



C++프로그래밍 프로젝트

프로젝트 명	Snake Game
팀 명	C++ 2분반 5조 (스테인크)
문서 제목	결과보고서


Version	1.3
Date	2020.06.19

팀	이연주 (팀장) _20191644
	심혜린 _20191620
	김예리 _20191565

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 “스테인크”의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 “스테인크”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

문서 정보 / 수정 내역

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19


Filename	프로젝트 수행 결과 보고서_C++ 2분반 5조
원안작성자	김예리
수정작업자	이연주, 김예리, 심혜린

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2020-06-16	김예리, 심혜린, 이연주	1.0	최초 작성	전체적인 틀 작성
2020-06-18	김예리, 이연주	1.1	수정 작성	추가 작성 및 내용 첨삭
2020-06-19	심혜린	1.2	수정 작성	추가 작성 및 내용 첨삭
2020-06-19	이연주	1.3	수정 작성	추가 작성 및 내용 첨삭

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

목 차

<u>1개요</u>	4
<u>2개발 내용 및 결과물</u>	5
<u>2.1목표</u>	5
<u>2.2개발 내용 및 결과물</u>	6
<u>2.2.1개발 내용</u>	6
<u>2.2.2시스템 구조 및 설계도</u>	6
<u>2.2.3활용/개발된 기술</u>	6
<u>2.2.4현실적 제한 요소 및 그 해결 방안</u>	6
<u>2.2.5결과물 목록</u>	7
<u>3자기평가</u>	8
<u>4참고 문헌</u>	8
<u>5부록</u>	8
<u>5.1사용자 매뉴얼</u>	8
<u>5.2설치 방법</u>	8

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀명	스테이크	
	Confidential	Version 1.3	2020-JUN-19

1 개요

평가기준 (10점)

프로젝트를 완성하기 위해 사용한 개발 방법을 기술하세요.

또한 사용하고 있는 외부 라이브러리와 해당 라이브러리를 획득/설치하는 방법을 기술하세요.

Snake Game은 방향키를 조작하여 Snake를 움직이며 지정된 미션을 수행하는 게임이다. 프로젝트에 사용된 언어는 C++이며, 코드는 크게 Map, Snake, Item, Gate, Score로 구성되어 있다. 코드의 구현 과정에서 다양한 라이브러리를 사용하였다.

Map은 배열을 사용하여 구현했으며, Gate의 요소에 따라 고유 숫자를 부여 Map에 요소를 표현하였다. GUI 프로그래밍은 Ncurses 라이브러리를 사용하였으며 Ncurses 라이브러리의 설치 방법은 다음과 같다.

```
$ sudo apt-get update
```

```
$ sudo apt-get install libncurses5-dev libncursesw5-dev
```

Ncurses 컴파일 방법은 다음과 같다.

```
$ g++ -o snake snake.cpp -lncurses2 -std=c++11
```

실제 게임은 Make 파일을 사용하여 컴파일이 가능하며, 커맨드 창에 ./snake를 입력하면 게임 실행이 가능하다.


Snake는 키보드의 입력에 따라 움직이며 Snake의 방향키 입력 여부를 확인하기 위해 Kbhut 오픈 소스를 사용하였다. 키보드의 입력이 들어올 때까지 기다리게 하기 위해 Getch 오픈 소스를 사용하였고, 들어온 입력 값에 따라 해당하는 key 값으로 변환해주는 역할을 하였다. Snake는 입력한 key의 방향에 따라 움직이며 게임의 미션을 수행하게 된다.

Item은 임의의 시간 이후에 임의의 개수의 Item(Growth, Poison)이 Map에 출력되며 5초가 지나면 Map위에서 사라진다. Snake가 Growth Item을 얻으면 길이가 늘어나고, Poison을 얻으면 줄어든다. Gate는 Map의 벽 부분에 2개 생성되며, Snake가 2개의 Gate 중 하나의 Gate에 들어가면 남은 Gate로 나온다. 게임의 요소(Item, Gate)는 임의의 시간이 측정되면 생성되고 사라진다. 시간 측정은 ctime 라이브러리의 time 함수를 사용하고, 종료시간을 저장하기 위해 Deque을 Container로 사용했다. 난수 생성은 Random을 사용하였다. random은 임의의 시간 측정 뿐만 아니라 요소가 생성될 좌표값 설정 및 요소의 개수도 관여한다. Item의 좌표를 저장하기 위해 Queue를 Container로 사용하였으며, 일정 시간이 지나면 저장한 좌표 위에 있는 아이템을 삭제하도록 코드를 구현했다. Gate의 좌표 설정은 먼저 vector(STL)에 pair을 이용해 x, y쌍의 좌표를 담은 후 Algorithm 라이브러리의 Shuffle을 사용하여 vector 안의 데이터의 위치를 랜덤하게 섞어 첫번째, 두번째 vector 값을 참조하는 방식을 사용했다. 이때 Shuffle의 3번째 매개변수로 random엔진의 생성자로 만든 임시객체를 넘겨준다. 라이브러리 및 외부 기술의 사용 방식은 다음과 같다.

```
#include <ncurses.h> // Ncurses
```

```
#include <termios.h>
```

```
#include <unistd.h>
```

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

```
#include <fcntl.h> //2~4 : kbhit
#include <random> //random
#include <ctime> //time
#include <queue>
#include <algorithm> //shuffle
#include <climits> //int의 max값을 가져오기 위한 lib
```

Score의 구현은 기존에 있던 Map 옆에 ScoreBoard와 Mission 창을 만들어 게임의 진행에 필요한 미션을 생성한다. 윈도우 생성은 역시 Ncurses 라이브러리를 이용하였다. 이때 미션의 값은 random을 사용하여 스테이지마다 임의의 미션을 수행하도록 하였다. 미션을 모두 수행하면 다음 스테이지로 넘어간다.

2 개발 내용 및 결과물

2.1 목표

작성요령 (10점)

프로젝트의 목표를 기술하세요. 각 단계별 목표를 구체적으로 쓰세요.


적용단계	내용	적용 여부
1단계	Map의 구현	적용
2단계	Snake 표현 및 조작	적용
3단계	Item 요소의 구현	적용
4단계	Gate 요소의 구현	적용
5단계	점수 요소의 구현	적용

1단계 (Map의 구현) :

- 배열을 이용하여 Map을 나타내려고 하였다.
- 벽, 모서리, Snake, Item, Gate를 각각 숫자로 표현하려고 하였다.
- 숫자로 표현한 벽, 모서리, Snake, Item, Gate를 Ncurses의 paint 로 각각 다른 색으로 표현하고자 하였다.
- while문을 반복하면서 계속 Map을 다시 그려주려고 하였다.

2단계 (Snake 표현 및 조작) :

- Game을 처음 시작할 때에 Snake의 위치를 초기화 하여 준다.
- 구조체를 통해 Snake의 좌표와 다음에 움직일 방향을 정해준다.

