




국민대학교
소프트웨어융합대학
소프트웨어학부

C++프로그래밍 프로젝트

프로젝트 명	C++ Snakegame with ncurses
팀 명	C 팀
문서 제목	결과보고서

Version	1.4
Date	2021-06-17

팀원	한승진 (팀장)
	이민지
	채지윤

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

CONFIDENTIALITY/SECURITY WARNING

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

문서 정보 / 수정 내역

Filename	최종보고서- C++ Snakegame with ncurses.doc
원안작성자	이민지, 채지윤, 한승진
수정작업자	이민지, 채지윤, 한승진

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2021-06-17	채지윤	1.0	최초 작성 및 수정	
2021-06-18	이민지	1.1	내용 추가 및 수정	수정된 연구내용 추가
2021-06-18	한승진	1.2	내용 추가 및 수정	1번, 3번 내용 수정

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

목 차

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

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

1 개요

1.1 프로젝트 소개

국민대학교 c++ 프로그래밍 교과목에서 c++언어와 ncurses 라이브러리를 활용하여 제작한 snake game입니다.

Snake game은 startscene, gamescene , gameoverscene 3가지 단계로 나뉘어서 게임이 진행됩니다. 이러한 단계들은 IScene.h을 상속받아 통합적으로 관리하게 되고 snake , item , gate 등 게임을 진행하는 과정에 필요한 요소들은 gamescene내에서 독립적으로 구현됩니다. 구성 요소들은 단계의 Update 메소드에서 독자적으로 호출되고 그에 따라 변경된 내용이 mapfile에 적용됩니다. 모든 요소들이 Update 되면 마지막 결과가 각 단계가 가지고 있는 Render 메소드를 통해 출력됩니다. 3가지 단계를 모두 끝내면 게임이 종료되게 됩니다

협력 개발은 github에서 이루어졌고 코드에 대한 수정사항이 있을 때 마다 카카오톡 단톡방에 수정된 내용과 관련 코드를 피드백 받고 최종 수정본이 github main branch에 반영되었습니다. 또한 다른 팀원들이 구현한 코드에 대하여 구조를 물어보거나 C++관련된 문법이나 각 단계별 구조에 대해다른 팀원의 의견을 받을 때도 카카오톡을 통해 이루어져 원활하게 소통 아래에 개발이 이루어 졌습니다.

외부 라이브러리는 ncurses를 사용하였는데 이는 텍스트 모드에서 UI를 구현할 수 있는 프레임 워크를 제공하는 모듈입니다 윈도우 등을 만들 수 있는 함수를 제공하며, 보다 확장된 라이브러리를 통해 패널, 메뉴, 폼 등 기본적인 curses라이브러리를 사용할 수 있습니다. 본 프로젝트에서는 기본적인 출력함수만 사용하였습니다.

ncurses는 웹사이트에서 wget명령어를 통해 다운받아 해당 디렉토리로 이동 후 ./configure를 통해 환경을 설정합니다. 이후 make와 make install 명령어를 통해 빌드를 하고 인스톨을 수행합니다. 성공적으로 ncurses가 설치되었다면 ls -la /opt/ncurses명령어를 통해 설치 여부를 확인할 수 있습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

2 개발 내용 및 결과물

2.1 목표

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
6단계	아이템 별 효과 구현	적용

1. Map

- 각 스테이지에 따른 Map 구성이 목표로 둔다. 각 Stage는 4개가 있는데 txt파일로 작성되어 62 x 32의 크기를 가진다. 각 stage는 요소들을 표현하는 숫자로 표현되어 있는데, 일반적인 벽(wall)은 '1' 그리고 Immune wall은 '2'로 마지막으로 snake가 움직일 수 있는 활동공간은 '0'으로 둡니다.
- 외부에 존재하는 map파일을 읽어 NCURSES Library 함수들을 이용해 화면으로 출력하는 것이 1단계의 목표입니다.

2. Snake

- 초기 길이: 3
- 초기 방향: LEFT
- Snake 색상: 파란색
- 사용자가 방향키를 입력하면 snake는 1초마다 입력된 방향으로 움직입니다.
- 몸의 길이가 3미만이 되면 게임이 종료됩니다.
- 진행방향과 반대 방향의 키를 누를 경우 게임이 종료됩니다.
- 머리가 벽에 닿거나, 몸통에 닿게 되면 게임이 종료됩니다.
- 머리가 아이템 좌표와 같아질 때 아이템을 얻습니다.

3. Item

- 아이템은 5초마다 벽이 아닌 곳에 랜덤하게 나타납니다.
- 아이템은 Snake, 벽과 겹치지 않게 생성합니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

- 아이템의 종류는 2가지로 화면상 초록색으로 보이는 아이템 "poison"과 노란색으로 보이는 "gift"가 구현되어 있습니다.
- 각각의 아이템은 Mapdata에서 poison은 '5' gift는 '6'로 정의합니다.
- Poison은 snake의 길이를 한 칸 줄이는 아이템이고 gift는 snake의 길이를 한 칸 늘립니다.
- 아이템 획득시 Map에서 사라집니다.

4. Gate

- Gate는 게임 시작순간부터 임의의(몇초) 시간 이후에 출현합니다.
- Gate는 각 모서리는 제외한 가장자리 벽에 생성됩니다.
- Gate는 몇초마다 임의의 위치에 랜덤하게 생성됩니다.
- Snake가 통과 중일 때는 Gate를 생성하지 않습니다.

5. Score

score

- score: giftScore * 10 + poisonScore * 10 + gateScore * 5로 최종 점수를 계산합니다.
- +: 현재까지 획득한 gift 수 입니다.
- -: 현재까지 획득한 poison 수 입니다.
- G: 현재까지 통과한 게이트 수 입니다.

mission

- Length: snake가 추가로 늘려야하는 snake 길이입니다.
- Gift: 획득해야할 gift 수입니다.
- Poison: 최대 허용 poison 수 입니다.
- Gate: 통과해야할 게이트 수 입니다.
- 모든 미션 완료시 다음 map으로 넘어갑니다.

6. 추가요소

- 추가적인 요소로 새로운 아이템 'biggift'를 추가시킵니다.
- biggift 아이템을 먹을 시 redzone(wall)장애물이 map에 생성됩니다
- biggift는 mapdata에서 '7'로 정의되며 gift와 posion과 같이 등장합니다.
- biggift는 화면상 보라색으로 보이는 아이템으로 구현하였고 gift와 같이 아이템을 먹을 시 추가 점수가 부여됩니다.
- 또 하나의 요소로 stage를 클리어 할 시 난이도 상승을 위해 snake의 이동속도를 빠르게 합니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

