서를 출력한다.

해당 라운드를 몇 번 째 실행하는지, 해당 round가 몇 번째 round인지, mode가 어떤 mode 인지 출력한다. 해당 내용을 출력하기 위해서 score\_save 배열과 count변수를 이용한다.





 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 6. 6단계 추가 구현

(1) 게임 첫 시작 시, 처음 화면에서 **map mode**를 선택

(2) 게임 처음, 중간, 끝 scene 추가

- 관련 함수: **startm()**

- 관련 변수: **mapMode, mapMode\_save, random**

Random여부를 사용자로 부터 입력 받아, 선택되는 형태이다. 이 option을 구현하기 위해서 cin.get()함수를 이용하여 문자 하나만 입력받게 했다. 입력받은 문자에 대해서 random 변수에 bool 형태로 true와 false를 저장한다.

Random일 경우 true, sequential mode일 경우 false이다.

r' 선택 시, Random mode가 되는데 rand()함수를 이용해 0부터 23 중 하나를 고른 뒤, **mapMode** 배열에 **mapMode\_save**에 저장된 map 순서를 저장한다. 따라서 진행되는 모든 맵 순서는 완벽하게 독립적이며, 다 다르다.

**mapMode** 배열을 이용하여, txt파일을 찾아 setmap()함수를 이용하여 MAP배열을 불러온 뒤 scene에 출력한다..

random에 저장 된 bool 값을 가지고 게임 진행 중과 최종 게임 종료 시, 출력되는 문장을 다르게 한다.

Ex) Sequential mode일 경우

1) Map 진행 중

[Sequential Mode]

2) 게임 종료 시, 출력

Mode:

[Sequential Mode]

(3) 게임 진행 중 poison item 획득 시, 뱀 소리가 남:

eatPoison()에 아스키 코드를 이용하는 코드를 사용해서 구현하였다.

-사용 코드: `std::cout << 'Wa' << std::flush;`

(4) 처음, 중간, 끝 화면 생성

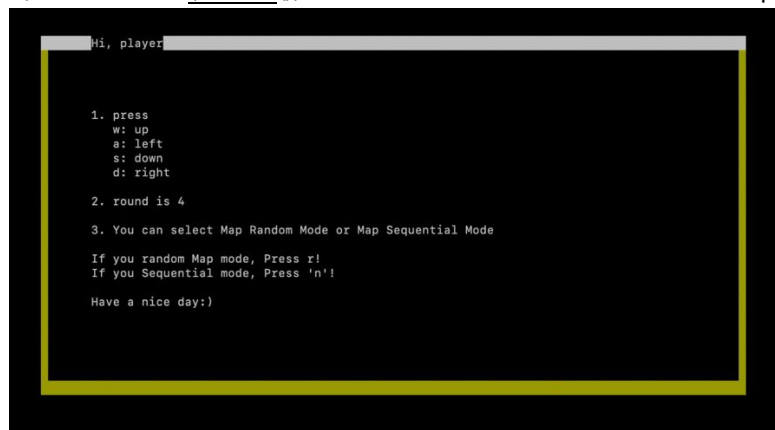
- 관련 함수: **midScene(), successScene(), completeScene()**

- 관련 변수: **WINDOW \*startWin**

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

1) startWin 이라는 새로운 window 를 통해, 각 stage 를 구분해 주는 장면을 띄운다. 게임 시작, 중간, 끝에 출력이 되며 간단한 키를 눌러 해당 화면을 종료하도록 설정하였다.

2) 처음 화면 (startm()): 간단한 게임 rule 설명과 함께 map mode 를 선택한다.



'r' 입력 시, random mode, 'n' 선택시 Sequential mode로 실행된다.

3) 중간 화면 (midScene(), successScene()):

- successScene(): 게임이 성공 했을 때 시행되는 함수이며

게임 try횟수, max body, growth item, poison item, gate 갯수를 출력한다.



'o'를 누르면 다음 단계를 진행할 수 있고, 'b'를 누르면 completeScene() 으로 이동한다.

마지막 4라운드 성공 시, 이 함수를 호출하지 않고 바로 completeScene()으로 이동한다.

- midScene(): 게임이 실패했을 때 시행되는 함수이며,

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

게임 실패 사유, max body, growth item, poison item, gate 갯수를 출력한다.

```

<This Round>
How many Play:                                2
How much is the longest body:                  3
How much of an growth item the snake ate:      0
How much of an poison item the snake ate:      1
How much of an gate the snake passed:         0

```

- 게임 실패 사유는 변수 reason를 이용하여 출력하도록 한다.
  - i. Reason=1 body가 3보다 작아질 경우

```

LOSE!

Reason: Because Body is too short(Too many Poison)

```

- ii. Reason=2 진행방향과 반대 방향으로 진행을 원할 경우

```

LOSE!

Reason: Wrong Direction Key!

```

- iii. Reason=3 벽에 부딪힌 경우

```

LOSE!

Reason: Colliding with Walls

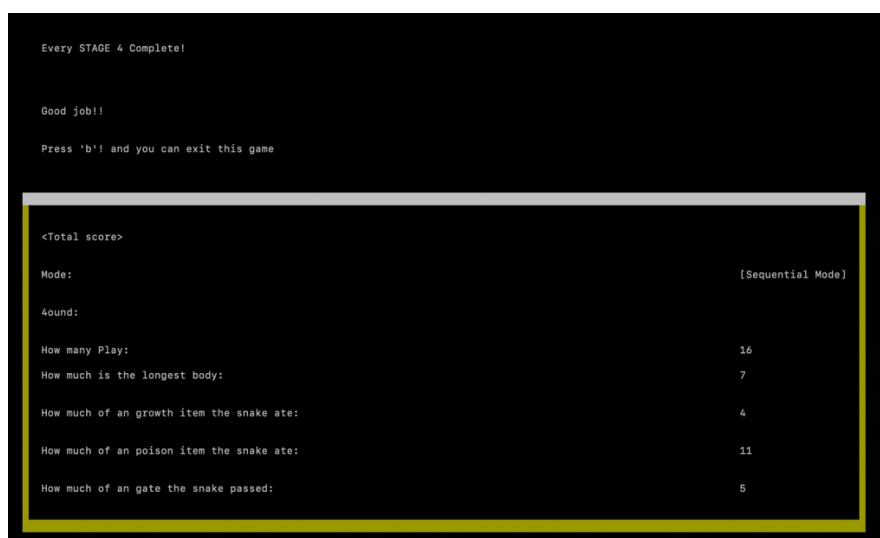
```

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

'o'를 누르면 다시 try 가능하며, 'b'를 누르면 completeScene() 으로 이동한 뒤 게임 종료한다.

4) 마지막 화면 (completeScene()): 게임 종료 시 시행되는 함수이다.

모든 라운드의 try횟수의 총 합, 달성했던 max body, 진행 했던 round의 갯수, 획득 했던 growth item, posion item, 지났던 모든 gate의 횟수를 각각 모두 더한 값을 출력한다.



\* 중간 종료와, 4 라운드 후 종료할 시, 출력되는 문장의 차이를 두었다.