 <div> <p>국민대학교 소프트웨어학부 C++ Project</p> </div>	결과보고서		
	프로젝트명	Snake Game	
	팀명	스테이크	
	Confidential	Version 1.3	2020-JUN-19

- 키보드 값을 입력 받는데, 입력이 들어오는 경우에는 해당 값을 다음좌표로 설정하여 주고, 입력이 들어오지 않는 경우에는 이전의 다음좌표를 유지하여 주도록 한다.
- Snake가 움직이게 하기 위해서, 머리부분(Head:3)은 다음 좌표를 이용하여 좌표를 찍어주고, 꼬리부분(Tail:4)은 그 이전의 Snake의 좌표들을 받아서 좌표를 찍어준다.
- Snake가 벽에 닿거나, 자신의 몸에 닿거나, 꼬리가 일정한 길이보다 짧거나, 진행방향이 반대인 경우에는 GameOver시키는 에러처리를 해주도록 한다.

3단계 (Item 요소의 구현) :

- 랜덤한 시간이 경과한 이후에 아이템이 하나씩 나타나게 한다.
- 아이템의 출현시간 뿐 아니라 아이템의 옵션(Poison or Growth)또한 랜덤하게 구현한다.
- 랜덤한 좌표를 구하여 Map을 설정하여 준다.

위와 같은 조건을 성립시키기 위해서는 **랜덤한 시드**로 랜덤을 구현하여야 한다.
seed값에 **time()**를 주어도 되지만, 서치결과 random 엔진을 사용하는 함수를 통해 랜덤을 구현하도록 하였다.

- 1) Main함수에서 아이템의 갯수가 Map에 3개이하인지 확인하도록 한다
- 2) 아이템 생성이후 5초가 경과하였는지 확인하게 한다.
- 3) 생성 후 5초가 경과하였다면 스스로 사라지게 한다.

- Item을 얻게 되었을 경우에는 각각의 함수(Growth or Poison)를 호출하여준다.
- Growth : 꼬리의 길이를 1 증가시켜주고, 그 이전의 꼬리의 좌표를 통해 방향값에 따라 마지막 추가된 꼬리의 좌표 또한 설정한다.
- Poison : 꼬리의 길이를 1 감소시켜주고, Map에서 해당 아이템이 있던 좌표를 0으로 설정하여 사라지게 설정한다.
- item을 얻지 못하게 된 경우에는, 처음 item이 생성되었을 때의 시간과 비교하여 5초가 경과하였는지 확인한다.
- 5초가 경과하였다면, item의 좌표를 받아와 그 아이템의 좌표를 0으로 만들어 주도록 한다.

4단계 (Gate 요소의 구현):


- Map의 벽에 해당하는 위치의 좌표 값을 받아서 랜덤한 위치에 한 쌍의 Gate를 나타나게 한다.
- 하나의 Gate에 Snake가 진입하면 다른 Gate로 진출한다.
- 게임의 다른 요소와 구별하기 위해 Gate는 Gate의 고유 색상(Magenta)으로 Map 위에 구현한다.
- 가장자리에 있는 Gate로 진출할 경우 Map의 안쪽 방향으로 고정한다. 즉,

상단 Gate일 경우 Snake는 아래쪽방향으로 진행
좌측 Gate일 경우 Snake는 오른쪽 방향으로 진행
하단 Gate일 경우 Snake는 위쪽 방향으로 진행
우측 Gate일 경우 Snake는 왼쪽 방향으로 진행

을 원칙으로 한다.

- 가장자리가 아닌 위치의 Gate로 진출할 경우 진입한 Gate의 진행방향 그대로 진출한다.
- 진행방향대로 진출하지 못하는 경우일 때(즉, Snake의 진행방향으로 진출할 경우 벽에 닿아 GameOver가 되는 상황일 때) 아래의 방향으로 진출한다.

진행방향이 위쪽일 경우 Snake는 오른쪽 방향으로 진행
진행방향이 왼쪽일 경우 Snake는 위쪽 방향으로 진행
진행방향이 아래쪽일 경우 Snake는 왼쪽 방향으로 진행
진행방향이 오른쪽일 경우 Snake는 아래쪽 방향으로 진행

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트명	Snake Game	
	팀명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

- 게임 시작 후 일정한 시간이 지나면 한 쌍의 게이트를 출현하게 하고, Snake가 Gate를 통과하기 전까지 Gate가 사라지지 않도록 고정해준다.
- Snake가 Gate를 통과하면 Map에 Gate를 없애고 다시 벽으로 만들어준다.
- Snake가 Gate를 통과한 이후 다시 일정한 시간이 흐르면 새로운 Gate를 출현하게 한다.

5단계 (점수 요소의 구현):

- 기존에 있던 Map 옆에 ScoreBoard와 Mission창을 띄우도록 한다.
- stage마다 다른 기준을 랜덤으로 설정하여 주도록 한다.
- 아이템을 얻거나, 꼬리가 증가하거나, 게이트를 통과하는 경우마다 count를 세도록 한다.
(count는 클래스 멤버변수로 지정하여 준다.)
- 모든 미션의 조건이 만족하는 경우를 처리하여 Stage를 각각 맞추어 넘어가도록 설정한다.

2.2 개발 내용 및 결과물

2.2.1 개발 내용

작성요령 (10점)

프로젝트의 수행의 내용을 구체적으로 기술한다. 세부 목표별로 어떤 결과를 어떤 방법으로 달성하였는지를 자세히 기술한다.

1. Map의 구현

Snake Game을 실행하기 위한 map을 ncurses를 이용해 터미널에 map의 틀을 만들어 유니코드와 Attribute를 설정하여 Immune Wall과 Wall을 구성하였다. map을 만드는 Map() 함수 안에서 Snake, Item, Gate의 특징을 설정하였고 map의 모든 벽은(Wall: 1)로, Snake의 머리(Head: 3), 꼬리(Tail: 4), Growth Item(5), Poison Item(6), 게이트(Gate: 7)로 표현하였다. 이 때, map안에서 Snake의 좌표 값에 따라 다르게 동작할 수 있도록 설정해주었다.

- 1) Snake의 Head가 Gate와 닿을 경우 check_gate()를 실행시켜 주었다.
- 2) Snake의 전체 Body를 돌면서 Body가 하나라도 벽에 닿거나 자신의 꼬리에 닿을 경우 GameOver()로 넘겨 에러를 처리해주었다.
- 3) Snake의 Body를 증가시키는 아이템인 Growth Item을 먹은 경우엔 growth()를 실행시켜 상황에 맞춰 Snake의 Body를 늘릴 수 있도록 해주었다.
- 4) Snake의 Body를 감소시키는 아이템인 Poison Item을 먹은 경우엔 poison()을 실행시켜 상황에 맞춰 Snake의 Body를 감소할 수 있도록 해주었다.


그리고 Gate를 벽인 부분에 랜덤으로 뜨게 하기 위해 map을 for loop으로 돌면서 map의 좌표가 1(벽의 특징)인 경우 wall이라는 pair로 값을 가지는 vector에 i와 j의 값을 push 하도록 해주었다.