2.2 개발 내용 및 결과물

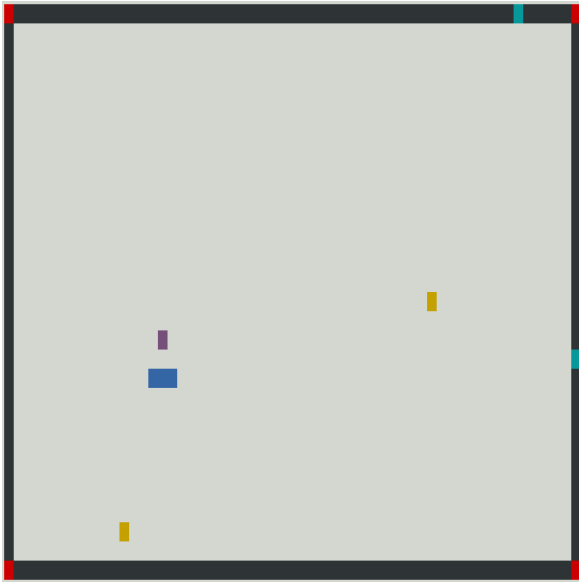
2.1.1 개발 내용

1. Map

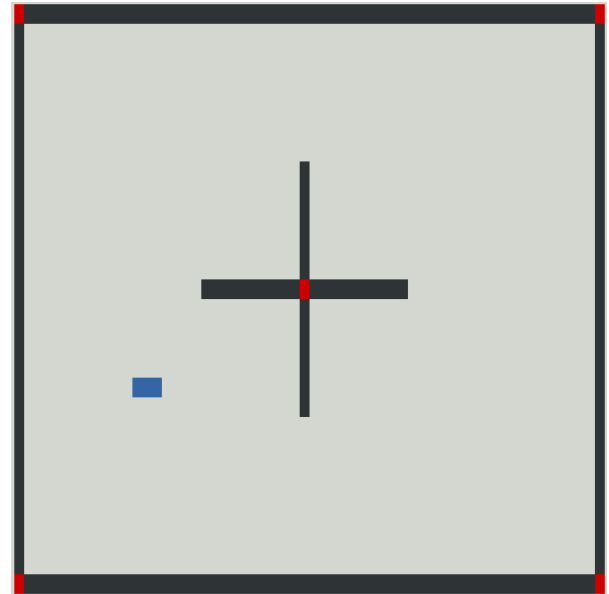
- Map 크기:62x32(width,height)의 txt파일로 map/ 디렉토리에 존재합니다. 디렉토리에 존재하는 map파일을 ifstream을 통해 읽어오게 되고 이를 2차원 배열인 data에 저장합니다. 각 stage별로 map의 모양이 바뀌는데 총 4단계로 이루어져 있습니다.
- Map의 구성요소들을 data에서 해당하는 char형 data로 저장하였는데 각각 빈공간은 '0' , 벽(wall)은 '1' , 접근불가벽(Immune Wall)은 '2', snake는 'x' , 각각의 아이템은 '5' , '6' , '7'로 마지막으로 gate는 '8'로 data에 정의되어 있습니다.
- 각 요소들은 data안에서 고유한 문자로 저장되어 있고 이를 구현하기 위해 main.cpp에서 반복을 통해 렌더링 합니다.
- 렌더링은 외부 라이브러리 함수인 NCRURSES를 통해 빈 공간은 회색, 벽(wall)은 검정색, 접근불가벽(Immune Wall)은 빨간색, snake는 파란색, poison은 초록색, gift는 노란색, biggift는 보라색, gate는 청록색으로 구현하였습니다.
- 이때 빈공간은 snake가 움직일 수 있는 공간을 의미합니다.
- 벽은 gate가 생성될 수 있는 벽으로 snake의 머리가 닿을 경우 게임 오버되는 조건을 가지고 있습니다.
- 접근 불가 벽(Immune Wall)은 gate가 생성되지 않는 벽으로 다른 게임요소와 상호 작용하지 않습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

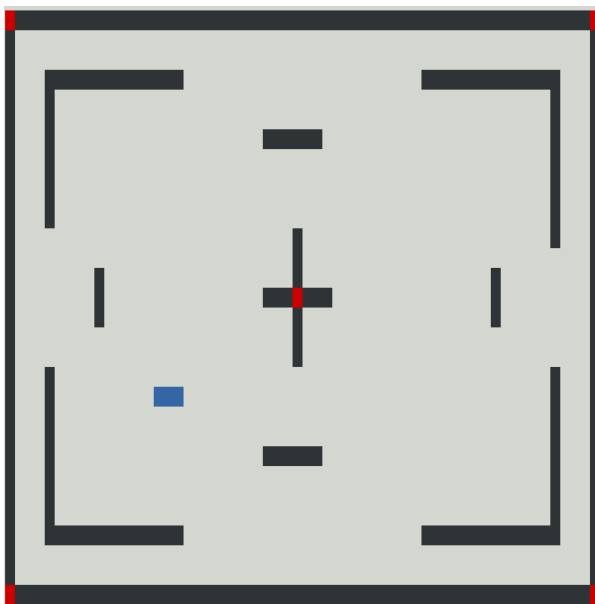
map1



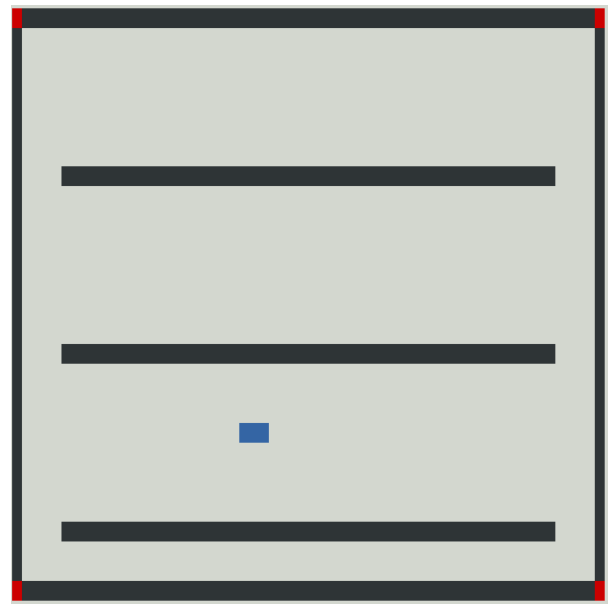
map2



map3



map4



 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

2. Snake

- snake의 몸체에 해당하는 snakepart는 std 라이브러리 벡터를 사용하였습니다.
- 스네이크의 위치는 (20,20)에서 시작하며 x축으로 길이가 3인 스네이크를 생성되도록 만들었고, 초기방향은 LEFT로 설정하였습니다.
- snake 조작에서 정반대 방향으로 키를 조작할 경우 스네이크가 죽도록 하였습니다.
- snake의 움직임 표현

방향키	좌표변화
상	행이 1만큼 감소
하	행이 1만큼 증가
좌	열이 1만큼 감소
우	열이 1만큼 증가

- insert를 사용하여 스네이크의 head 부분을 추가해주고 pop_back하여 맨 뒤 원소를 삭제합니다.
- snake의 머리에 해당하는 snake[0] 부분과 snakebody가 충돌할 경우, snake가 wall, immune wall과 충돌할 경우 false를 반환하여 충돌했음을 알립니다.
- snake의 길이가 3보다 작을 경우 게임이 종료됩니다.

3. Item

기본적인 아이템은 2가지로 poison과 gift가 있습니다.

(추가 구현으로 biggift 아이템이 있지만 이후 추가요소에서 서술하도록 하겠습니다.)

아이템의 위치는 rand 메소드를 통해 랜덤으로 좌표를 뽑아 만약 wall, snake등이 아닌 빈 공간일시 뽑은 3개의 좌표를 아이템을 관리하는 vector에 저장합니다.

아이템의 구성 또한 랜덤으로 결정되는데 rand메소드를 통해 구현하였습니다. 이렇게

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

뱀은 3개의 아이템은 pushdata 메소드를 통해 mapdata에 저장되는데 이때 item이 생성된 시간 dropitemtime을 저장합니다. 그리고 update를 반복함으로써 만약 현재 시간과 dropitemtime 시간이 임의의 시간을 초과할 경우 기존의 아이템을 deleteitem 메소드 item vector에서 삭제하고 다시 3개의 아이템을 생성하도록 합니다. 이러한 방식은 이후 추가요소에서 서술되는 biggift아이템 또한 동일하게 적용될 것입니다.