**'b' 입력 시, 게임이 종료된다.**

**(5) round 종료 시, break 또는 continue 선택 가능 (중간 종료 가능)**

모든 \*Scene함수에서 진행 여부를 키로 입력받아, bool continued 변수에 저장하며 play() 함수에서 이 변수를 읽는다. 계속 진행할 경우 true를 저장하고 중단을 원할 경우에는 false를 저장한다. play()함수에서 false를 읽게 된다면 바로 completeScene()으로 이동한다.

**(6) 중간, 끝 화면에서 게임 종료 이유, 점수 판을 보여줌**

모든 \*Scene함수에서 score\_save, total\_num, reason 변수에 점수를 저장하여 scoreScene()함수에서 출력한다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

각 \*Scene함수는 scoreScene 함수를 공동으로 사용하는데 scoreScene함수 실행 시, scene\_option 매개변수를 통해, option을 설정하여, 출력되는 내용을 다르게 만든다.

- scene\_option==1 함수 midScene()에서 호출한 경우
- scene\_option==2 함수 successScene()에서 호출한 경우
- scene\_option==3 함수 completeScene()에서 호출한 경우

[관련 코드] – 실행 화면은 위에 각 scene 함수 설명 시, 출력 된 화면의 일부이다.

```
// print died, max body, growth num, poison, gate
if (scene_option==1) //call by midscene()
{
    mvwprintw(startWin,31,39,"<This Round>");
    mvwprintw(startWin,33,39,"How many Play: ");
    mvwprintw(startWin,35,39,"How much is the longest body: ");
    mvwprintw(startWin,37,39,"How much of an growth item the snake ate: ");
    mvwprintw(startWin,39,39,"How much of an poison item the snake ate: ");
    mvwprintw(startWin,41,39,"How much of an gate the snake passed: ");

    mvwprintw(startWin,33,150,to_string(score_save[count][1]).c_str());
    mvwprintw(startWin,35,150,to_string(score[0]).c_str());
    mvwprintw(startWin,37,150,to_string(score[1]).c_str());
    mvwprintw(startWin,39,150,to_string(score[2]).c_str());
    mvwprintw(startWin,41,150,to_string(score[3]).c_str());
}
```

## 2.2.2 시스템 구조 및 설계도

### 1. Map

-코드 최초 작성: 이성주 (100%)

기본 설정 및, Map 을 scene 에 띄우기

-관련 함수: colorInit()

```
enum BLOCKS{
    NONE,
    EMPTY,
    WALL,
    IMMUNE_WALL,
    HEAD,
    BODY,
    GROWTH,
    POISON,
};

void Snake::colorInit(){
    start_color();
    init_pair(EMPTY, COLOR_BLACK, COLOR_BLACK);
    init_pair(WALL, COLOR_BLUE, COLOR_BLUE);
    init_pair(IMMUNE_WALL, COLOR_WHITE, COLOR_WHITE);
    init_pair(HEAD, COLOR_YELLOW, COLOR_YELLOW);
    init_pair(BODY, COLOR_CYAN, COLOR_CYAN);
    init_pair(GROWTH, COLOR_GREEN, COLOR_GREEN);
    init_pair(POISON, COLOR_RED, COLOR_RED);
    init_pair(GATE,COLOR_MAGENTA,COLOR_MAGENTA);
}
```

프로젝트에서 사용할 요소들을 enum 으로 선언 후, 색을 입히는 colorInit()을 구현한다. colorInit 함수는 Ncurses 의 init\_pair 을 이용하여 각각의 요소에 따른 색들을 입힌다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 2. Snake

- 코드 최초 작성: 이성주 (90%)
- 코드 수정: 김수빈 (moveSnake에 Gate부분을 추가, 반대방향 오류를 수정) (10%)
- 함수 **moveSnake(char key)**를 만들어서 snake의 출현, 이동방향을 구현한다.

```
wattron(mapWin, COLOR_PAIR(BODY));
for(int i = 1; i < len; i++)
    mvwprintw(mapWin, v[i].y, v[i].x, " ");
wattroff(mapWin, COLOR_PAIR(BODY));

wattron(mapWin, COLOR_PAIR(HEAD));
mvwprintw(mapWin, v[0].y, v[0].x, " ");
wattroff(mapWin, COLOR_PAIR(HEAD));
wrefresh(mapWin);
```

Ncurses의 wattron를 이용하여 Head와 Body의 속성을 설정해주고 Ncurses의 mvwprintw를 이용하여 Snake의 위치에서 문자열을 출력할 수 있도록 한다.

```
void Snake::moveSnake(char key) {
    // snake가 움직일 때 빈 부분 처리
    wattron(mapWin, COLOR_PAIR(EMPTY));
    for(int i = 0; i < len; i++)
        mvwprintw(mapWin, v[i].y, v[i].x, " ");
    wattroff(mapWin, COLOR_PAIR(EMPTY));
    wrefresh(mapWin);

    // 아무것도 입력되지 않았을 때는 dir의 처음 방향인 오른쪽으로 감


    if(key != 'n')
        dir = key;

    if(dir == 'w'){ // 위쪽
        if(v[0].x == growth.x && v[0].y - 1 == growth.y)
            eatGrowth();
        else if(v[0].x == poison.x && v[0].y - 1 == poison.y)
            eatPoison();
        else if((v[0].x == gate1.x && v[0].y - 1 == gate1.y) || (v[0].x == gate2.x && v[0].y - 1 == gate2.y))
            Gate(1);
        else {
            for(int i = len; i >= 1; i--){
                v[i].x = v[i-1].x; v[i].y = v[i-1].y;
            }
            v[0].x = v[1].x; v[0].y = v[1].y - 1;
        }
    }
}
```

화살표는 w, a, s, d 키를 이용할 수 있으며 각각 위, 왼, 아래, 오른쪽으로 Snake가 움직일 수 있게 해 준다. 아무 키도 입력되지 않았을 때는 이전 진행방향으로 계속 진행하도록 설정한다. 코드의 일부 인, 위쪽으로 진행할 시, 코드를 첨부한다.

```
if (MAP[v[0].y][v[0].x] == 1){
    game=false;
    success=false;
}
```

Snake가 wall 통과 시 게임이 종료되게 한다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

함수 play()를 만들어서 방향키에 따른 snake 의 이동과 Snake 의 규칙을 만족할 수 있도록 구현한다. Key 는 현재 진행 방향이며, tmp 는 이전 진행 방향이다.

```
if((key=='a'&&tmp=='d')|| (key=='d'&&tmp=='a')|| (key=='w'&&tmp=='s')|| (key=='s'&&tmp=='w')) {
    game=false;
    success=false; }
```

Snake 는 자신의 Body 를 통과할 수 없으며, 진행방향과 반대방향으로 이동하려고 할 때 게임은 종료된다.