마지막으로는, map을 for loop을 통해 돌면서 각 요소의 특징에 맞춰 map에서 표현하기 위해 색깔을 지정해 주었다.

2. Snake의 구현

Snake의 좌표는 구조체배열을 두가지를 사용하여 표현하였다.

- 1) 먼저 Snake의 좌표를 저장하는 구조체를 이용하여 2차원배열로 생성되어 있는 map의 좌표를 저장하여 준다.

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테이크	
	Confidential	Version 1.3	2020-JUN-19

2) 키보드에서 입력 받은 key에 따라서 그 다음 이동할 다음 좌표를 지정하여 준다. key값에 따라 Snake의 머리(Head: 3)에 1칸을 더하거나 뺄 으로서 다음 좌표를 지정하여 주고, 꼬리(Tail: 4)는 이전의 Snake의 좌표를 저장하여 주도록 한다.
Snake의 좌표를 저장하였다면, 그 다음 Map에서 Snake에 해당하는 '3', '4'를 따로 표현하여 준다.

Snake에서, 조작으로 인한 Game이 종료될 조건을 만들어주었다,

- 1) 들어온 key값이 진행하고 있는 방향과 반대방향이라면 gameover로 에러처리를 해준다. (key를 입력 받았을 때 반대방향인 key의 값도 같이 저장하여 주도록 한다)
- 2) key가 들어올 때에 반대방향으로 저장되어 있는 방향과 다른 방향인 경우에만 동작하고, 그렇지 않다면 바로 에러처리로 넘어가도록 구현하였다.
- 3) Snake가 벽에 닿을 경우(구조체를 이용하여, Head의 좌표에 해당하는 map이 가지는 값이 벽(Wall: 1)과 같으면), 혹은 자신의 몸통에 닿을 경우 (3, 4) 마지막으로 일정한 꼬리의 길이보다 짧은 경우 각각 에러처리를 통해 gameover되도록 구현하였다.

3. Item 개발 내용


Item은 일정한 시간(랜덤)이 지난 후에 나타나도록 구현하고, 5초뒤에 사라지게 하도록 구현하였다.

- 1) item은 시간이 측정되지 않은 경우에, 랜덤한 시간을 구하고 시작시간을 저장해 놓도록 하였다.
- 2) 시작시간을 현재시간에서 뺀 값이 랜덤한 시간보다 크거나 같을 경우, Map의 아이템의 개수를 확인하고, 3이하인 경우에만 아이템의 좌표와, growth인지 poison인지 랜덤하게 구하도록 한다.
- 3) 랜덤한 좌표와 아이템이 지정되면 map에 나타내도록 배열에 지정된 좌표에 해당 아이템을 상징하는 번호(5, 6)를 지정하여 줌으로써 map에 표시되도록 한다. 그리고 5초이후에 사라지게 하기 위하여 종료시간을 저장해 놓도록 하였다. 그리고 종료시간을 Deque에 담고, 아이템의 좌표 또한 Queue에 담도록 한다.
- 4) Snake의 Head가 아이템(Growth, Poison)에 닿을 경우 각각의 함수를 호출하여 준다.
- 5) Growth아이템을 얻었을 경우 꼬리의 길이를 +1해주고, 방향에 따라 꼬리가 추가되는 좌표가 다르기 때문에 마지막 꼬리의 좌표에 이어질 방향의 좌표를 더해줌으로써 꼬리를 추가하여 주도록 한다.
- 6) Poison아이템을 얻었을 경우, Map에 해당좌표의 값을 0으로 바꿔 줌으로써, 마지막 꼬리를 지워주도록 하고, 전체 꼬리의 길이를 1감소하도록 한다.
- 7) Deque안에 있는 값들을 반복문을 통해 현재시간에서 그 값을 뺀 값이 5초이상이면 해당하는 아이템이 Queue의 top값에 존재하므로 해당좌표를 배열에서 0으로 바꿔준 뒤 Pop해주도록 하고, 종료시간 또한 Deque에서 pop_front해주도록 한다.

4. Gate 개발 내용

1) Gate의 생성

random, ctime 라이브러리를 이용하여 게임 시작 후 랜덤한 시간이 지나면 Gate가 생성되게 하였다. 게이트가 생성되는 조건으로 시간과 Gate에 관련하여 판별하는 변수를 각각 생성하였다. 시간이 측정되지 않은 상태와 Gate가 Map에 생성되지 않은 상태가 모두 충족되었을 때 Gate를 생성하게 하였으며, 게임이 처음 시작되었을 때에는 두 조건이 모두 충족될 수 있도록 이를 확인하는 변수를

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

통해 초기화 해주었다. Gate는 2~6초 사이의 랜덤한 시간에 생성되도록 설정하였고, 이 시간이 측정되면 시간 변수의 값을 변경하여 Map에 Gate를 생성하는 함수로 이동할 수 있도록 하였다.

Gate는 Gate의 고유색(Magenta)로 표현하여 Map상에 출력되게 하였다. 출력시에 Gate를 상징하는 번호(7)을 이용한다.

2) Gate의 위치

Gate가 생성되는 위치의 설정은 vector를 사용하였다. Map을 반복하여 돌아 벽인 부분의 좌표 값을 vector에 저장한다. 좌표(x, y)값은 pair을 이용하여 vector에 하나의 쌍으로 넣을 수 있게 하였다. vector에는 Map의 모든 벽부분의 좌표 값이 담겨있게 된다.

Gate의 위치 한 쌍을 랜덤하게 설정하기 위하여 vector를 shuffle해주었다. shuffle을 통해 Map의 윗부분부터 순서대로 담겨있던 벽부분의 좌표 값들을 랜덤하게 섞어서 vector안에 담기게 해주었기에 vector의 첫번째, 두번째 좌표 값을 가져오면 중복되지 않는 하나의 Gate 좌표 쌍을 얻을 수 있게 된다.

3) Gate의 진행

Gate가 진출하기 전에 Gate가 진출할 Gate의 위치를 체크해준다. 진출할 Gate가 가장자리일때와 아닐 때 Snake가 진출하는 방향이 각각 다르기 때문에 진출 Gate의 위치에 따라 Gate의 진행 방향을 다르게 처리해 주어야 한다. 게임의 상황에 따라 Snake는 하나의 진출 방향을 가지게 되는데, 이 방향을 위의 switch문을 통하여 상, 하, 좌, 우 4가지 방향 중 하나를 설정할 수 있게 하였다.

3-1) 가장자리에 있을 때

Gate의 좌표가 가장자리인지 체크를 해준 후 목표에서 설정한 진출 방향대로 Snake의 진행 방향을 설정해준다.