3.1.gift

mapfile에서 '5'의 문자로 정의 되어있고 이는 렌더링을 통해 노란색으로 화면에 출력됩니다.

snake가 gift를 획득하게 되면 movesnake 함수에서 gift변수가 참이 됩니다. snake의 update부분에서 gift가 참 일시 조건문으로 들어가게 되고 PushData 메소드에서 gift가 false일시 snake의 좌표요소를 담고 있는 vector의 마지막 요소를 삭제합니다. 만약 gift가 true일시 본 조건문을 실행하지 않기 때문에 길이가 한 칸 늘어나게 됩니다.

3.2 poison

mapfile에서 '6'의문자로 정의되었고 이는 렌더링을 통해 초록색으로 화면에 출력됩니다.

snake가 poison을 획득하게 되면 movesnake 함수에서 poison변수가 참이 됩니다. snake의 update메소드 부분에서 poison이 참 일시 snake의 좌표요소를 담고 있는 vector의 마지막 요소를 삭제하게 됩니다. 앞서 gift를 먹지 않았다면 snake의 길이가 유지되지만 posion을 획득함으로써 삭제가 두 번 일어나 결과적으로 snake의 길이가 한 칸 줄어들게 됩니다.

4. Gate

4.1 Gate 생성

- Snake game 시작 시부터 가장자리 wall 임의의 위치에 생성합니다.
- 10초마다 새로운 Gate를 생성합니다.
- Snake가 통과 중일 때는 Gate를 생성하지 않습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

- Wall에 Gate 한 쌍을 생성합니다.
- 가로 벽에서 Gate 하나, 세로 벽에서 Gate 하나가 생성되도록 합니다.
- Gate 의 개수를 세어서 생성시 마다 1씩 증가 하고 gate가 2가 될 때 까지만 생성합니다.

4.2 Gate 통과

- Snake의 head 좌표와 gate의 좌표가 같다면 gate를 통과중이라고 판단하고 gateflag를 true로 설정합니다.
- Gateflag가 true라면 통과한 Gate의 좌표값에 따라 direction을 바꿔줍니다.

Gate(In)	Direction
Row==1	RIGHT
Row==height--2	LEFT
Height==1	DOWN
Height==width-2	UP

- Gate 통과 횟수를 저장하는 변수를 1 증가시킵니다.

5. Score board

5.1 Score board

- 현재까지 snake가 획득한 점수들을 출력합니다.
- totalScore(총 점수), giftScore(획득한 gift item 수), poisonScore(획득한 poison item 수), gateScore(통과한 gate 횟수)로 이루어져있습니다.

5.2Mission board

- 각 map마다 요구하는 mission이 다르며 각 미션의 종류는 stage의 mission을 통해 변경가능합니다.
- 미션 종류에는 lengthScore(snake가 초기보다 늘려야하는 몸의 길이), giftScore, poisonScore, gateScore를 통해 수행되어지며 미션을 완수할 경우 'V'자를 표시하도록 설계하였습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

5.3 추가 내용

- 각 맵에 주어진 모든 미션을 완수할 경우 다음 스테이지로 넘어가도록 설정하였습니다.

5.4 실제 화면

- board 초기화면

```

      | S C O R E |
      -----
Score : 0
  + : 0
  - : 0
   G : 0

      | M I S S I O N |
      -----
Length : 0/1 ( )
  Gift : 0/6 ( )
Poison : 0/6 ( )
   Gate : 0/1 ( )

```

- score 갱신 및 mission 성공

```

      | S C O R E |
      -----
Score : 40
  + : 4
  - : 0
   G : 0

      | M I S S I O N |
      -----
Length : 4/1 (V)
  Gift : 4/6 ( )
Poison : 0/6 ( )
   Gate : 0/1 ( )

```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

6. 추가요소

게임규칙과 별도로 추가한 요소는 두가지로 biggift 아이템과 snake의 속도를 변경하였습니다.

6.1. biggift

- mapfile에서 '7'의 문자로 정의하였고 이는 렌더링을 통해 보라색으로 화면에 출력됩니다.
- snake가 'biggift'를 획득하게 되면 movesnake 함수에서 biggift의 변수가 참이 되게 됩니다. snake의 update 메소드 부분에서 biggift이 참 일시 map의 한 부분을 랜덤으로 추출하게 됩니다. 추출한 좌표에서 (y , x+1) , (y+1 , x) , (y+1 , x+1)의 좌표를 Wall로 정의되는 '1'로 변경함으로써 map에 추가적인 장애물(red zone)을 생성해줍니다.

6.2. sake의 속도

- snake의 속도는 gamescene::Update의 upsleep(delaytime) 메소드에 의해 결정됩니다. delaytime은 stage의 변수로 150000으로 초기화 되어있습니다. 만약 플레이어가 미션을 클리어하고 다음 스테이지로 넘어갈 시 stage->delaytime - 20000의 구문을 통해 delaytime이 줄게 되고 이로 인해 snake game의 전체적인 속도가 빨라지게 됩니다.

6.3. 추가요소의 의미

- snake의 속도와 'biggift'는 snake game의 전체적인 난이도 상승과 게임 속도 빠르게 만들어 게임의 재미를 높이기 위해 고안되었습니다. biggift를 통해 미션을 클리어하는 속도를 올리고 추가적인 장애물 요소로 인한 페널티 생성, snake 자체의 속도를 올림으로써 난이도를 높였습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

2.1.2 시스템 구조 및 설계도

적용단계	내용	소스 파일
1단계	Map의 구현	make_map.cpp , gamescene.cpp, stage.cpp, function.cpp
2단계	Snake 표현 및 조작	snake.cpp, item.cpp
3단계	Item 요소의 구현	1.item.cpp , gamescene.cpp 2. snake.cpp
4단계	Gate 요소의 구현	Item.cpp,snake.cpp
5단계	점수 요소의 구현	snake.cpp, score.h, menu.cpp, stage.h
6단계	추가요소	stage.cpp , gamescene.cpp , item , itemmanager

1. MAP(개발자: 한승진)

주요 메소드: MapManager::Load()

서브 메소드: function.cpp/ChangeScene()

게임의 3단계 중 게임의 실행부분은 gamescene단계입니다. gamescene에서는 각 stage의 미션을 클리어시 다음 stage로 넘어가기 위해 현재 stage를 보관하다가 미션을 클리어 한다면 새로운 GameScene을 function.cpp에 정의된 ChangeScene메소드를 사용하여 바꾸어 줍니다.

```
GameScene::GameScene()
{
    srand(time(NULL));
    me = new Player();
    mapManager = new MapManager();
    mapManager->Load();
}
```

새로운 GameScene이 선언되면서 mapManager.cpp안에 정의되어있는 Load함수가 "map/map + (현재 stage+ 1) + .txt"경로안에 있는 map파일을 불러옵니다. 외부 파일을 불러오는 방법은 ifstream을 통해 readFile.getLine(temp, 120), 120크기의 char형 상수를 받아 본 프로그램에서 mapfile처럼 사용할 char형 2차원 배열 data에 저장합니다. 저장된 mapdata를 gamescene의 Render 메소드를 통해 화

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

면상의 출력해 줍니다. 2차원 mapdata에 접근하여 저장되어있는 값을 switch , case문을 통해 해당 문자에 맞는 색을 출력해줍니다. 화면에 색을 출력하는 것은 NCURSES를 통해 구현하였는데 attron(COLOR_PAIR())을 통해 색상을 정해주고 mvprintw(i , j , " ")를 통해 이를 출력하였습니다.

이러한 과정은 미션을 클리어한 후 새로운 stage를 불러올때 반복되어 실행되고 총 4번의 mapfile을 불러오게 됩니다. 이와 별개로 게임을 시작하기전에 플레이어에게 대기화면을 보여주는 startscene과 게임오버될시 플레이어에게 재도전의 기회를 주는 gameoverscene또한 같은방식으로 동작하게 됩니다.