### 3. Item (Growth, Poison)

- 코드 최초 작성: 이성주 (100%)
- 코드 수정: 김수빈 (4가지 Map구현에 따른 immune wall 조건을 추가했으며, 결과 출력을 위한 변수 계산 부분을 추가, 6단계를 위한 구현일 뿐, 코드의 로직 수정은 없음) (0%)

Growth Item 과 Poison Item 을 출현시키기 위하여 createGrowth()와 createPoison() 함수를 구현한다. rand 함수를 이용하여 Item 들의 위치를 x, y 에 임의로 설정한다.

```
void Snake::newCreate(){
    int end = time(NULL);
    if(end - Time >= 5) {
        createPoison();
        createGrowth();
    }
}
```

Growth Item 과 Poison Item 이 출현한 후 5 초가 지나면 사라지고 새로운 위치에 생기도록 한다.





```
void Snake::createPoison(){
    int x,y;
    bool continuePlay = true;

    Time = time(NULL);
    mvwprintw(mapWin, poison.y, poison.x, " ");

    while(continuePlay){
        continuePlay = false;

        // wall에 생기는 것 방지
        x = rand()%38 + 1;
        y = rand()%20 + 1;
        if ((MAP[y][x] == 1) || (MAP[y][x] == 2)) {
            continuePlay=true;
            continue;
        }
        // snake과 겹치는지 확인
        for(int i = 0; i < len; i++){
            if(v[i].x == x && v[i].y == y){
                continuePlay = true;
                break;
            }
        }

        // growth 겹치는지 확인
        if(growth.x == x && growth.y == y)
            continuePlay = true;
    }

    poison.x = x; poison.y = y;
    isPoison = true;

    wattron(mapWin, COLOR_PAIR(POISON));
    mvwprintw(mapWin, y, x, " ");
    wattroff(mapWin, COLOR_PAIR(POISON));
    wrefresh(mapWin);
}

void Snake::eatPoison(){
    std::cout<<'a'<<std::flush;
    coordinate* temp = new coordinate[len];
    for(int i = 0; i < len; i++){
        temp[i].x = v[i].x; temp[i].y = v[i].y;
    }

    score[2]=score[2]+1;

    if(len-1 < 3){
        game=false;
        success=false;
        reason=1;
    }
    len--;
    delete[] v;
    v = new coordinate[len];
    v[0].x = poison.x; v[0].y = poison.y;

    for(int i = 1; i < len; i++){
        v[i].x = temp[i-1].x; v[i].y = temp[i-1].y;
    }

    isPoison = false;
    createGate();
}
```

```
void Snake::createGrowth(){
    int x,y;
    bool continuePlay = true;

    Time = time(NULL);
    mvwprintw(mapWin, growth.y, growth.x, " ");

    while(continuePlay){
        continuePlay = false;

        // wall에 생기는 것 방지
        x = rand()%38 + 1;
        y = rand()%20 + 1;
        if ((MAP[y][x] == 1) || (MAP[y][x] == 2)) {
            continuePlay=true;
            continue;
        }
        // snake과 겹치는지 확인
        for(int i = 0; i < len; i++){
            if(v[i].x == x && v[i].y == y){
                continuePlay = true;
                break;
            }
        }

        // poison과 겹치는지 확인
        if(poison.x == x && poison.y == y)
            continuePlay = true;
    }

    growth.x = x; growth.y = y;
    isGrowth = true;

    wattron(mapWin, COLOR_PAIR(GROWTH));
    mvwprintw(mapWin, growth.y, growth.x, " ");
    wattroff(mapWin, COLOR_PAIR(GROWTH));
    wrefresh(mapWin);
}

void Snake::eatGrowth(){
    coordinate* temp = new coordinate[len];
    for(int i = 0; i < len; i++){
        temp[i].x = v[i].x; temp[i].y = v[i].y;
    }

    len++;
    score[1]++;
    if (score[0]<len) score[0]=len;
    delete[] v;
    v = new coordinate[len];

    for(int i = 1; i < len; i++){
        v[i].x = temp[i-1].x; v[i].y = temp[i-1].y;
    }
    v[0].x = growth.x; v[0].y = growth.y;

    isGrowth = false;
    createGate();
}
```

Growth Item 을 먹으면 Snake 의 길이가 1 씩 늘어나고, Poison Item 을 먹으면 Snake 의 길이가 1 씩 줄어드는 함수 eatGrowth(), eatPoison()을 구현한다.

각 함수에 temp 라는 동적배열을 임의로 만들고 늘어나고 줄어든 Snake 의 값을 저장하고 Snake 를 재할당한다. eatPoison() 함수에는 Snake 의 길이가 3 보다 작으면



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

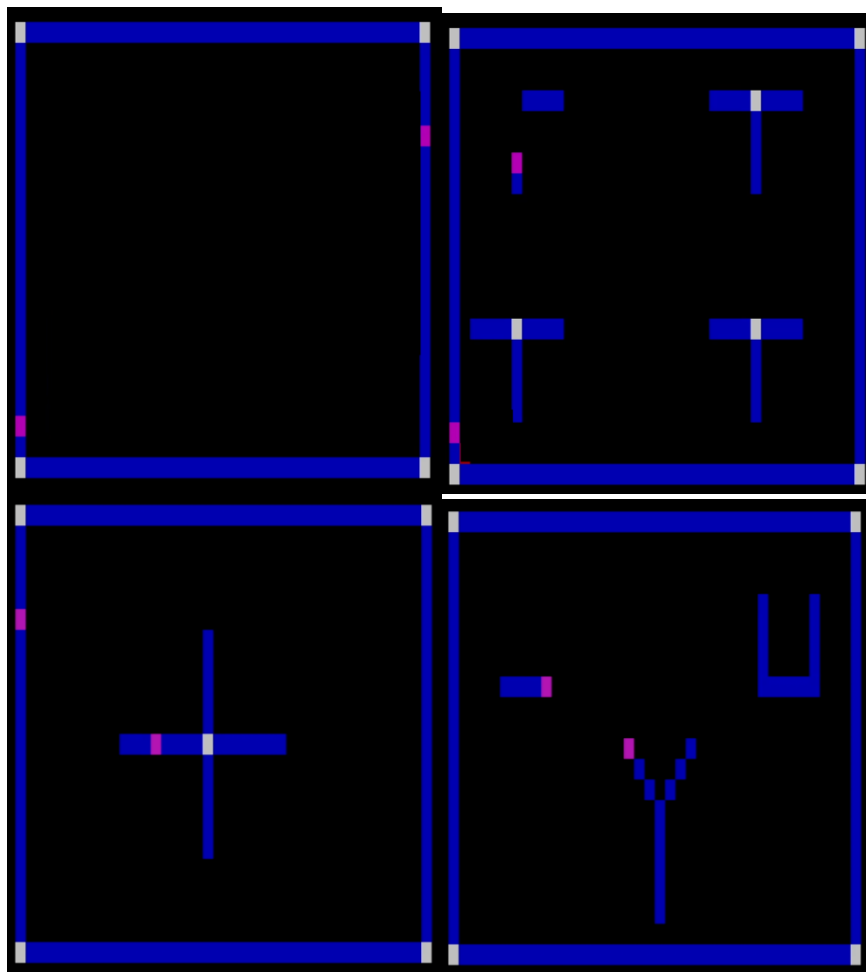
게임이 종료가 되는 코드가 포함되어 있다. 추가적으로 Poison 아이템 출력 시, 백소리가 나는 코드가 구현되어있다.