3-2) 가장자리가 아닐 때

Gate의 좌표가 가장자리에 위치해 있지 않은 경우, 진입 Gate로 진행한 방향 그대로 게임을 진행하여야 한다. 그러나 Map의 모양에 따라서 진행방향대로 진출할 수 없는 상황(진출이후 바로 Game Over가 되는 경우)이 존재한다. 이러한 경우에는 위의 목표에서 설정한 방향대로 진출 방향을 수정해주어야 한다. 그렇기 때문에 진출 Gate에서 진행 방향대로 1칸 움직였을 때 좌표가 벽인지 아닌지 판단해주어야 한다. (예를 들어, 진행 방향이 왼쪽일 때 진출할 Gate의 왼쪽 좌표가 벽으로 구현되어 있는 상황이다.) 판단 결과에 따라 Snake의 진행 방향을 재설정 해준다.

4) Snake의 이동

Snake가 Gate에 진입했을 때, Snake의 머리를 진출 Gate 안쪽 좌표로 설정한다. Snake의 길이만큼 for문을 돌려서 Snake의 꼬리들이 이전 좌표를 가지고 가도록 설정하였다. 이때 Snake는 앞에서 설정한 진행 방향으로 움직인다.

5) Gate 삭제 및 재생성


Gate를 모두 통과하였으므로 Gate 생성시에 만들었던 Gate 변수를 초기화하여 Map에 더이상 게임에 진행될 Gate가 없으며, 새로운 Gate를 생성할 수 있는 상태임을 표현하였다. 이로 인하여 새로운 Gate는 2~6초 사이 랜덤한 시간 내에 재생성 된다.

Map상에서 Gate를 다시 일반 벽으로 출력해주기 위해 Gate 좌표 부분에 설정해준 Gate 고유 번호(7)을 벽의 고유 번호(1)로 재설정해준다.

5. 점수요소의 구현

Map이 띄워져 있는 곳 옆에 새로운 윈도우 2개를 기존 윈도우에 만들도록 한다.

1) ScoreBoard:

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트명	Snake Game	
	팀명	스테이크	
	Confidential	Version 1.3	2020-JUN-19

현재의 꼬리길이(클래스내의 변수인 Tail사용)와 유니코드 ('\'), 그리고 미션에 해당하는 꼬리 길이 (Stage.cpp - TAIL)로 표현하고, 그 외에도 아이템을 얻었을 때에 count하는 각각의 변수를 통해 나타내도록 한다.

Ncurses를 통해 mvwprintw(!=mvprintw)로 각각을 출력하여 주고, wborder을 통해 각 윈도우의 경계를 표시하여 준다.

2) Mission:

Stage.cpp파일을 통해, 현재 스테이지에 해당하는 미션의 난이도를 설정하여주고, 그 값을 받아와, 윈도우에 출력해 주도록 한다.

Mission또한 ScoreBoard와 같이, mvwprintw와 wborder을 사용하여준다.

3) 해당 미션을 성공했을 경우, char타입의 변수(tail, growth, poison, gate의 횟수가 미션을 성공하였는지 여부를 체크하는 변수)를 'v'로 변경해준다. 모든 값이 'v', 즉 모든 미션을 성공했을 경우, stage와 관련된 함수를 변경해주고, map을 변경해준다. 그 후, Mission또한 Stage에 맞추어 다시 설정되도록 한다.

2.2.2 시스템 구조 및 설계도

작성요령 (30점)

프로젝트의 각 세부 목표의 주요 기능(알고리즘 등)에 대해서 기술한다. 세부 목표별로 수정한 프로그램 소스 파일을 나열하고, 해당 파일에서 세부 목표를 달성하기 위해 작성한 클래스/함수에 대해 나열하고, 각 요소에 대해 간략한 설명을 작성한다. 또한 각 요소의 개발자를 명시한다.

Main: 이연주, 김예리, 심혜린

while문을 계속 반복해주면서, 에러가 들어오지 않는 경우 계속해서 Map을 다시 그려주고, Score를 띄워주도록 한다. 그리고 0.5초마다 값을 입력 받도록 하여 게임을 진행하게 하도록 한다.

<Item & Gate>

1) 생성

만약 아이템과 게이트가 시간이 측정되지 않은 상태인 경우에는 아이템을 띄울 준비를 해야 한다는 의미이므로 아이템의 시작점, 게이트의 시작점을 각 변수에 저장하고 각각 랜덤한 초를 설정한다. Item은 시간이 측정되지 않은 경우를 검사하는 변수 t_ck, Gate는 시간변수 tga_ck와 map에 Gate가 그려지지 않은 경우를 측정하는 변수인 gate_Assign 2가지를 이용한다. 시간 측정과 관련된 코드는 다음과 같다.


```
start_t = time(NULL); // 시간 측정
uniform_int_distribution<int> rc(1, 5); // 랜덤
rt = rc(gen);
t_ck = 1; // 시간 측정됨
```

시간이 측정되어 있으며, 랜덤으로 설정한 시간이 지났음을 판단하는 조건

```
t_ck == 1 && time(NULL) - start_t >= rt
```

그리고 설정한 초가 지났을 경우 아이템 같은 경우는 map에 그려진 아이템의 개수가 3이하인 경우에 assign_()함수를 호출하여 아이템을 찍어주고 정지점을 찍어준다.

```
stop_t = time(NULL);
```

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

게이트 같은 경우에는 시간을 측정되어 있으면서 Map에 게이트가 없는 경우 아이템과 같은 경우를 반복한다. Item과 Gate는 코드 수행 후 시간 변수를 0으로 초기화하여 시간이 다시 측정될 수 있게 한다.

2) Item 삭제

그리고 아이템이 존재하면(dq에 정지점이 저장되어 있는 경우) dq를 탐색하면서

```
for (;it != dq_t.end(); it++){
```

```
    t = *it;
```

현재시간과 비교하여 5초가 지난 경우에는 queue에서 아이템의 좌표를 가지고 와서 0으로 만들어주고 pop시켜주도록 한다.

```
    if (time(NULL) - t >= 5){ //5초 이상인 경우
```

```
        x = q_xy.front().first; y = q_xy.front().second; //x, y값을 받아
```

```
        snake.setZero(x, y); //아이템 삭제
```

그리고 정지점 또한 제외시켜 줌으로써 map에서

아이템을 지운다.

```
        q_xy.pop();
```

```
        dq_t.pop_front(); //아이템과, stop지점을 각 container에서 제외
```

```
    }
```

3) Gate 삭제


Gate는 Snake가 Gate를 통과하는 동안 map에 유지하며 통과 이후 map에서 삭제해야 한다. 게임의 진행에 따른 Gate의 변화를 처리하기 위해 Gate가 map에 존재하고 있음을 의미하는 변수 gateAssign을 이용하였다. map에 Gate가 없으면 Gate 생성을 위해 gate_xy() 함수 호출 후 gateAssign을 1로 수정하였으며, Snake가 Gate를 통과한 이후 Gate의 임무 수행을 마쳤음을 표현하기 위해 Snake가 Gate를 통과하는 Exit_gate(int, int, int)함수 마지막에 gateAssign을 0으로 초기화하여 main에서 다시 랜덤한 시간 소요 후 Gate를 재생성 할 수 있도록 설정하였다.