2. SNAKE(개발자:이민지)

struct snakepart

snake.h 파일에는 snakeclass 클래스와 snakepart의 구조체가 존재합니다. snakepart는 스네이크 몸체를 구성 및 좌표를 표시하기 위한 x,y 값을 담고 있습니다.


```
struct snakepart{
    int x, y;
    snakepart(int col, int row);
    snakepart();
};
```

class snakeclass

snakeclass이름을 가진 클래스에는 위에서 구조체를 선언한 snakepart를 벡터로 가지는 실제 snake를 생성하였으며 direction, movesnake(), collision()등의 여러 변수 및 함수를 포함하고 있습니다.

```
class snakeclass{
    //점수 : points, 시간 : del
    int maxheight;
    int maxwidth;
    char direction;
    std::vector<snakepart> snake;

public:
    bool gift;
    bool poison;
```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

```

bool biggift;

bool isDied;
snakeclass();
~snakeclass();
void Update();
bool collision();
void movesnake(float eTime);
void PushData();

};

```

snakeclass::snakeclass()

초기 snake 생성 및 방향입니다. snake는 (20,20)을 시작으로 하여 x축 방향으로 20,21,22의 길이가 3인 벡터 snake를 생성합니다. 이때 PatchData에 char인 'x'를 추가하여 ncurses 상에서 스네이크에 해당하는 좌표를 표현하도록 합니다. 초기방향은 left로 설정합니다.

```

snakeclass::snakeclass()
{

    getmaxyx(stdscr, maxheight, maxwidth);
    for(int i=0; i<3; i++){
        snake.push_back(snakepart(20+i, 20));
    }
    gift = false;
    poison = false;
    biggift = false;

    //초기방향 : left
    direction = 'l';
    srand(time(0));
    for(int i = 0; i<snake.size(); i++){
        mapManager->PatchData(snake[i].y , snake[i].x , 'x');
    }
}

```

snakeclass::collision()

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

collision()은 충돌여부를 판단하는 함수입니다. snake의 좌표값이 wall을 의미하는 1, immune wall을 의미하는 2랑 같게 된다면 스네이크는 빈공간이 아닌 벽에 부딪혔다는 의미입니다. 또한 스네이크 몸체에 스네이크 머리가 닿는 경우도 발생합니다. 이 때 true를 반환하여 충돌을 했음을 보이고 위와 같은 경우가 아닐 시 false를 반환합니다.

```
bool snakeclass::collision() // get item? or get die?
{
    if (mapManager->data[snake[0].y][snake[0].x] == '1'){ return true;}
    if (mapManager->data[snake[0].y][snake[0].x] == '2'){ return true;}
    //snakebody에 snakehead 닿을 시 충돌
    for(int i =2; i<snake.size(); i++){
        if(snake[0].x == snake[i].x && snake[0].y == snake[i].y)
            return true;
    }
    return false;
}
```

snakeclass::movesnake(float eTime)

movesnake()는 실제로 게임이 진행시 방향키에 따라 스네이크가 움직이게 되는 실질적인 스네이크 알고리즘 부분입니다. 방향키를 조작 시 현재 방향(초기방향은 left)을 기준으로 반대방향으로 조작하게 될 경우 isDied라는 변수가 true가 되어 snake가 죽게 됩니다. isDied가 false 일 경우 조작한 방향에 따라 벡터 snake에 해당되는 좌표값을 추가해줍니다. 예를 들어 방향이 Left 일 경우 스네이크 head에 해당하는 snake[0].x, snake[0].y 좌표중 x좌표의 왼쪽에 스네이크 좌표를 추가해줍니다.

```
void snakeclass::movesnake(float eTime)
{
    int tmp;
    tmp = getch();
    switch(tmp)
    {
        case KEY_LEFT:
            if(direction!='r')
                direction = 'l';
            else isDied = true;
            break;
        case KEY_UP:
```



```
        if(direction!='d')
            direction = 'u';
        else isDied = true;
        break;
    case KEY_DOWN:
        if(direction!='u')
            direction = 'd';
        else isDied = true;
        break;
    case KEY_RIGHT:
        if(direction!='l')
            direction = 'r';
        else isDied = true;
        break;
    }
    if(!isDied)
    {
        if(direction=='l')
            snake.insert(snake.begin(), snakepart(snake[0].x-1, snake[0].y));
        else if(direction=='r')
            snake.insert(snake.begin(), snakepart(snake[0].x+1, snake[0].y));
        else if(direction=='u')
            snake.insert(snake.begin(), snakepart(snake[0].x, snake[0].y-1));
        else if(direction=='d')
            snake.insert(snake.begin(), snakepart(snake[0].x, snake[0].y+1));
    }
    if(collision() == true){isDied = true;}
```

snakeclass::PushData()

PushData() 함수입니다. Snake가 각 아이템과 충돌했다면 gift, poison, biggift를 true로 설정합니다. movesnake()에서 snake head의 좌표가 insert를 통해 추가되었으니 매번 snake의 마지막 좌표(꼬리)를 pop_back을 통해 없애줘야합니다. 마지막 좌표를 없애지 않아도


 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

되는 경우는 gift 혹은 biggift를 먹어 snake의 길이를 늘려줘야 할 경우이며 이 경우는 통합적으로 (big) gift item 섭취시 gift를 true로 반환해주었기 때문에 pop_back을 실행하지 않습니다. poison 아이템을 섭취했을 경우엔 기존의 길이 유지를 위한 pop_back 함수 실행 및 아이템 섭취로 인한 pop_back 실행으로 함수가 총 두 번 호출되어야 합니다. 추가 구현한 biggift 부분입니다. biggift 아이템을 섭취하게 되면 맵 상에서 2 x 2 크기의 벽이 생성되도록 설정하였습니다. 따라서 이 벽에 부딪힐 경우 게임이 종료됩니다. 또한 snake의 길이가 3보다 작아질 경우 isDied 가 true를 반환하여 게임이 종료되도록 설정하였으며 스네이크가 죽지 않는 한 PatchData를 계속하여 호출하여 mapdatadml 의 char 값을 'x'로 변경하여 게임을 진행시킵니다.

```
void snakeclass::PushData() {
    if(!gift)//add len
    {
        mapManager->PatchData(snake[snake.size()-1].y, snake[snake.size()-1].x, '0');
        snake.pop_back();
    }

    if(poison)
    {
        mapManager->PatchData(snake[snake.size()-1].y, snake[snake.size()-1].x, '0');
        snake.pop_back();
    }

    if(biggift)
    {
        while(1){
            int x = rand() % WIDTH;
            int y = rand() % HEIGHT;
            if(mapManager->data[y][x] == '0'){
                mapManager->PatchData(y, x, '1');
                mapManager->PatchData(y-1, x, '1');
                mapManager->PatchData(y, x+1, '1');
                mapManager->PatchData(y-1, x+1, '1');
                break;
            }
        }
    }
}
```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

```

    }
}

if(snake.size()<3) isDied = true;
if(!isDied)
    mapManager->PatchData(snake[0].y , snake[0].x , 'x');
}

```

3. ITEM(개발자: 한승진)

주요 메소드: item::getrandpos() , itemManager::istimeover() , itemManager::deleteitem , itemManager::itemtype() , itemManager::pushdata() , itemManager::Update();

서브 메소드: snake::PushData() , snake::movesnake

주요 클래스: itemManager , item , snakeclases

item::getrandpos()

item을 저장하기 위한 vector를 하나 생성합니다. 이때 vector가 담고 있는 것은 item class인데 item class는 myposition의 좌표요소와 그 type을 지니고 있습니다. 좌표요소를 결정하는 것은 item::getrandpos에 의해 정해지는데 $\text{int } x = \text{rand}() \% \text{WIDTH};$ 그리고 $\text{int } y = \text{rand}() \% \text{HEIGHT}$ 을통해 임의의 값을 가져오게 됩니다. 이때 불러온 값이 $\text{data}[y][x] == '0'$ 일 경우 즉 빈 공간일 경우에만 itemvector에 저장합니다.