#### 4. Gate (김수빈) (100%)

- **관련 함수:** createGate( ), Gate(), Passing\_Gate(int option, int gate\_x, int gate\_y)

Gate 는 Snake 가 최초로 growth item 을 획득 했을 시, 나타나도록 했으며 각 map 에 따라 고정된 위치에 나타나도록 구현하였다.

- Gate 생성 함수인 **createGate( )**에 구현 되어있다. (snake.cpp 758 번째 줄 부터 위치)



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

- **Gate()** 함수를 통해 Gate를 이용한 Snake 진행방향 설정을 Passing\_Gate로 전달한다. (snake.cpp 1030번째 줄 부터 위치)

주어진 규칙에 따라, 진행방향, 진행방향의 오른쪽, 진행방향의 왼쪽, 진행방향의 반대방향 순으로 진행 가능하게 하며, 이에 대한 옵션을 passing\_Gate 의 인자로 전달한다.

또한, gate 를 지난 후, 해당 방향으로 이동하기 위해 dir 과 tmp 의 방향을 조정해준다. If-else if 의 형태로 묶고, 내부 코드의 형태는 조건마다 gate 의 좌표 방향만 다르고 구조는 동일하다. 그렇기에 다른 방향의 코드는 생략한다.

#### ■ Option:

- 1: 오른쪽으로 진행해서 gate에 접근한 경우
- 2: 왼쪽으로 진행해서 gate에 접근한 경우
- 3: 위로 진행해서 gate에 접근한 경우
- 4: 아래로 진행해서 gate에 접근한 경우

```
void Snake::Gate(int option){
    //중간에 위치한 gate처리
    int a=v[0].x;
    int b=v[0].y;
    if(option==1) b--;
    else if(option==2) a--;
    else if(option==3) b++; //위에서 아래로 진행
    else if(option==4) a++;

    if(a==gate1.x&&b==gate1.y) {
        if(gate2.x==39){ //option:1 오
            dir='a';
            tmp=dir;
            passing_Gate(2,gate2.x,gate2.y);
        }
        else if(gate2.x==0){ //option:2 왼
            dir='d';
            tmp='d';
            passing_Gate(1,gate2.x,gate2.y);
        }
        else if(gate2.y==0){ //option:3 위
            dir='s';
            tmp=dir;
            passing_Gate(4,gate2.x,gate2.y);
        }
        else if(gate2.y==21){ //option:4 아
            dir='w';
            tmp=dir;
            passing_Gate(3,gate2.x,gate2.y);
        }
    }

    else if(option==1){ //위
        if(MAP[gate2.y-1][gate2.x]==0) {
            dir='w';
            tmp=dir;
            passing_Gate(3,gate2.x,gate2.y);
        }
        else if(MAP[gate2.y][gate2.x+1]==0) {
            dir='d';
            tmp=dir;
            passing_Gate(1,gate2.x,gate2.y);
        }
        else if(MAP[gate2.y][gate2.x-1]==0) {
            dir='a';
            tmp=dir;
            passing_Gate(2,gate2.x,gate2.y);
        }
        else if(MAP[gate2.y+1][gate2.x]==0) {
            dir='s';
            tmp=dir;
            passing_Gate(4,gate2.x,gate2.y);
        }
    }
}
```

- **Passing\_Gate**(int option, int gate\_x, int gate\_y) 함수를 만들어서 각각의 wall 조건에 따른 방향을 설정한다. 이것은 전달 받은 option에 따라 이동한다. (snake.cpp 810번째 줄 부터 위치)



```
void Snake::passing_Gate(int option,int gate_x,int gate_y){
    score[3]+=1;
    if(option==1){ //오른쪽으로 진행
        //printf("a");
        wattron(mapWin, COLOR_PAIR(EMPTY));
        mvwprintw(mapWin, v[len-1].y, v[len-1].x, " ");
        wattroff(mapWin,COLOR_PAIR(EMPTY));

        for(int i = len-1; i >= 1; i--){
            v[i].x = v[i-1].x; v[i].y = v[i-1].y;
        }
        v[0].x=gate_x+1;
        v[0].y=gate_y;
        //printf("(%d, %d)\n", v[0].x, v[0].y);

        wattron(mapWin, COLOR_PAIR(BODY));
        for(int i = 1; i < len; i++)
            mvwprintw(mapWin, v[i].y, v[i].x, " ");
        wattroff(mapWin, COLOR_PAIR(BODY));

        wattron(mapWin, COLOR_PAIR(HEAD));
        mvwprintw(mapWin, v[0].y, v[0].x, " ");
        wattroff(mapWin, COLOR_PAIR(HEAD));
        wrefresh(mapWin);

        for(int k=1;k<len;k++){
            wattron(mapWin, COLOR_PAIR(EMPTY));
            mvwprintw(mapWin, v[len-1].y, v[len-1].x, " ");
            wattroff(mapWin,COLOR_PAIR(EMPTY));

            for(int i = len-1; i >= 1; i--){
                v[i].x = v[i-1].x; v[i].y = v[i-1].y;
            }
            v[0].x=v[0].x+1;
        }
    }
}
```

option 1 은 오른쪽, 2 는 왼쪽, 3 은 위쪽, 4 은 아래쪽이다. If-elseif 의 형태로 묶고, 내부 코드의 형태는 조건마다 gate 의 좌표 방향만 다르고 구조는 비슷하다. 그렇기에 다른 방향의 코드는 생략한다. 위 코드는 Gate 가 가장자리 wall 에 나타났을 때이다.

다음은 Gate 가 map 의 중간에 있는 wall 에 위치했을 때의 코드이다. 함수로는 Gate(int option)을 만들었다. 각각의 경우의 수를 생각하며, if-elseif 의 조건을 걸었다.

## 5. Score (김수빈) (100%)

점수판을 만들기 위해 createScore( )함수를 구현한다. Snake 최대 길이, Growth 와 Poison 을 먹은 횟수, Gate 이용 횟수가 들어있다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

```

void Snake::createScore(){
}
for(int i=0; i<NLINE-5; i++){ //0-35
}
for(int j=NCOL+20; j<NCOL+55; j++){ //50-104
}
if(i==0){
    watttron(mapWin, COLOR_PAIR(IMMUNE_WALL));
    mvwprintw(mapWin, i, j, " ");
    wattroff(mapWin, COLOR_PAIR(IMMUNE_WALL));
}
else if (i==0 || i==NLINE-6 || j==NCOL+20 || j==NCOL+54)
{
    watttron(mapWin, COLOR_PAIR(POISON));
    mvwprintw(mapWin, i, j, " ");
    wattroff(mapWin, COLOR_PAIR(POISON));
}
}
}

mvwprintw(mapWin,0,71,"Score Board");
mvwprintw(mapWin,3,63,"BODY");
mvwprintw(mapWin,6,63,"GROWTH");
mvwprintw(mapWin,9,63,"POISON");
mvwprintw(mapWin,12,63,"Gate");
}
}

```

## 6. 추가 구현 및 코드 수정(김수빈) (100%)

6단계 미션 구현과 전체적인 코드 오류 수정 및 탈고를 진행하였다.