<GameOver>

클래스 내부함수인 GameOver에서 error처리를 통해 throw되어 들어온 인자값에 대해서 해당 GameOver문구를 출력해주도록 switch문을 구현하였다.

Snake Class default Constructor : 이연주, 김예리, 심혜린

Map을 stage1으로 먼저 할당하여주고, 모든 미션의 check변수는 0으로 설정하여준다. 그리고 방향의 좌표 또한 설정하여주고 게임이 시작하면, 모든 Snake의 head와 tail들의 좌표를 고정시켜주며, 방향 또한 왼쪽으로 고정하여 준다. 이동방향과 반대되는 오른쪽 방향을 fail_dir로 설정하여주고, Gameover를 판별하는 변수와 꼬리의 길이 또한 default값으로 설정하여 주도록 하였다.

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

Map : 이연주, 김예리, 심혜린

- void Map()

게임을 위해 필요한 모든 요소들의 특징을 명시해주고 ncurses를 이용해 색을 설정해주었다. map안에서 Snake의 좌표가 명시한 요소들의 특징과 일치할 경우 실행해야할 함수들로 넘겨주거나, 에러처리를 해주어 각각의 에러에 맞는 출력을 하도록 하였다.

Gate의 생성을 위해 vector를 사용하였다. 벽의 좌표 값을 담은 vector wall을 사용하였으며, Map()함수에서 이중 for문을 돌며 해당 vector에 map의 모든 벽의 좌표 값 쌍(y, x)을 담았다. vector안에 x, y 2개의 데이터를 넣기 위해 pair를 사용했다. 해당 코드는 아래와 같다.

```
wall.push_back(pair<int, int>(i,j));
```

Snake 조작 : 이연주, 심혜린

- void Game()

키보드 입력 값이 들어오지 않으면 이전의 key(dir)값을 유지하여 주고, 새로운 key값이 들어오게 되면, 그 값이 이전의 방향과 반대가 아닌 경우(!= fail_dir)에 그 방향과, 반대방향의 key값을 정해주도록 한다. 그리고 꼬리의 마지막부분(아직 이동하기 전)은 0으로 설정하여 주고, 꼬리의 끝 부분부터 자신의 앞 꼬리의 좌표를 받아와 이동하도록 하고, 꼬리가 다 이동하였다면, snake의 head부분은 입력 받은 key값의 방향을 적용하여 좌표를 정해주도록 한다.

Item , Mission : 이연주

-void Score()

미션을 stage.cpp에서 각 stage별로 랜덤한 난이도로 정하여 map윈도우 옆에 새로운 윈도우로 창을 띄워준다. 현재 미션의 진행상황과 미션을 나타내 주고, 미션이 성공하였다면 'v'표시를 해줌으로써 체크할 수 있다.

그리고 만약 4가지 미션에 대해 모두 통과되었다면 카운트하여 해당 스테이지로 넘어가도록 설정하였다.

-int GameOver(int ch = -1)

ch는 gameover의 종류로써 들어온 gameover종류를return 해주고 throw문으로 넘어가 main의 while문을 break하고 나서 에러의 이유를 출력해주도록 구현하였다.

-int itemTmp()

아이템의 개수가 3개이하인지 확인하는 함수이다. 한 화면, 즉 map에 한번에 나타날 수 있는 아이템의 개수는 최대 3개이므로 반환 값이 3보다 크면 아이템을 만들지 않는다.


-void assign_l()

아이템의 종류를 랜덤하게 설정하고 좌표도 랜덤하게 설정하도록 하는 함수이다. 랜덤하게 뽑은 x, y의 좌표를 pair로 묶어서 만약 그 좌표가 벽이나 snake와 같지 않은 경우에 queue에 넣어준다. queue에 넣은 좌표 값은 만약 5초 이내에 아이템을 먹지 않았을 경우 pop해주어 아이템을 사라지게 하려는 용도로 사용되었다. 그리고 해당 좌표에 랜덤으로 설정한 아이템을 map에 찍어준다.

-int getXPoint(); x의 좌표를 랜덤하게 설정하기 위한 함수이다.

-int getYPoint(); y의 좌표를 랜덤하게 설정하기 위한 함수이다.

-void growth()

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

growth아이템을 먹었을 경우 처리하기 위한 함수로, 아이템을 먹었다면, tail의 크기를 1 늘려주고, 방향에 따라 (next_X와 next_Y 클래스 내부변수를 이용하여) tail의 좌표는 tail-1번째 꼬리의 값을 이용하여 나타내 준다.

-void poison()

poison아이템을 먹었을 경우 처리하기 위한 함수로, 아이템을 먹었다면, tail의 크기를 1 줄여주고, 해당 좌표는 다시 0으로 설정하여 꼬리를 사라지게 해준다.

-void setZero(int, int)

들어오는 좌표 값을 map에서 0으로 설정하게 하기 위한 함수이다.

-stage.cpp

get_Cnt 함수에 stage를 체크하는 변수를 인자로 전달하여, stage에 따라 랜덤한 난이도를 설정하도록 하고, 그 값을 해당파일의 지역변수에 할당해 주도록 한다. 그리고 각 stage에 대한 Map을 2차원 배열로 해당 파일에 선언해 주도록 한다.

Gate 좌표 생성: 심헤린

- void gate_xy()

Gate를 map에 생성시키기 위하여 Gate의 랜덤한 좌표 값 한 쌍을 생성해주는 함수다. main에서 gate가 생성될 수 있는 조건에 충족되면 이 함수를 호출한다.

Map()에서 벽의 좌표를 담은 vector wall를 shuffle함수를 통하여 랜덤하게 재구성한다. vector wall은 랜덤한 순서의 좌표쌍으로 구성되었기에 해당 vector의 첫번째와 두번째 좌표를 gate의 위치로 설정하면 Gate가 생성될 때마다 중복되지 않는 랜덤한 위치의 좌표를 갖게 된다.

```
shuffle(wall.begin(), wall.end(), gen);
```


생성된 Gate의 좌표 값을 저장하기 위해 iterator를 사용한다. 좌표 값은 class에 private로 Gate별로 x, y를 각각 선언한 4개의int type 변수에 저장한다. Gate가 가지게 될 좌표는 vector의 처음 2개의 값이므로 iterator를 첫번째 Gate는 wall.begin(), 두번째 Gate는 wall.begin()+1로 설정하였다. Map()에서 이중 for문을 돌며 vector에 좌표를 저장할 때 좌표 x, y의 순서쌍은 (x, y)가 아닌 (y, x)로 저장되기 때문에, iterator가 Gate의 y값은 first, x값은 second로 접근하도록 하였다. 이후 해당 좌표에 Gate를 출력하도록 해당 위치의 값을 벽을 의미하는 1에서 Gate를 의미하는 7로 초기화 해준다. Iterator 관련 코드는 다음과 같다.

```
vector<pair<int, int>>::iterator iter;
iter = wall.begin();
g1_y = iter->first;
g1_x = iter->second;
```

Gate(가장자리) : 김예리

-void check_gate()

void gate_xy()를 통해 만들어진 Gate가 map에서 확인되었을 때 실행되는 함수이다. 이는 현재 Snake의 머리 좌표와 Gate의 좌표를 앞서 말한 함수에서 생성된 두 개의 Gate중 어느 좌표 값이 들어온 Gate인지를 확인하여 나갈 Gate의 좌표 값을 void is_Edge(int, int)로 넘겨준다.