itemManager::Update()

update는 반복을 통해 변경된 요소들을 즉각적으로 mapdata에 구현되어집니다. 아이템의 type은 $\text{rand}() \% 3$ 을 통해 3가지의 경우로 나누고 이에 poison , gift , biggfit로 나뉘게 됩니다. 나뉘어진 아이템들은 itemvector로 저장되고 저장된 벡터의 좌표를 pushdata를 통해 실제 mapfile에 저장합니다. 이때 아이템들이 생성된 시간을 lastDropTime에 저장해 놓음으로써 아 아이템이 출현시간을 조정할 수 있습니다.

itemManager::itemtype()

update에서 결정된 타입에 따라 $\text{data.push_back}(\text{item}(\text{"type"}));$ 을 통해 itemvector에 저장합니다. 이때 item 타입에 item의 타입을 인자로 주어 random한 좌표를 지니는 item객체가 완성됩니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

itemManager::istimeover()

update에서 아이템의 발생시간을 lastDropTime에 저장해놓고 Update메소드 안에서 이를 $if(eTime - lastDropTime > time)$ (eTime은 현재시간)과 같이 비교함으로써 아이템이 생성된 아이템의 출현시간이 얼마나 됐는지 알 수 있습니다.

itemManager::pushdata()

itemvector에 저장되어있는 item 타입을 실제로 mapdata에 저장하는 메소드입니다.

itemManager::deleteitem()

snake가 item을 획득하게 된다면 map상에서 해당 아이템은 사라져야 합니다. snake가 지나감으로써 mapdata에서의 값은 snake를 의미하는 'x'로 바뀌게 되고 또한 itemvector의 해당요소를 삭제해줘야 함으로 `data.erase(data.begin() + target)` 을 통해 해당 요소를 제거할 수 있습니다. 이때 target은 해당하는 요소의 위치를 의미합니다.

snakeclass::movesnake()

movesnake메소드는 snake의 헤드가 플레이어가 누르는 방향으로 이동함을 구현한 것입니다. 이때 움직인 snake의 head가 특정 아이템을 획득 `mapManager->data[snake[0].y][snake[0].x] == 'item'` 과 같은 상태가 됐을 경우 각 item에 맞는 조건 변수가 true로 변경됩니다.

snakeclass::PushData()


pushdata에서는 snake의 상태에 따른 변화를 mapdata에 저장합니다. 이때 아이템의 조건 변수가 참일 경우 해당 아이템이 지니는 효과를 받습니다.

gift

`mapManager->PatchData(snake[snake.size()-1].y, snake[snake.size()-1].x, '0');`
`snake.pop_back();` 과 같이 gift 아이템을 먹을경우 본 조건문을 실행하지 않게 되므로 길이가 한 칸 늘어나게 됩니다.

poison

`mapManager->PatchData(snake[snake.size()-1].y, snake[snake.size()-1].x, '0');`
`snake.pop_back();` 만약 poison을 획득한다면 snake vector의 꼬리 즉 마지막요소를 삭제해줍니다. 이는 만약 gift를 먹지 않는다면 두번 실행되게 되므로 결과적으로 snake의 길이가 한 칸 줄어들게 됩니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

4. GATE (개발자 : 채지윤)

itemManager::MakeGate()

wall인 지점에 세로 gate하나 가로 gate하나 생성합니다. Gate는 10초마다 랜덤으로 생성되며 Gate는 '8'로 지정하여 wall과 구분해줍니다.DeleteGate()에서 gate를 삭제시마다 0으로 만들어 gate는 0일때만 생성하여 여러개의 gate가 동시에 생성되는 걸 방지해줍니다.

```
void itemManager::MakeGate()
{
    if (Gatecnt==0)
    {
        Gate1.y=rand()%HEIGHT;
        if (Gate1.y>(HEIGHT/2)) Gate1.y=HEIGHT-2;
        else Gate1.y=1;
        Gate1.x=rand()%(WIDTH-4)+2;

        Gate2.y=rand()%(HEIGHT-4)+2;
        Gate2.x=rand()%WIDTH;
        if (Gate2.x>(WIDTH/2)) Gate2.x=WIDTH-2;
        else Gate2.x=1;


        mapManager->PatchData(Gate1.y ,Gate1.x , '8');
        Gatecnt++;
        mapManager->PatchData(Gate2.y ,Gate2.x , '8');
        Gatecnt++;
    }
}
```

itemManager::DeleteGate()

함수 실행시 Gatecnt가 2일 때 gate를 삭제합니다. Gate가 다시 Wall로 바뀌는 것이기 때문에 Wall을 뜻하는 '1'로 지정해줍니다. Gate 삭제시마다 Gatecnt를 1씩 감소시켜 해당 변수를 참조하여 MakeGate()에서 이용할 수 있게 만들어줍니다.

```
void itemManager::DeleteGate()
{

```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

```

if (Gatecnt==2)
{
    mapManager->PatchData (Gate1.y ,Gate1.x , '1');
    Gatecnt--;

    mapManager->PatchData (Gate2.y ,Gate2.x , '1');
    Gatecnt--;
}
}

```

4.1 Gate통과시

Snake가 하나의 Gate에 통과중일 경우 Snake 헤드부분 좌표를 다른 Gate좌표로 지정하여 다른 Gate로 출현하도록 합니다. Snake의 헤드 좌표가 Gate와 동일한 경우 해당 Gate에 들어왔다고 판단하고 다른 헤드 좌표를 다른 Gate 좌표로 바꿔줍니다. Gateflag를 이용하여 Snake가 Gate를 통과중이라는 것을 표시해줍니다.

```

if (mapManager->data[snake[0].y][snake[0].x] == '8'){
    gateflag=true; //gate통과중 표시 ->게이트 생성 정지
    me->SetGateScore (me->gateScore + 1);
    if (snake[0].y == Gate1.y && snake[0].x == Gate1.x){

        if (Gate2.x == 1){
            direction = 'r';
            snake[0].x = Gate2.x + 1;
            snake[0].y = Gate2.y;
            mapManager->PatchData (snake[0].y , snake[0].x ,
            'x');
        }
        else{
            direction = 'l';
            snake[0].x = Gate2.x - 1;
            snake[0].y = Gate2.y;
            mapManager->PatchData (snake[0].y , snake[0].x ,
            'x');
        }
    }
}

//gate2 진입시

```



```
else if (snake[0].y == Gate2.y && snake[0].x == Gate2.x) {  
    if (Gate1.y == 1) {  
        direction = 'd';  
        snake[0].x = Gate1.x;  
        snake[0].y = Gate1.y + 1;  
        mapManager->PatchData(snake[0].y, snake[0].x, 'x');  
    }  
    else {  
        direction = 'u';  
        snake[0].x = Gate1.x;  
        snake[0].y = Gate1.y - 1;  
        mapManager->PatchData(snake[0].y, snake[0].x, 'x');  
    }  
}  
gateflag=false;  
}
```