<수정>

### 1) Map 을 4 개 구성, 관련 함수 구현

-관련 함수: setMap(int count)

Count를 매개변수로 받아와 txt파일에 저장된 숫자 배열을 MAP배열에 저장한다.

함수 내에서 좀 더 단일화된 방법으로 txt를 불러 오기 위해, MAPS라는 정적변수를 선언한다.

```
#define MAPS "map"
```

```

int MAP[NLINE][NCOL]={ ... };

void Snake::setMap(int set_count){ //txt파일로 부터 map 배열을 생성

    ifstream in((string)MAPS + to_string(mapMode[set_count]) + ".txt");

    for (int i = 0; i < NLINE; i++) {
        for (int j = 0; j < NCOL; j++) {
            in >> MAP[i][j];
        }
    }
}

```

### 2) 이전 방향 저장을 위한 전역변수 선언 및 관련 코드 추가 (moveSnake())

```
int tmp='d'; //이전 방향을 저장하는 전역 변수
```

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## <6단계>

### (1) 게임 첫 시작 시, 처음 화면에서 map mode 를 선택

- 관련 함수: **startm()**
- 관련 변수: **mapMode, mapMode\_save, random**
  - 1) Random여부를 사용자로 부터 입력 받아, random 변수에 bool 형태로 저장한다.
  - 2) 'r' 선택 시, rand()함수를 이용해 0부터 23 중 하나를 고른 뒤, **mapMode** 배열에 **mapMode\_save**에 저장된 map 순서를 저장한다.
  - 3) **mapMode** 배열을 이용하여, map을 불러온다.
  - 4) 최종 게임 종료 시, random에 저장 된 bool 값을 가지고 출력을 다르게 한다.

### (2) Poison item 획득 시, 뱃소리 나기

- 관련 함수: eatPoison()

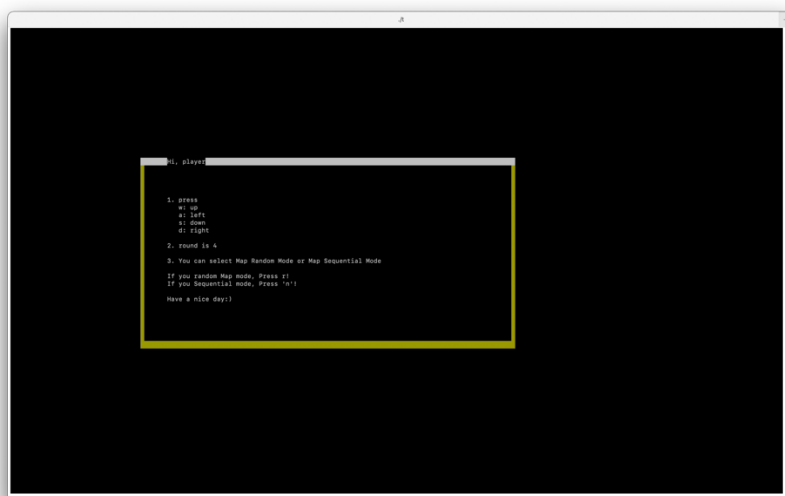
첫 줄에

```
std::cout<<'Wa'<<std::flush;
```

코드를 집어 넣어, 획득 시 소리가 나게 하였습니다.

### ● 처음, 중간, 끝 화면 생성

- 관련 함수: midScene(), successScene(), completeScene()
- 관련 변수: WINDOW \*startWin
  - 1) **startWin** 이라는 새로운 window 를 통해, 각 stage 를 구분해 줄 함수를 생성한다.
  - 2) 처음 화면 (startm()): 간단한 게임 rule 설명과 함께 map mode 를 선택한다.



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

- 'r' 입력 시, random mode, 'n' 선택시 Sequential mode로 실행된다.

Random mode 시, mapMode\_save 에 저장된 경우의 수 중 하나를 골라 mapMode 변수에 저장한다. 이때 rand() 함수를 사용한다.

```
int mapMode_save[24][4]={1,2,3,4},{1,2,4,3},{1,3,2,4},{1,3,4,2},{1,4,2,3},{1,4,3,2},
{2,1,3,4},{2,1,4,3},{2,3,1,4},{2,3,4,1},{2,4,3,1},{2,4,1,3},
{3,1,2,4},{3,1,4,2},{3,2,1,4},{3,2,4,1},{3,4,1,2},{3,4,2,1},
{4,1,2,3},{4,1,3,2},{4,2,3,1},{4,2,1,3},{4,3,1,2},{4,3,2,1}};
int mapMode[4]={0,0,0,0};

while (true){

    c=std::cin.get();
    if (c == 'r'){

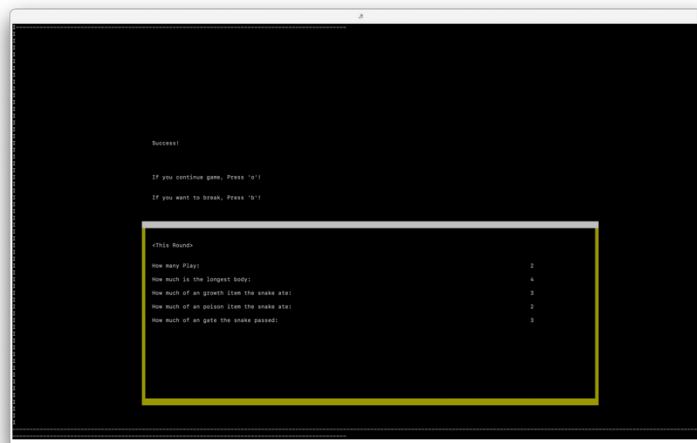
        random=true;
        int pic=rand()%23;
        for(int i=0;i<4;i++) mapMode[i]=mapMode_save[pic][i];

        break;
    }
    else if (c=='n') {
        random=false;
        for(int i=0;i<4;i++) mapMode[i]=mapMode_save[0][i];
        break;
    }
    else continue;
}
```

3) 중간 화면 (midScene(), successScene()):

- successScene(): 게임이 성공 했을 때 시행되는 함수이다.

- 게임 try횟수, max body, growth item, poison item, gate 갯수를 출력한다.



- 'o'를 누르면 다음 단계를 진행할 수 있고, 'b'를 누르면 completeScene() 으로 이동한다.

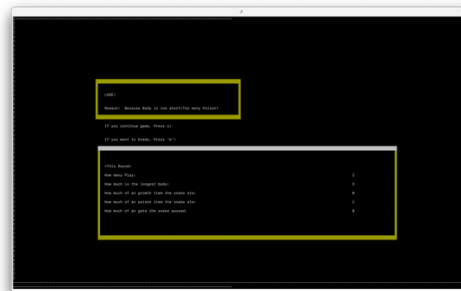
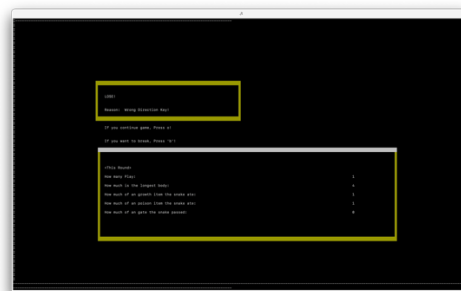
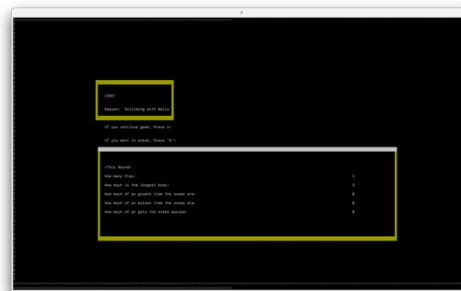
- midScene(): 게임이 실패했을 때 시행되는 함수이다.