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

-void is_Edge(int, int)

void check gate()로 넘어온 나가는 Gate의 좌표 값을 인자로 전달받아 이 좌표 값이 가장자리인지 판별해주는 함수로 함수 내부에서 좌표 값이 가장자리일때와 아닐 때의 처리를 해주었다. 그리고 가장자리인 경우에는 진행 방향이 고정되어 있으므로 void Exit_gate(int, int, int)로 좌표 값과 방향을 판별하는 값을 넘겨준다.

-void Exit_gate(int, int, int)

Snake가 다시 Gate를 나갈 수 있도록 처리해주는 함수이다. switch문은 사용하여 1(상), 2(하), 3(좌), 4(우)로 case를 나누었고, 가장자리인 경우에는 진행 방향이 고정되어 있기 때문에 인자로 넘겨온 1,2,3,4로 방향을 판별한 후, 이에 맞는 처리를 해 Snake의 진행방향을 바꾸어 주었다. 이후 Snake의 머리를 진출하는 Gate으로 설정하여 이동시킨다. Snake의 꼬리들은 이전 좌표를 따라간다.

Gate(가장자리가 아닐 때): 심혜린

Gate가 가장자리가 아닐 때의 처리 역시 위의 Gate가 가장자리일 때 사용한 is_Edge(int, int)와 Exit_gate(int, int, int)함수를 이용한다. 이 경우 is_Edge(int, int)에서 가장자리인지 판단하는 조건문에 부합하지 않을 때인 else문에서 조건을 판단한다. 가장자리가 아닐 때에는 진출하자마자 GameOver가 발생할 수 있기 때문에 이를 방지하기 위하여 현재 Snake의 진입 방향이 진출 Gate로 진출할 수 있는지 여부를 확인하여야 한다. 가장 먼저 Snake의 진입 방향을 체크해주고 진행 방향에 따라 진출 방향이 벽으로 막혀 있는 경우를 판단해 준다. 조건은 다음과 같다.

```

상: map[ny+1][nx] == 1
하: map[ny-1][nx] == 1
좌: map[ny][nx-1] == 1
우: map[ny][nx+1] == 1

```

이 조건에 해당하는 경우 목표 설정시에 정해 두었던 방향을 Exit_gate()의 3번째 인자로 넘겨주며, 해당하지 않는 경우(즉, 진행 방향 그대로 진출할 수 있는 경우) 진입한 진행방향과 같은 방향을 인자로 넘겨준다. 이후 Exit_gate(int, int, int)을 통해 Snake는 Gate를 무사히 통과할 수 있게 된다.

Gate 수행 이후: 심혜린, 이연주, 김예리

Gate 통과 이후 Exit_Gate(int, int, int)안에서 Gate관련 값들을 수정 및 초기화한다. Snake는 Gate를 진출이후 Gate관련 모든 수행을 마치기 때문에, Snake이동 코드 하단에 밑의 코드를 추가해준다.

Gate를 통과했으므로 Score의 게이트 미션 성공 카운트를 올려준다.


```
gate_Cnt += 1;
```

Gate 수행 임무를 마쳤으므로 해당 변수 초기화 후 새로운 Gate를 생성할 수 있도록 한다.

```
기존: gateAssign = 1; 변경 후: gateAssign = 0;
```

위의 gate_xy()에서 map의 벽을 Gate로 수정해주었던 것을 다시 벽으로 바꿔주어 map에서 gate를 삭제해준다.

```
기존: map[g1_y][g1_x] = 7; 변경 후: map[g1_y][g1_x] = 1;
```


 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

2.2.3 활용/개발된 기술

작성요령 (10점)

프로젝트 수행에 사용한 외부 기술/라이브러리를 나열하여 작성한다. 각각 기술을 이 프로젝트에 적용할 때, 도움 받거나 해결하고자 하는 기능에 대해 상세히 설명한다.

NCURSES / STL 라이브러리 등을 포함하여 설명한다.

또한, 이 프로젝트를 수행하면서, 새롭게 고안한 알고리즘 등이 있다면 설명한다.

-1) NCurses: 전체적인 Map, Snake, Score등을 표현하기 위해 사용된 라이브러리로써, GUI프로그래밍을 하도록 도와준 라이브러리이다.

Map과 Snake는 유니코드를 통한 문자로 표현하였고, color를 표현하는 init_pair를 이용하여 각각의 성질에 따라 다른 색상 및 유니코드로 표현하려 하였다.

또한 Score는 새로운 윈도우를 생성하는 WINDOW를 이용하였고, 경계를 표시하는 wborder, 그리고 mvprintw와는 다른 mvwprintw등을 사용하였고, 모든 윈도우들을 refresh혹은 wrefresh를 이용하여 계속 새로 그려주었다.

-2) Vector(STL): vector는 Gate가 생성되는 좌표를 정해주기 위해 사용되었다. Gate는 Map의 벽부분에 랜덤하게 2개 생성된다. 따라서 Gate의 위치를 선택하려면 Gate가 생성될 수 있는 좌표만 담겨 있는 곳이 필요했다. 이를 위해 vector를 map에 있는 모든 벽 좌표(x, y)쌍을 저장하기 위한 용도로 사용하였다.

-3) Deque(STL): 종료시간을 저장하기 위한 Container로 사용하였다. 앞의 요소를 삭제하기에 Vector를 사용하는 데에는 무리가 있었기 때문에, Deque를 선택하였다. Deque에 종료시간을 저장하고 계속 현재시간과 비교하면서, 일정한 시간이 지나게 되었다면 해당 요소를 front_pop해주는 데 사용하였다.


-4) QUEUE(STL): queue는 아이템의 좌표를 저장하는 Container로 사용하였다. 5초뒤에 아이템이 사라지도록 구현하기 위해서는 아이템의 좌표를 알아 두어야 했기 때문에 Queue를 사용하여 좌표를 저장하도록 하였다. Deque에서 종료시간에 해당하게 되는 경우에, 그에 해당하는 아이템의 좌표를 가져와서, setZero라는 함수를 호출하여 해당 좌표의 배열 값을 0으로 초기화 함으로써, 아이템이 사라지도록 구현하였다. 배열 값을 0으로 초기화 한 후에는, Queue에서 pop하도록 한다.

-5) Kbhut(오픈소스): 입력 값이 들어가는지 아닌지 확인하는 함수이다. 입력 값이 들어오지 않으면 이전의 방향을 유지하고, 입력 값이 들어오면 들어온 key값을 입력 받아 방향을 바꿔주어야 하기 때문에 값이 들어오는지 아닌지 판단하는 방법이 필요해서 kbhit라는 오픈소스 함수를 사용했다.