5. SCORE BOARD (개발자:이민지)

5.1 score.h

Player라는 클래스를 생성합니다. 이 클래스에는 스네이크가 게임을 진행하면서 얻은 스네이크 길이, 각 아이템별 점수, 게이트 통과 점수, 전체 점수에 관련된 변수를 가지고 있습니다. 관련된 각 변수마다 set 함수를 만들어 여러 파일에서 사용가능하도록 확장성을 높였습니다. totalscore를 얻는 방식은 giftScore * 10 + poisonScore * 10 + gateSCORE * 5 로 설정하였습니다.

```
class Player  
{  
public:
```




```
int lengthScore = 0;
int giftScore = 0;
int poisonScore = 0;
int gateScore = 0;
int totalScore = 0;

Player()
{
    lengthScore = 0;
    giftScore = 0;
    poisonScore = 0;
    gateScore = 0;
    totalScore = 0;
}
~Player() {}
void SetLengthScore(int value)
{
    lengthScore = value;
}
void SetGiftScore(int value)
{
    giftScore = value;
}
void SetPoisonScore(int value)
{
    poisonScore = value;
}
void SetGateScore(int value)
{
    gateScore = value;
}
void SetTotalScore(int value)
{
    totalScore = giftScore * 10 - poisonScore * 10 + gateScore
* 5 ;
}
};
```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17


우선 snake.cpp 파일에서 snakeclass::movesnake() 함수에서 각 아이템을 섭취하게 될 경우 각 아이템에 해당되는 Score를 갱신해주었습니다.

```

if(mapManager->data[snake[0].y][snake[0].x] == '5'){
    itemmanager->deleteitem(snake[0].y , snake[0].x);
    me->SetPoisonScore(me->poisonScore + 1);
    me->SetLengthScore(me->lengthScore - 1);
    poison = true;
}
else{
    poison = false;
}
if(mapManager->data[snake[0].y][snake[0].x] == '6'){
    itemmanager->deleteitem(snake[0].y , snake[0].x);
    me->SetGiftScore(me->giftScore + 1);
    me->SetLengthScore(me->lengthScore + 1);
    gift = true;
}
else{
    gift = false;
}
if(mapManager->data[snake[0].y][snake[0].x] == '7'){
    itemmanager->deleteitem(snake[0].y , snake[0].x);
    me->SetGiftScore(me->giftScore + 1);
    me->SetLengthScore(me->lengthScore + 1);
    biggift = true;
    gift = true;
}
else{
    biggift = false;
}

```

이 후 menu.cpp 에서 MENU::DrawScore(), MENU::DrawMission()을 통해 ncurses 화면에 점수를 출력하도록 하였습니다. 화면에 보여질 점수판은 실제 게임이 작동되는 화면이 보여지는 왼쪽이 아닌 오른쪽 화면에 보이도록 하였습니다. int로 받은 각 점수를 문자열로 변환하여 화면상에 보이도록 설정해 두었습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

MENU::DrawScore()

|SCORE| 에는 totalScore, giftScore, poisonScore, gateScore 총 네개의 점수가 기록될 것입니다.

```
void MENU::DrawScore()
{


    move(7, maxwidth / 5*4 -3 );
    printf("| S C O R E |");

    for (int i = 0; i < 26; i++)
    {
        move(8, maxwidth / 5 * 4 -9 + i);
        addch('-');
    }

    int totalScore = me->totalScore;
    string totalScore_str = to_string(totalScore);
    move(10, maxwidth / 5 * 4 -3);
    printf("Score : ");
    move(10 , maxwidth / 5 * 4 +6);
    printf(totalScore_str.c_str());

    int giftScore = me->giftScore;
    string giftScore_str = to_string(giftScore);
    move(12, maxwidth / 5 * 4 +1);
    printf("+ : ");
    move(12 , maxwidth / 5 * 4 +6);
    printf(giftScore_str.c_str());

    int poisonScore = me->poisonScore;
    string poisonScore_str = to_string(poisonScore);
    move(14, maxwidth / 5 * 4 +1);
    printf("- : ");
    move(14 , maxwidth / 5 * 4 +6);
    printf(poisonScore_str.c_str());
```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

```

int gateScore = me->gateScore;
string gateScore_str = to_string(gateScore);
move(16, maxwidth / 5 * 4 +1);
printw("G : ");
move(16 , maxwidth / 5 * 4 +6);
printw(gateScore_str.c_str());

}

```

MENU::DrawMission

MENU::DrawMission에서는 stage.h에 미리 만들어 둔 미션 기준을 호출합니다. 각 스테이지 별 미션 기준을 모두 완료하는 것을 미션으로 두고 있습니다.

```

void MENU::DrawMission()
{
    int *nowMission = stage->getNowMission();

    move(19, maxwidth / 5 * 4 -5);
    printw("| M I S S I O N |");

    for (int i = 0; i < 26; i++)
    {
        move(20, maxwidth / 5 * 4 - 9 + i);
        addch('-');
    }

    move(22, maxwidth / 5 * 4 -4);
    printw("Length   : %d/%d   (%c)", me->lengthScore, nowMission[0],
Complete(me->lengthScore, nowMission[0]));

    move(24, maxwidth / 5 * 4 -2);
    printw("Gift      : %d/%d   (%c)", me->giftScore, nowMission[1],
Complete(me->giftScore, nowMission[1]));

    move(26, maxwidth / 5 * 4 -4);
    printw("Poison    : %d/%d   (%c)", me->poisonScore, nowMission[2],

```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

```
Complete(me->poisonScore, nowMission[2]));

    move(28, maxwidth / 5 * 4 -2);
    printf("Gate    :   %d/%d    (%c)",    me->gateScore,    nowMission[3],
Complete(me->gateScore, nowMission[3]));

}
```

MENU::Complete(int present, int goal)

미션을 완수하였는지 안하였는지를 판별하기 위해 Complete(int present, int goal)함수를 만들었습니다. 현재 점수와 미션 점수를 비교하여 미션을 완수하였을 경우 'V'를 표시되도록 하여 미션을 완수하였음을 알리도록 하였습니다.

```
char MENU::Complete(int present, int goal)
{
    if (present >= goal)
        return 'V';
    else
        return ' ';
}
```

제작된 맵은 총 4개입니다. 각 맵에서 요구되어지는 미션을 모두 충족하였을 경우 게임은 성공하였고 다음 수준의 맵으로 이동해야 합니다. 이를 위해 clear라는 bool 타입 변수를 생성하였습니다. 이 scoreboard판도 game 과 마찬가지로 update와 render 작업이 반복되어지는데 각 맵에서의 주어진 미션들을 모두 완수하였을 경우 clear는 true가 되어집니다. 이 clear는 다른 헤더파일인 gamescene의 Update 함수에서 다음 stage로 이동 시 사용되어질 것입니다.

6. PLUS ITEM (개발자:한승진)

주요 메소드: item::getrandpos() , itemManager::istimeover() , itemManager::deleteitem , itemManager::itemtype() , itemManager::pushdata() , itemManager::Update()

서브 메소드: snake::PushData() , snake::movesnake

주요 클래스: itemManager , item , snakeclases , stage

추가요소는 snakegame의 난이도를 결정하는 요소로 이루어져 map에 장애물을 생성하고 snake의 길이를 늘리는 'biggift'아이템과 stage를 클리어 할 시 snake game의 전체적인 속도를 증가시키는 게임적 장치로 이루어져 있습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

biggift

biggift를 구현하는 메소드와 클래스는 위에서 서술한 item과 거의 유사하므로 biggift의 획득 효과만을 작성하겠습니다.

biggift를 snake의 헤드가 획득할 시 getrandpos메소드를 통해 좌표를 랜덤하게 얻게 됩니다. 그렇게 얻은 좌표가 빈공간 즉 data[y][x] == '0' 일 경우

```
mapManager->PatchData(y ,x , '1');
```

```
mapManager->PatchData(y-1 ,x , '1');
```

```
mapManager->PatchData(y ,x+1 , '1');
```

```
mapManager->PatchData(y-1 ,x+1 , '1');
```