게임 실패 사유, max body, growth item, poison item, gate 갯수를 출력한다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

게임 실패 사유는 변수 reason를 이용하여 출력하도록 한다.

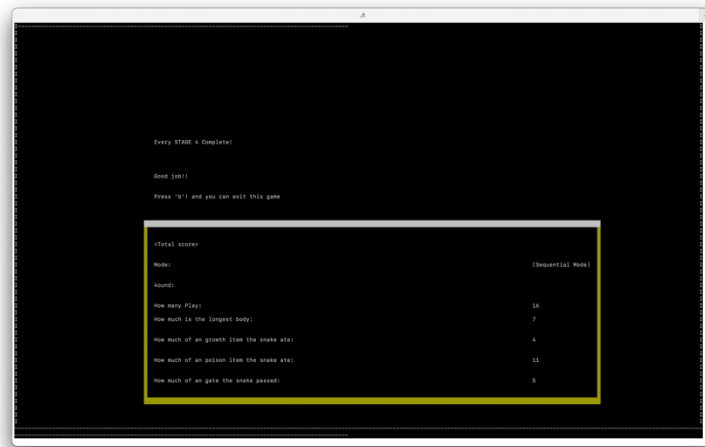
- i. Reason=1 body가 3보다 작아질 경우
- ii. Reason=2 진행방향과 반대 방향으로 진행을 원할 경우
- iii. Reason=3 벽에 부딪힌 경우



'o'를 누르면 다시 try 할 수 있고, 'b'를 누르면 completeScene() 으로 이동한다.

- 5) 마지막 화면 (completeScene()): 게임 종료 시 시행되는 함수이다.  
모든 라운드의 try횟수의 총 합, 달성했던 max body, 진행 했던 round의 갯수, 획득 했던 growth item, posion item, 지났던 모든 gate의 횟수를 각각 모두 더한 값을 출력한다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20



\* 중간 종료와, 4 라운드 후 종료할 시, 출력되는 문장의 차이를 두었다.

**'b' 입력 시, 게임이 종료된다.**

- **round 종료 시, break 또는 continue 선택 가능 (중간 종료 가능)**

- 관련 함수: midScene(), successScene()

- 관련 변수: bool continued

'o'를 입력 받으면 계속 진행하고, 'b'를 받으면 completeScene() 함수로 넘어가며 게임이 종료된다.

이 내용은 bool 변수인 continued에 의해 관리된다.

[관련 코드] -midScene()과 , successScene() 함수내용의 별반 차이가 없다.

```

mvwprintw(startWin,22,39,"If you continue game, Press o!");
mvwprintw(startWin,25,39,"If you want to break, Press 'b'!");
scoreScene(1);

wrefresh(startWin);
char ch=0x00;
while (true){
    std::cin.get(ch);
    if (ch == 'o'){
        continued=true;
        break;
    }
    else if (ch=='b') {
        continued=false;
        break;
    }
    else continue;
}

```



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

-play() 함수 내부의 관련 코드

```

if(game==false) {
    if (success==false) {
        midScene();

        if(continued==false) {
            completeScene();
            exit(1);
        }

        init(count);
        setDefault();
        continue;
    }
    if(count==3){
        completeScene();
        exit(1);
    }
    successScene();

    if(continued==false) {
        completeScene();
        exit(1);
    }
}

```

- 중간, 끝 화면에서 게임 종료 이유, 점수 판을 보여줌 (snake.cpp 408 line)

- 관련 함수: scoreScene(int scene\_option), \*Scene()

- 관련 변수: int score\_save[4][5], int total\_num, int reason

midScene()과 successScene() 그리고 completeScene()에 점수판을 보여주는 용도로 사용한다.

- 1) 각각 함수에서 scoreScene함수 실행 시, scene\_option 매개변수를 통해, option을 설정하여, 출력되는 내용을 다르게 만든다.
  - i. scene\_option==1 함수 midScene()에서 호출한 경우
  - ii. scene\_option==2 함수 successScene()에서 호출한 경우
  - iii. scene\_option==3 함수 completeScene()에서 호출한 경우

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

[관련 코드] – 실행 화면은 위에 각 scene 함수 설명 시, 출력 된 화면의 일부이다.

```
// print died, max body, growth num, poison, gate
if (scene_option==1) //call by midscene()
{
    mvwprintw(startWin,31,39,"<This Round>");
    mvwprintw(startWin,33,39,"How many Play: ");
    mvwprintw(startWin,35,39,"How much is the longest body: ");
    mvwprintw(startWin,37,39,"How much of an growth item the snake ate: ");
    mvwprintw(startWin,39,39,"How much of an poison item the snake ate: ");
    mvwprintw(startWin,41,39,"How much of an gate the snake passed: ");

    mvwprintw(startWin,33,150,to_string(score_save[count][1]).c_str());
    mvwprintw(startWin,35,150,to_string(score[0]).c_str());
    mvwprintw(startWin,37,150,to_string(score[1]).c_str());
    mvwprintw(startWin,39,150,to_string(score[2]).c_str());
    mvwprintw(startWin,41,150,to_string(score[3]).c_str());
}
```

### 2.2.3 활용/개발된 기술

1. <ncurses.h>를 사용하여 게임 로직에 필요한 snake, map, items, gate, score, mission 등의 정보를 출력하고 사용자 인터페이스를 구현하였다.
2. <fstream>를 사용하여 txt 파일의 map의 정보를 불러와 저장했다.
3. map.txt의 "map" 이름을 불러오기 위해서 MAPS로 정의했다.
4. <cstdlib>에 있는 rand( )함수를 사용하여 Item들이 무작위 위치에 생성되도록 했다.
5. String 관련 헤더는 문자열을 입력받거나 출력하기 위하여 선언했다.