-6) Getch(오픈소스): 입력 값이 들어올 때까지 기다리는 함수로써 입력 값이 들어오면 그 값에 해당하는 key값을 찾아서 방향을 바꾸어 주도록 한다.

-7) Random: Item의 요소(Growth, Poison)인지, 혹은 아이템이 몇 초 뒤에 생성될 것인지, Gate는 몇 초 뒤에 생성될 것인지, 각 아이템 및 gate의 좌표를 어떻게 할 것인지, 각 Stage마다 어떤 미션을 줄 것인지에 대한 모든 것은 랜덤한 상황에, 랜덤한 상황으로 발생하기 때문에, Random을 사용함으로써, 완전한 난수의 값을 얻게 할 수 있다.

-8) Time(Ctime): 아이템 및 게이트가 랜덤한 시간에 발생하게 하기 위해서, 아이템이 5초뒤에 사라지게 하기 위해서는 시간을 측정해 주어야 하기 때문에 사용하였다. 아이템 및 게이트가 랜덤한 시간이 지정되면, 시작점과 현재 시간을 계속 비교하면서 해당시간이 경과한 경우 아이템을 생성하게 하였고, 종료조건 또한 마찬가지로, 종료점을 저장한 후 현재 시간과 계속 비교하여 5초가 경과한 경우 해당 아이템을 사라지게 하는데 사용하였다.

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테이크	
	Confidential	Version 1.3	2020-JUN-19

-9) Shuffle(Algorithm): Gate의 위치 한 쌍을 랜덤하게 설정하기 위해서 사용하였다. Gate의 위치가 벽인 부분에 랜덤하게 나타나도록 해야 하기 때문에 벽 부분의 좌표 값들을 Shuffle을 통해 랜덤하게 섞어서 이를 vector안에 담기도록 해주었다. 좌표 한 쌍이 들어있는 vector의 첫 번째, 두 번째 값을 가져오면 중복되지 않는 하나의 Gate 좌표쌍을 얻을 수 있고 이를 Gate로 설정한다.

-10) INT_MAX(Climit): 시작점과 종료점을 서넴문에서 선언할 때 초기화로, 큰 값을 주어야 하는데 특정한 값을 크게 주는 것 보다는 int형의 최댓값으로 선언 및 초기화를 하는 것이 더 좋을 것 같아, 선언하였다.

2.2.4 현실적 제한 요소 및 그 해결 방안

작성요령 (5점)


제안된 프로젝트의 단계 별 수행에 있어, 제한 요소를 찾아 작성한다. 해당 제한 요소를 해결하기 위해서 어떤 방법으로 해결하였는지 작성한다.

- 방향을 설정하기 위하여 getch()라는 모듈을 이용하고자 하였으나, 입력이 들어오지 않을 때 이전의 방향을 유지하는 데에 어려움이 있었다. 윈도우에서는 'kbhit()'라는 모듈을 이용하여 입력 값이 들어왔는지에 대한 유무를 판단할 수 있는데 리눅스에서는 지원하지 않아, 오픈소스를 인용하여 kbhit를 사용하고자 했다.
- 아이템이 게임이 시작한 이후로 계속 랜덤한 시간이후에 발생하고 5초뒤에 사라지게 하는 과정에서, clock()모듈을 사용하였으나, 타입을 지정하거나 맞추는 데에 어려움이 생겨 일정한 시간을 측정하는 데에 어려움을 겪어, sleep을 이용하여 스레드를 통해 프로그램을 구현하려 하였으나 모든 프로그램이 멈추거나, 느려지는 상황 또한 발생하였다. 그러다 time()모듈을 발견하여, 타입을 지정하기에 쉬워 위와 같은 문제를 쉽게 해결할 수 있었다.
- 게임이 실행되면서, 가장자리(벽)부분이 사라지는 현상이 발생하였다. gate와 item중 어느 부분에서 일어나는지 확인해 보기 위해 출력을 해본결과, 가장자리인 경우, queue에 해당 좌표를 담으면 안되는데, 가장자리인 경우까지 queue에 담았고, 5초뒤에 사라지는 함수를 호출하게 되어, 해당 좌표를 0으로 바꾸게 된다는 것을 파악하게 되었다. 그래서 가장자리가 아닌 경우에만 queue에 담도록 조건을 추가하여 줌으로써, 해당 오류를 해결하였다.
- Map을 돌면서 Vector에 벽에 해당하는 좌표 값을 담을 때 x, y 쌍을 담는 방법에 있어 어려움이 있었다. 처음에는vector대신 tuple을 사용하였지만 에러가 발생하여 vector에 pair를 사용하여 문제를 해결하였다.

2.2.5 결과물 목록

작성요령 (5점)

결과물 목록을 작성한다. 목록은 제출하는 파일과 각 파일의 역할을 간략히 설명한다.

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

- map_fin.cpp: Snake와 관련한 클래스의 멤버함수가 구현되어 있다. 실제 프로그램의 main이 구현되어 있는 파일로, Snake Game의 동작이 실행되는 파일이다.
- map_fin.h: Snake Game의 동작을 위해 만든 Snake 클래스의 프로토타입을 정의하여 둔 헤더파일이다.
- stage.cpp: 각 Stage마다 다른 모양으로 하기 위한 stage배열이 선언되어 있으며, 각 Stage마다 미션을 랜덤하게 받기 위하여 구현되어 있는 함수 또한 선언되어 있다.

3 자기평가

작성요령 (5점)

프로젝트를 수행한 자기 평가를 서술한다. 팀원 개개인의 자기 평가가 포함되어야 하며, 본인의 역할, 프로젝트 수행 시 어려운 점, 도움이 되었던 점, 이 프로젝트 운영에 개선이 필요하다고 생각하는 점을 충분히 서술한다.

이연주(20191644):


아이템 생성과 미션에 대한 부분을 진행하였다. 아이템의 처리와, 미션창을 띄우는 부분은 어려운 부분이 없었으나, 처음 Map을 생성할 때에 0.5초마다 틱을 주고 snake의 입력 값이 없을 때에 그 이전 값을 유지하려고 하는 부분과 아이템이 랜덤한 시각에 생성되고, 5초뒤에 사라지도록 하는 부분에서 구현하는데 큰 어려움이 있었던 것 같다. 처음에는 규모가 좀 크다고 생각해서 막막함이 많이 컸었는데, 각자 파트를 나누고 하나하나 합쳐보니, 처음에 생각했던 것만큼 규모가 크다고는 이제 생각이 들지 않는 것 같다. 이 프로젝트를 통해 클래스의 구조와, time, random등을 사용하는 데에 도움이 되었으며 특히 Container를 사용하는 방법 등을 익히는 데에 큰 도움을 얻은 것 같다. 그리고 이 Game을 구현하는 데에 아쉬운 점이 있었다면 가독성을 보다 높이기 위해 파일을 여러 개로 나누어 모듈화를 좀더 세심하게 하였으면 좋았을 것 같다는 아쉬움을 갖게 되었다. 그러기 위해서는 코드의 구조에 대해서 조금 더 정확하게 파악을 해야 할 것 같다는 생각을 했다.