위와 같이 해당 좌표에 2x2크기의 장애물을 형성하게 됩니다. 이때 gift의 변수 또한 같이 참이 되므로 길이가 늘어나게 되어집니다.

stage::delaytime , gamescene

delaytime은 stage클래스에 정의되어 있는 변수로 초기값은 150000입니다.

if (menu->clear) stage에서 주어진 미션을 클리어하게되면 본 조건이 참이 되게되는데 이때 stage -> delaytime -= 20000을 통해 delaytime을 줄이게 됩니다. gamescene::Update메소드는 while문을 통해 반복하는데 usleep(stage->delaytime)을 통해 딜레이 시간을 가지게 됩니다. 따라서 이 delaytime을 줄임으로써 snakegame의 속도가 빨라지게 됩니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

2.1.3 활용/개발된 기술

1. ncurses

snakegame 윈도우를 생성하고 snake/wall/item/gate 등 색깔을 만들기 위해 해당 라이브러리를 사용하였습니다. initscr()메소드를 통해 ncurses기반 모듈을 실행시킵니다. 이후 attron과 wbkgd와 같은 색상 관련 메소드들을 통해 배경화면을 흰색으로 설정하고 게임의 다른 요소들 또한 각각의 색상을 부여하였습니다. GUI 기반 게임을 만들기 위해 사용하였습니다.

2. cstdlib

rand()함수를 호출하여 랜덤한 값을 주기위해 사용하였습니다.

3. ctime , chrono

ctime과 chrono 시간과 관련된 라이브러리입니다. 특히 chrono는 c++에서만 지원되며 ctime보다 다양하고 편리한 기능이 구현되어 있습니다.

```
std::chrono::system_clock::time_point start = std::chrono::system_clock::now();
```

와 같은 구문을 통해 현재시간을 저장할 수 있습니다.

본 게임에서는 gamescene의 Update 메소드에서 계속해서 현재시간을 받아와 필요한 지속시간이 존재하는 게임요소에서 사용되어집니다. 현재시간은 eTime으로 표현되며 시간과 관련된 게임요소들은 item과 gate등이 있습니다. 위에서 서술 했듯이 현재시간 - 요소의 생성시간을 계산함으로써 지속시간을 정해 줄 수 있습니다.

4. vector

vector는 STL라이브러리중 기본적인 컨테이너로 snake, item이라는 객체를 저장하기 위해 사용했습니다. snake의 head에 접근하는 방법은 snake[0]을 통해 접근하고 insert를 통해 요소를 추가하였습니다. 요소의 삭제는 snake.pop_back()를 사용하였습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17


2.1.4 현실적 제한 요소 및 그 해결 방안

1. 아이템과 게이트를 구현하는데 있어 출현시간에 관해 어떻게 초단위를 계산하여 생성과 소멸과정을 구현해야할지 잘 떠오르지 않았습니다. 이에 관해 공부하던 도중 c++에서 제공하는 chrono 라이브러리를 알게 되었고 이를 사용하여 구현하게 되었습니다. gamescene의 Update메소드가 반복 되면서 현재시간을 받아오게 됩니다. 이는 getElapsedTime 메소드를 통해 구현되어 있습니다. 이때 gate와 아이템을 생성하면서 droptime이라는 변수를 만들어 생성될 때의 시간을 저장해두고 update의 현재시간을 이와 비교하여 시간이 ~초만큼 지남을 구현 할 수 있었습니다.
2. 5단계의 구현에서 주어진 mission을 클리어 한 후 다음 stage로 넘어가는 과정에서 snake가 전단계의 위치에 그대로 있으면 새로 불러온 mapfile에 의하여 바로 게임오버되는 한계가 있었습니다. 이를 해결하기위해 mission을 클리어하게 되면 그 다음 stage를 불러내는 과정에서 item , snake , mission등 모든 게임요소들을 다시 생성하였습니다. 따라서 snake가 시작하는 장소를 지정해두고 map을 만들 때 snake가 시작하는 장소는 장애물을 지정하지 않음으로써 이를 구현할 수 있었습니다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

2.1.5 결과물 목록

파일명	역할
Makefile	컴파일을 하기위한 makefile (기여자: 한승진)
make_map.h	MapManager클래스 선언 (기여자: 한승진)
make_map.cpp	mapdata관련 메소드 구현 ex) Load , GetData (기여자: 한승진)
gameoverscene.h	GameOver 클래스 선언 플레이어의 input을 받아 플레이어가 'y'를 누를시 게임이 재시작되고 'n'을 누를시 게임이 종료된다. (기여자: 한승진)
gameoverscene.cpp	gameover시 재도전을 위한 GameOver클래스 구현 (기여자: 한승진)
gamescene.h	게임이 실행되는 단계를 구현하는 gamascene클래스 선언 (주 기여자: 한승진 50% 부 기여자 :채지윤 25% 부 기여자: 25%)
gamescene.cpp	gamescene 클래스 구현 및 렌더링 (주 기여자: 한승진 50% 부 기여자 :채지윤 25% 부 기여자: 이민지 25%)
menu.h	menu class 선언 (기여자: 이민지)
menu.cpp	점수 및 미션 현황에 관련된 점수판 구현 (기여자: 이민지)
snake.h	snake class 선언 (주 기여자: 이민지 60% 부 기여자 :채지윤 20% 부 기여자: 한승진 20%)
snake.cpp	snake의 움직임에 관한 함수 구현 (주 기여자: 이민지 60% 부 기여자 :채지윤 20% 부 기여자: 한승진 20%)
item.h	item과 gate 생성을 위한 class가 선언되어 있다.


 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C 팀	
	Confidential Restricted	Version 1.2	2021-06-17

	(주 기여자: 한승진 60% 부 기여자 :채지윤 20% 부 기여자: 이민지 20%)
item.cpp	item과 gate 함수 구현 (주 기여자: 한승진 60% 부 기여자 :채지윤 20% 부 기여자: 이민지 20%)
stage.h	현재 단계를 나타내는 stage클래스 선언 및 미션 정의 (기여자: 한승진)
stage.cpp	stage 클래스 구현 및 게임속도에 관련있는 delaytime 정의 (주 기여자: 한승진 80% 부 기여자: 이민지 20%)
startscene.h	게임이 시작하기전 대기화면을 구현하는 startscene클래스 선언 (기여자: 한승진)
startscene.cpp	플레이어의 input을 받아 플레이어가 'y'를 누를시 게임이 시작되고 'n'을 누를시 게임이 종료된다. (기여자: 한승진)
function.h	snake게임에 있어 기반이 되는 함수들을 선언 (기여자: 한승진)
function.cpp	각 단계에 맞는 Update와 Render가 이루어지고 현재시간을 받아오는 GetElapsedTime이 구현되어있다. (기여자: 한승진)
main.cpp	main 함수 호출 (기여자: 한승진)
score.h	Player class 선언 (각 점수 및 확장성을 위한 set함수 생성) (기여자: 이민지)
myposition.h	NCURSES를 편리하게 이용하기위해 만든 좌표 클래스이다 (기여자: 한승진)