### 2.2.4 현실적 제한 요소 및 그 해결 방안

1. [1 단계]에서 Snake의 진행 방향과 반대 방향으로 움직였을 때, 게임이 종료가 되게 만들어야 했지만, 처음에는 진행 방향과 반대 방향으로 가도 계속 움직이는 Snake였다. 단지 Play( )함수에 진행 방향과 반대 방향의 키 조건만 달아주면 가능한 것이었다.
2. 처음 코드를 짤 때 파일들을 나누지 않았다. 코드가 완성된 후 각 함수에 대한 파일들을 나누려고 하다 보니 겹치는 변수들과 중복되는 함수들이 너무 많아 못 나누게 되었다. extern과 static를 사용하여 바꾸려고 했으나, 결국 나누지 못했다.
3. Scene에서 사용자가 mode를 입력할 시에 입력을 잘 받지 못하는 경우가 생겼다. 수업시간 중에 배운 입출력 함수 중 cin.get()함수를 통해 해결하였다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 2.2.5 결과물 목록



## 3 자기평가

이성주 : 1, 2, 3 단계를 맡았다. 처음 만드는 것이고, 첫 시작을 어떻게 시작해야 할지 막막했다. 하지만, 다양한 자료들을 찾아보면서 어떤 식으로 구현하면 좋을지, Ncurses의 사용법 등 여러가지의 정보와 지식을 쌓을 수 있었다. 또한, 코드가 막힐 때 같은 팀인 수빈님의 도움을 받아 잘 해결할 수 있었다. 팀으로 구성되지 않았더라면 혼자 머리를 싸매고 끙끙거리고 있었을 텐데, 2인의 소수 팀프로젝트로 동료 간의 협력과 도움이 잘 나타나는 프로젝트 활동이었다. 처음에 프로젝트를 할 때만 해도 엄청난 스트레스로 다가왔는데, 게임이 완성된 후에 완제품을 봤을 때는 성취감과 뿌듯함이 이루 말 할 수 없었다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

김수빈 : 4, 5, 6 단계와 코드 수정을 맡았다. 성주님이 구현해 놓은 코드를 파악하고 그에 맞춰서 작성하는 것이 조금 시간이 걸렸지만, 전반적인 베이스를 해놓으셔서 ncurses 라이브러리를 이해하기 좀 더 쉬워졌다. 처음 팀 프로젝트를 겪었고, 비대면 상황인지라 프로젝트 진행이 성공적으로 이뤄질 지 가장 걱정하였다. 또한, 각자 코드를 구현하고 합쳐야 하기 때문에 그 과정에서 코드 구현 방식의 차이, 이해도 등을 어떻게 해결해야하나 걱정이었는데, 다행히 별다른 어려움 없이 마칠 수 있었던 것 같다. 생각보다 디자인 측면에서 좀 아쉬움이 많이 남고, 그동안 여러 플래시게임을 해보며 디자인이 얼마나 중요한 지 못 느꼈는데, 이번 구현을 하며 게임의 디자인에 대한 부족한 점이 크게 다가왔다. 해당 내용에 대해서 만일 도움될 자료를 찾았다면 좋았겠지만 찾지 못하여서 간단한 제공 함수를 통해 하이라이팅만 하고 마치게 되어 크게 아쉬움이 남는다. 하지만 여러가지 기능을 구현해보고 특히, 6 단계 구현을 할 때 여러 아이디어를 실행하며, 한계도 느끼고 뿌듯함도 느꼈다. 프로젝트를 진행이 개발자로서 크게 성장할 수 있는 좋은 기회가 될 수 있다고 익히 들었는데 그것을 느낄 수 있는 성과였다고 생각이 든다.

## 4 참고 문헌

번호	종류	제목	출처	발행년도	기타
1	도움자료	NCURSES.pdf	<a href="https://ecampus.kookmin.ac.kr/local/ubdoc/?id=755885&amp;tp=m&amp;pg=ubfile">https://ecampus.kookmin.ac.kr/local/ubdoc/?id=755885&amp;tp=m&amp;pg=ubfile</a>		
2	웹 페이지	[Unix C]NCURSES 란?	<a href="https://widian.tistory.com/58">https://widian.tistory.com/58</a>	2009.10.05	
3	웹 페이지	[Make 튜토리얼] Makefile 예제와 작성 방법 및 기본 패턴	<a href="https://www.tuwlab.com/ece/27193">https://www.tuwlab.com/ece/27193</a>	2017.02.05	

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

4	웹 페이지	Linux man page	<a href="https://linux.die.net/man/3/wbkgd">https://linux.die.net/man/3/wbkgd</a>	Wbkgd 명령어 말고도 다른 ncurses 와 관련된 명령어를 다수 참고 함.
---	-------	----------------	---	--

## 5 부록

### 5.1 사용자 매뉴얼

#### 1. 주의사항

- (1) 화면의 크기를 전체 화면으로 실행해 주시기 바랍니다.
- (2) 코드에 주석 처리 되어있는 echo checking 을 할 수 있는 부분이 있습니다.
- (3) 전체 화면이 아닐 시 프로그램이 제대로 작동하지 않을 수 있습니다.
- (4) 프로그램을 작동하는 컴퓨터에 Ncurses 라이브러리가 설치되어 있어야 합니다.
- (5) Gate 진행 시, passing\_Gate()를 진행하는 데에 진행 컴퓨터 환경에 따라 시간이 걸릴 수 있으니 천천히 여유를 가지고 방향키를 조정해주시길 바랍니다.
- (6) Mac OS 에서 실행 시, Ncurses 32bit 에 대해 의도치 않은 오류가 발생할 수 있습니다. 가끔 발생하는 오류이니, 종료 후 재 컴파일 후 실행해주시면 정상적으로 작동함을 볼 수 있습니다.
- (7) Mac OS 에서 컴파일 시, 첨부된 Makefile 컴파일에 대한 접근 거부 오류가 발생할 수 있습니다. 발생 시, 다음과 같은 소스코드로 실행 부탁드립니다.

```
g++ main.cpp snake.cpp -o snake -lncurses
./snake
```

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 2. < 게임실행과정 >

(1) 게임을 실행하게 되면 첫 화면으로 다음과 같은 화면이 뜬다.



1) Snake 는 간단한 게임 설명과 함께

map 을 순차적으로 진행 할 지, 랜덤하게 진행 할 지에 대한 물음이 뜬다.

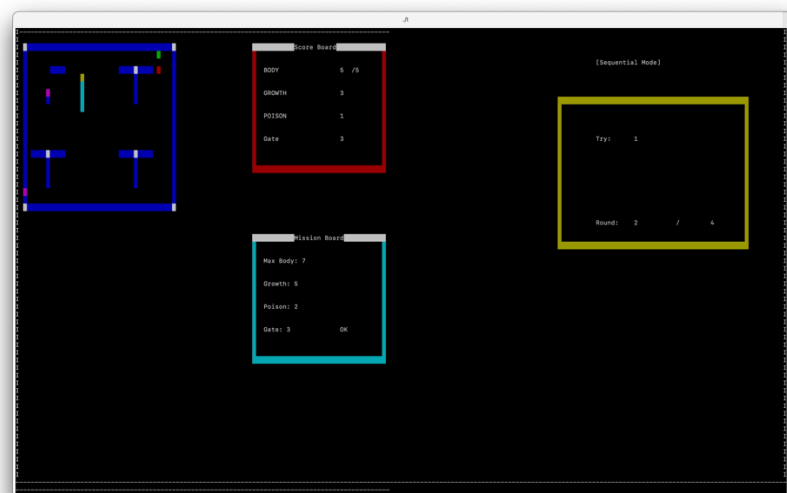
2) 'r'을 누르면 4 개의 map 이 무작위로 뜨고, 'n'을 누르면 단계적으로 map 이 진행된다.