김예리(20191565):

게이트가 가장가리가 아닐 때를 처리하였다. 처음에 게이트의 좌표를 랜덤으로 구현하는 것을 만들지 못하여서 먼저 다른 부분들을 생각하는 대로 만들게 되었다. 그래서 코드를 작성할 때 내가 만든 함수가 잘 작동할 수 있을지에 대한 걱정이 많았다. 하지만 게이트를 랜덤으로 띄우는 것을 성공한 후에 만든 함수가 잘 실행되는지를 확인하고 게이트를 구현하기 전 부분을 팀원들과 같이 수정해 나가면서 내가 맡은 부분을 구현하기 위해 필요한 부분들을 다시 생각해 볼 수 있었다. 그래서 유동적으로 함수가 잘 구현되기 위해 새로운 함수를 추가해 맡은 부분의 동작을 이루어 낼 수 있었고, 처음 생각했던 흐름대로 진행되어서 뿌듯했다. 또한, 프로젝트를 진행하면서 코드의 흐름이나 팀원들의 아이디어를 들으면서 내가 생각하지 못했던 부분에 대해 다시 깊게 고민해볼 수 있었던 시간이었다.

심혜린(20191620):

게이트에 관한 부분을 주로 개발하였다. Vector와 pair을 사용하는 아이디어를 생각한 이후 게이트를 처리하는 알고리즘의 설계에 관해서는 생각보다 구현하기 어렵지 않았지만, ncurses 라이브러리가 익숙하지 않아서 고안해낸 알고리즘을 어떻게 화면에 출력하여 처리할지에 관하여 고민이 많았던 것 같다. 또, 평소에 클래스를 잘 사용해보지 않아 클래스 구현에 있어 어려움을 겪었다. C++이라는 언어에 익숙하지 않았던 만큼 프로젝트를 처음 접했을 때 많이 막막하고 어렵게 다가왔지만, 팀원들과 함께 프로젝트를 진행하며 협업의 중요성에 대해 깨닫고 프로젝트의 재미를 찾게 된 것 같다. 또한, 프로젝트 수행 과정 중 막혔던 부분에서 내가 생각하지 못했던 방법으로 코드를 구현하는 팀원들의 모습을 보며, 코딩을 할 때 더욱 다양한 방식으로 생각을 할 수 있는 계기가 되었다. 기존에 능숙하게 사용하지 못했던 Random, vector, pair, iterator, time등에 대해 공부할 수 있었으며, 평소에 어려워했던

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테인크	
	Confidential	Version 1.3	2020-JUN-19

클래스에 익숙해지는 데에 많은 도움을 얻었다. 아쉬운 점은 소스 코드가 하나의 메인 파일에 너무 긴 코드로 이루어 졌다는 점이다. 파일을 분리하고 싶었지만 class와 라이브러리에 대한 지식 부족으로 실현하지 못하였다. 추가적으로 게임의 시작과 끝, 스테이지의 이동 때 게임 이펙트를 추가하고 싶은 욕심이 있다.


참고 문헌

번호	종류	제목	출처	발행년도	저자	기타
1	웹페이지	[Linux] linux에서 getch() 및 kbhit() 사용하기	출처: https://anythink.tistory.com/entry/Linux-linux에서-getch-및-kbhit-사용하기 [투명링크]	2013. 12. 31. 12:17		
2	웹페이지	Linux Non-blocking getch() - kbhit()대체 가능	출처 : https://m.cafe.daum.net/jwmyself/Bb1/55?q=D_nNt8ZxUMUJU0&	2004/11/26		
3	웹페이지	C / C ++ – Linux의 kbhit () 함수	https://eastskykang.wordpress.com/2015/01/28/c-c-kbhit-function-in-linux/	2015 년 1 월 28 일	강동호	
4	웹페이지	C, C++ 프로그램 실행 시간 측정	https://hijuworl.d.tistory.com/1	2018. 3. 29. 23:56		
5	웹페이지	C++ std::Shuffle 함수	http://kinggodzero.insoya.com/221454233850	2019.01.29		

4 부록

작성요령 (15점)

프로젝트의 결과물을 사용하기 위한 방법에 대해서 작성하세요.

 국민대학교 소프트웨어학부 C++ Project	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	스테이크	
	Confidential	Version 1.3	2020-JUN-19

4.1 사용자 매뉴얼

<사용 매뉴얼>

1) Snake 조작 방법

↑, ↓, ←, → 상 하 좌 우 4가지의 방향키를 이용하여 Snake를 움직인다. Snake 조작 시 주의해야할 사항이 있다. 먼저 Snake는 현재 진행하고 있는 방향은 Snake의 Head가 기준이며, 이의 반대방향인 Tail방향으로 이동할 수 없다. 예를 들어 Snake가 오른쪽으로 진행하고 있는데 왼쪽 키를 누른다면 이는 Game Over에 해당한다. 다음으로는 Snake는 하얀색으로 표시된 벽을 통과할 수 없으며 자신의 Body 또한 통과할 수 없다.

2) Item의 역할

Snake Game진행 시 Green Square과 Red Square 두 가지의 아이템이 나타난다. 첫 번째로 초록색 아이템은 Snake의 꼬리를 +1 추가시켜주는 아이템이다. 두 번째로 빨간색 아이템은 Snake의 길이를 -1 해주며 주의해야할 점은 Snake Body의 길이가 3보다 작아지면 Game Over이다.

3) Gate의 역할

Snake Game 진행 시 Magenta Square가 Map의 벽 부분에 한 쌍으로 나타난다. 두개의 Gate중 아무 곳으로 진입할 수 있으며, 한 곳에 진입하면 남은 Gate는 진출하는 Gate가 된다. 게임의 미션에 따라서 Gate를 n회 통과하여야 한다.

4) Game clear

Snake Game을 Clear하기 위해서는 stage1부터 stage4까지 도달하여 Snake Game 시작 시 우측에 보이는 Misson을 완료해야 Game Clear를 할 수 있으며 각 stage마다 미션은 랜덤으로 주어지게 된다. 이를 성공했을 시 다음 stage로 넘어가게 되고, Misson의 B는 게임을 하면서 도달해야 할 Snake의 Body길이를 뜻하며 +는 Growth Item(초록색 아이템) -는 Poison Item(빨간색 아이템) G는 통과해야할 Gate를 뜻한다.

4.2 설치 방법

map_fin.cpp

map_fin.h

Makefile

- 1) 이 세가지 파일을 같은 Directory안에 저장한 후 make를 실행하여 준다.
- 2) 명령창에 make를 입력한 후make를 통한 컴파일이 성공적으로 이루어진 것을 확인한다.
- 3) terminal(Linux)이나 cmd(Window)에서 ./snake를 작성한 다음 Enter키를 누른다.
- 4)Snake Game을 진행한다.