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

3 자기평가

이름	자기평가
이민지	snake 및 scoreboard를 담당해서 작업하였다. 혼자서 프로젝트를 진행한 것이 아닌 협업이다보니 내가 작성한 코드와 팀원들이 작성한 코드를 합치는 것이 가장 어려웠다. 내가 작성한 알고리즘 뿐만 아닌 모든 코드의 알고리즘을 이해해야만 프로젝트를 진행할 수 있었고 다른 파트 부분을 위해 전체적으로 변수, 함수를 확장성 있게 작성해야했다. 다들 깃허브를 활용하는 것이 미숙하다 보니 pull request 등으로 서로의 코드를 확인, 공유하는 것이 아니라 메신저를 통해 서로의 상황을 공유하고 설명하는 등 불편한 점도 있었다. 다음부터는 github를 좀 더 잘 활용하여 프로젝트를 진행하면 좋을 것 같다는 생각이 들었다. 프로젝트를 직접 진행해보니 여러 클래스 생성 및 상속, 헤더 파일로 분할 등을 적용해볼 수 있어서 실제 실력이 향상됨을 느꼈다.
채지윤	프로젝트의 게이트를 담당하였다. 프로젝트를 하면서 다른 팀원이 작성한 코드들을 설명없이 이해하기가 어려웠다. 그리고 코드작성시 코드와 변수명 같은걸 맞춰서 해야하고 또 다른 파트의 코드도 다 이해하고 있어야 추가적인 기능을 위한 코드 작성이 가능하다는 것을 알았다. 프로젝트를 하면서 직접 여러 헤더파일과 클래스의 상속 등을 실제로 적용해보면서 실습,과제와는 또 다르게 상속의 중요성에 대해서 깨닫게 되었다. 프로젝트를 하며 실력이 많이 늘었다고 생각이 들지만, c++을 어려워하는 나에겐 과제,실습,프로젝트를 모두 단기간에 하기가 좀 버거웠다.
한승진	처음으로 맡게된 협업 프로젝트에서 다른 팀원들과 어떻게 하면 원하는 목표를 이룰 수 있을지 많이 고민하였다. map을 만드는 1단계, item을 생성하는 3단계, 마지막으로 6단계의 추가요소를 구현하였다. 코드를 구현하면서 이번 프로젝트는 나에게 있어 처음으로 어느정도 규모가 있는 프로젝트 였는데 각각의 요소들의 소스파일들과 헤더파일들이 꼬이지 않게 설계를 하는 것이 중요한 단계임을 알게되었다. 또한 프로젝트에서 수정사항을 git을 통해 팀원들이 수정사항을 공유하게 만드는 것이 아닌 직접 공지하는 방식이었기에 놓치는 부분들이 생겼다. 이러한 부분으로 인해 팀원들이 같은 작업을 반복하게 만드는 결과를 만들었기에 개선이 필요하다 생각한다. 하지만 직접 수정사항을 알리고 코드에 대해 설명하는 과정에서 팀원들에게 더욱 디테일한 부분들을 설명 해줄 수 있어 소통이 원활하게 이루어지고 이로 인해 프로젝트를 완수할 수 있었다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

4 참고 문헌

번호	종류	제목	출처	발행년 도	저자	기타
1	웹사이트	NCURSES Programming HOWTO	https://ko.wikipedia.org/wiki/Ncurses	2010. 08.12	gsong	ncurses관련 함수 검색
2	웹사이트	GNU make - 규칙내 명령 작성(Writing the Commands in Rules)	http://korea.gnu.org/manual/release/make/make-sjp/make-ko_5.html	2000. 05.09	선정필	make관련 함수 검색

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

5 부록

5.1 사용자 매뉴얼

<게임 시작하기>

1. 자신이 새롭게 만든 폴더에 git clone을 하여 코드를 복제합니다.
2. cd snakegame_ncurses를 통해 실질적으로 makefile이 있는 공간으로 이동합니다.
3. make 명령어를 이용하여 소스파일들을 컴파일 시켜줍니다.
4. 이후 snakegame이 자동으로 실행됩니다.

```

→ Desktop git:(main) * cd Snake_game
→ Snake_game git:(main) * make
g++ src/snake.cpp src/gameoverscene.cpp src/gamescene.cpp src/stage.cpp src/make_map.cpp
src/item.cpp src/menu.cpp src/startscene.cpp src/function.cpp src/main.cpp -lncurses -o
tmp/a.out && /tmp/a.out
In file included from src/snake.cpp:8:

```

<게임 시작화면 >

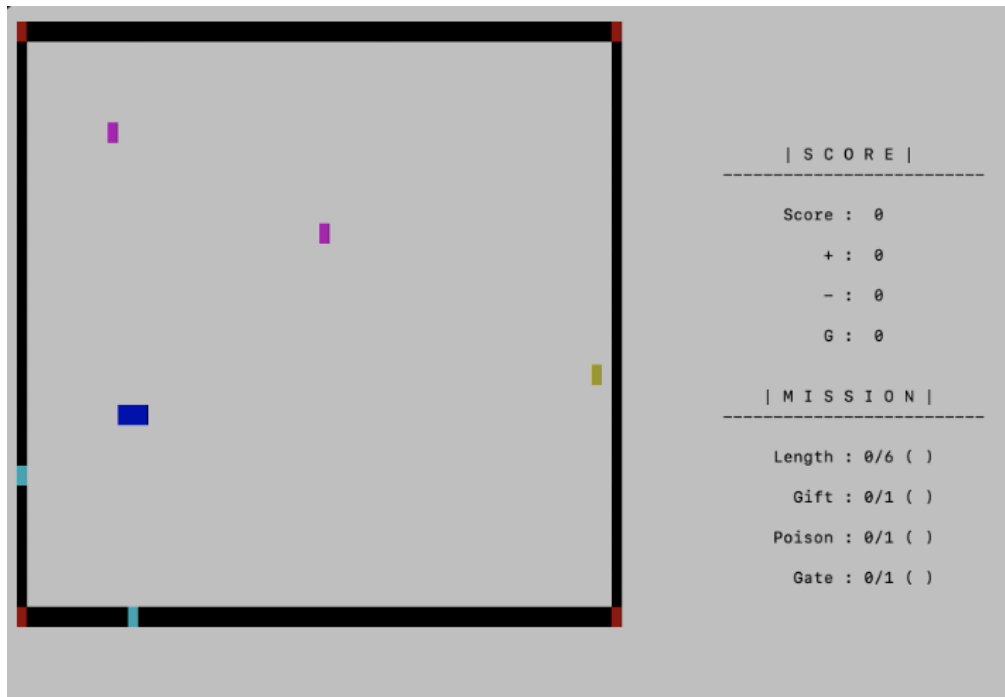
게임 시작화면에서 게임을 하고 싶다면 Y(y) 게임을 하고 싶지 않다면 N(n)을 누릅니다.




 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

<게임 실행시 화면>

우측에 현재 게임 점수를 표시하는 화면과 미션이 나타난다. 해당 미션을 클리어하면 자동으로 다음 map으로 넘어갑니다. 방향키를 이용하여 snake를 움직이고, 현재 이동방향과 반대방향의 키를 눌렀을 때는 게임이 종료됩니다. 노란색,보라색 item을 먹으면 길이가 1 증가하게 되고 초록색 아이템을 먹으면 길이가 1감소하게 됩니다. 또한 보라색 아이템을 먹으면 랜덤하게 map의 빈 공간에 벽(red zone)이 생깁니다. 몸의 길이가 3보다 작아지거나 snake가 벽이나, 자기 몸에 닿게 되면 게임이 종료됩니다. 처음 초기 스네이크 몸의 길이가 3이기 때문에 게임 실행 후 초록색 아이템을 바로 먹을경우 게임이 종료됨을 유의합니다.



 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	C++ Snakegame with ncurses	
	팀 명	C팀	
	Confidential Restricted	Version 1.2	2021-06-17

<죽었을 때 나오는 화면>

게임을 처음부터 다시 실행하고 싶다면 Y(y) 게임을 하고 싶지 않다면 N(n)을 누릅니다.



5.2 설치 방법

5.2.1 컴파일

- 개발 OS : Linux Ubuntu 16.04 LTS
- Version : C++ 14
- Compile Command : make

5.2.2 실행

컴파일과 동시에 자동으로 실행됩니다.