(2) 다음은 SnakeGame 의 화면이다. 오른쪽 파란색 테두리가 게임을 실행하는 화면이고, 빨간색 테두리는 점수판, 하늘색 테두리는 미션 성공 여부판이다. 왼쪽에 있는 노란색 테두리는 총 4 판의 기회를 얻을 수 있는 것을 표시하고 랜덤 map 모드인지, 순차 map 모드인지 알 수 있는 창이다.

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20



- 1) 처음 Snake 의 길이는 3 부터 시작된다. Growth(초록색) item 획득 시 몸의 길이가 1 씩 늘어나고, Poison(빨간색) item 획득 시, 몸의 길이가 1 씩 줄어든다. Wall 에 닿으면 실패를 알리는 화면이 뜬다.
- 2) Snake 이 growth item 을 최초로 획득하게 되면 Gate(핑크색)이 뜨는데, 한 Gate 를 지나면 다른 Gate 로 나오게 된다.

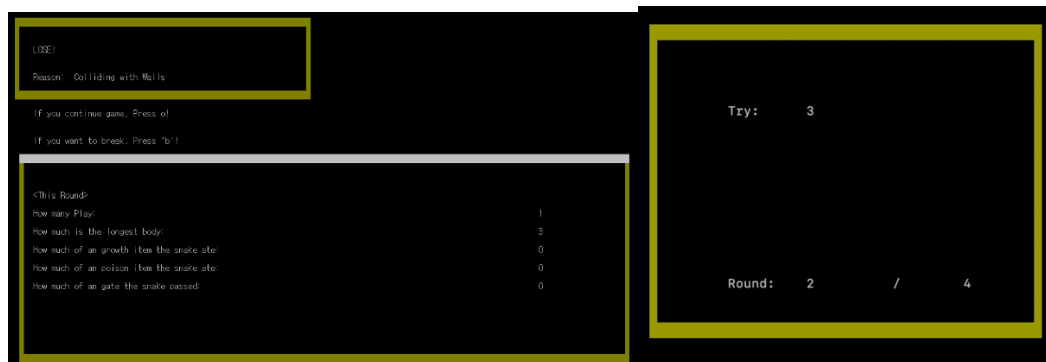


 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

(3) 게임을 진행하다가 게임의 규칙에 위배될 시, 게임 실패 창이 발생한다. 최종적으로 획득한 점수 판의 화면이 등장하며, 재시도 여부를 물어본다.

- 1) 다시 시도 하려면 'o' 이대로 종료하고 싶으면 'b' 를 입력한다. 만일, 'o' 입력 시, 해당 stage 를 다시 실행 가능하며, Try 횟수는 1 증가하게 된다. 이는 map 화면 오른쪽에서 확인할 수 있다.

#### [실행 화면]



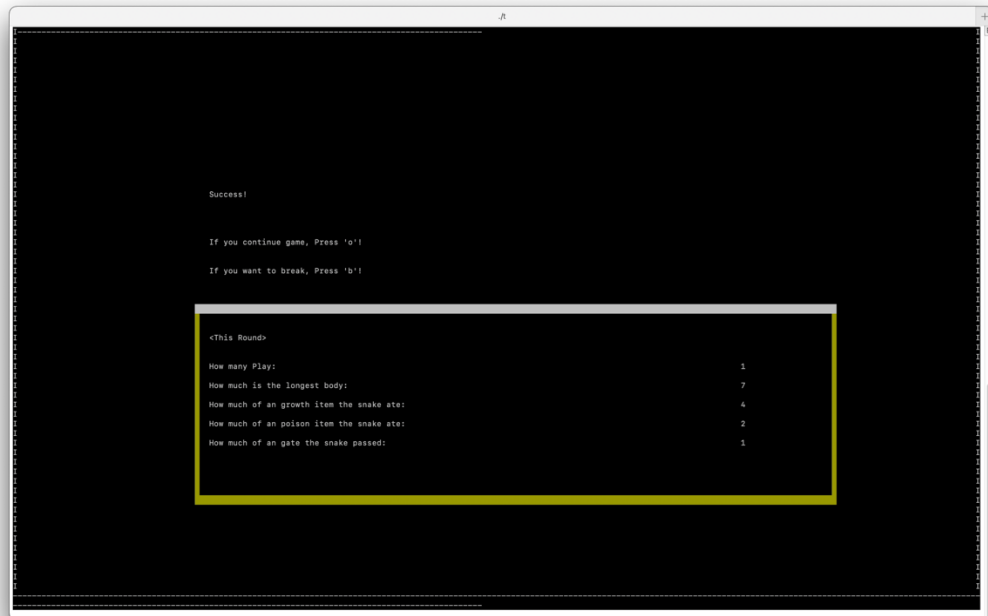
(4) Misson 을 두개 이상 달성 시, 게임 성공 시 창이 발생한다. 4 라운드를 제외하고 최종적으로 획득한 점수 판의 화면이 등장하며, 재시도 여부를 물어본다.

- 1) 다음 stage 하려면 'o', 이대로 종료하고 싶으면 'b' 를 입력한다. 만일, 'o' 입력 시, 다음 stage 를 실행 가능하며, 모든 set 은 원상태로 초기화된다. 해당 라운드가 4 라운드 이거나, 'b' 입력 시, 최종 화면을 출력한다.



 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

### [실행 화면]



- 2) 최종 화면은 위 completeScene(); 함수 설명에서 첨부한 총 7 가지의 내용이 등장하며 'b' 입력 시, 게임이 종료된다.

### [실행 화면]



\*최종화면은 중간에 종료할 때와 4 라운드를 모두 마치고 종료 할 때의 출력되는 출력문 차이를 두었습니다. (그림에서 확인가능)

 <b>국민대학교</b> <b>컴퓨터공학부</b> <b>C++ 프로그래밍</b>	<b>결과보고서</b>		
	<b>프로젝트 명</b>	Snake Game	
	<b>팀 명</b>	팀 K	
	Confidential Restricted	Version 1.1	2021-JUN-20

## 5.2 설치 방법

### 1. MakeFile 이용

- (1) 한 폴더에 모든 파일을 저장한다. (snake.cpp, snake.h, map[i].txt(1<=i<=4), main.cpp, Makefile)
- (2) 폴더 경로로 터미널 창을 연다.
- (3) 터미널 창을 열었으면, make 를 입력한다. 그러면 다음 그림과 같이 main 과 snake 의 오브젝트 파일이 컴파일 되고, "K"가 링킹 되며, 빌드가 완료된다.

```
seongju@DESKTOP-LTUBTPI:/mnt/c/Users/kmusw/Desktop/real$ make
Compiling      main.o
Compiling      snake.o
Linking K
Build complete.
```

- (4) 링킹 된 K 를 ./K 를 친 후, 엔터를 누르면 팀 K 의 Snake Game 이 실행된다.

```
seongju@DESKTOP-LTUBTPI:/mnt/c/Users/kmusw/Desktop/real$ ./K
```

### 2. G++ 컴파일 방법 이용 (Mac OS 에서 권장)

- (1) g++ main.cpp snake.cpp -o \${filename} -lncurses
- (2) ./\${filename}
  - **real** g++ main.cpp snake.cpp -o snake -lncurses
  - **real** ./snake

