**C++프로그래밍**

**프로젝트**

|  |  |
| --- | --- |
| 프로젝트 명 | *Snake game* |
| 팀 명 | *팀 H* |
| 문서 제목 | snake game 결과보고서 |

|  |  |
| --- | --- |
| **Version** | 1.1 |
| **Date** | 2021-06-19 |

|  |  |
| --- | --- |
| **팀원** | 강다윤 (팀장) |
| 유광열 |

|  |
| --- |
| **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 “팀 H”의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 “팀 H”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다. |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | 최종보고서-snake-game.doc |
| **원안작성자** | 강다윤, 유광열 |
| **수정작업자** | 강다윤, 유광열 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2021-06-19 | 유광열 | 1.0 | 최초 작성 | 2, 4, 6단계 전반적인 보고서 작성 |
| 2021-06-19 | 강다윤 | 1.1 | 2차 작성 | 1, 3, 5단계 전반적인 보고서 작성 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**목 차**

[1 개요 4](#_Toc43103653)

[2 개발 내용 및 결과물](#_Toc43103654) 7

[2.1 목표](#_Toc43103655) 7

[2.2 개발 내용 및 결과물](#_Toc43103656) 8

[2.2.1 개발 내용](#_Toc43103657) 8

[2.2.2 시스템 구조 및 설계도](#_Toc43103658) 11

[2.2.3 활용/개발된 기술](#_Toc43103659) 27

[2.2.4 현실적 제한 요소 및 그 해결 방안](#_Toc43103660) 27

[2.2.5 결과물 목록](#_Toc43103661) 28

[3 자기평가](#_Toc43103662) 29

[4 참고 문헌 8](#_Toc43103663)

[5 부록 8](#_Toc43103664)

[5.1 사용자 매뉴얼 8](#_Toc43103665)

[5.2 설치 방법 8](#_Toc43103666)

# 개요

이번 snake-game 프로젝트에서 사용된 외부 라이브러리는 ncurses 한 가지입니다. 리눅스 우분투 개발환경에서 ncurses를 설치하는 방법은 ncurses설치 방법 가이드를 따라서

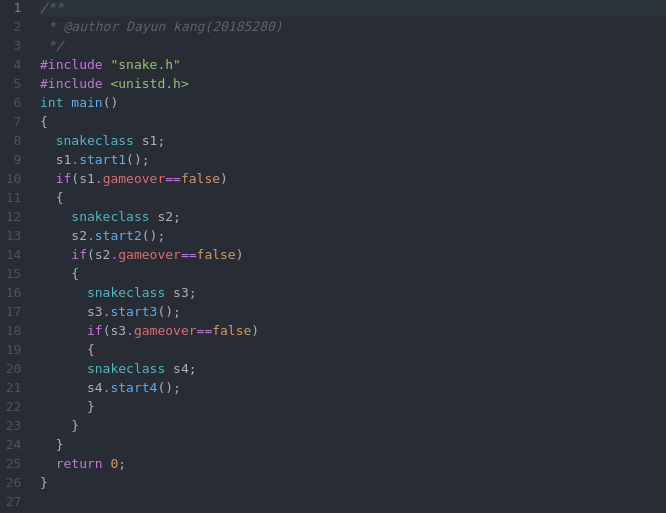
wget https://ftp.gnu.org/pub/gnu/ncurses/ncurses-6.2.tar.gz -P ~/Downloads

를 입력하여 설치하였습니다. 그 후 각 코드별로 #include <ncurses.h>를 기입하여 개발하였습니다. 또한 make 유틸리티를 이용하여 컴파일을 다음과 같이 진행하고 실행했습니다.

g++ map.cpp main.cpp snake.cpp scoreboard.cpp -lncurses

저희 조는 매번 Zoom을 활용하여 매일 회의를 진행하였으며, 회의 전까지 각자 맡은 부분을 구현하여 회의를 진행하였습니다.

* **main.cpp**



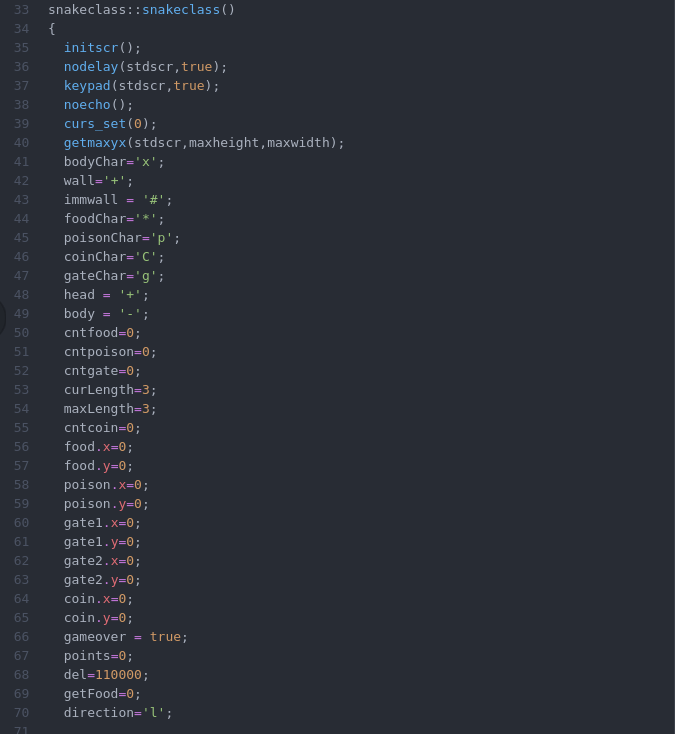
main.cpp에서 snakeclass 객체를 만들고, 전 게임의 gameover 값에 따라 다음 단계의 게임을 진행할지 말지 결정한다. Stage가 변하도록 설정하는 파일이다.

* **map.cpp**



map.cpp는 해당 satge별로 1~4단계 별로 맵을 구성하는 파일로 1단계 구성 부분이 다음과 같고, 그 다음 스테이지도 이와 유사한 방식으로 맵을 만든다.

* **snake.cpp**



snake.cpp는 snake의 생성 움직임, 아이템 생성, 게이트 생성 등 snake-game의 주요 전반적인 부분을 담고 있는 cpp파일이다.

* **scoreboard.cpp**



scoreboard.cpp 파일은 화면 우측에 구현되는 score를 표시하는 파일로 snake의 현재 길이, 최대 길이, 아이템 획득 수, 게이트 통과 횟수, coin등을 나타낸다.

# 개발 내용 및 결과물

## 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 | Coin 요소의 구현 (추가 아이디어의 구현) | 적용 |

**1단계 Map의 구현**

* Snake Game는 총 4단계로 이루어져 있다. 단계가 올라갈수록 벽이 추가되거나 맵이 바뀌도록 한다.Wall과 Immune Wall을 구분하여 만든다.

**2단계 snake 표현 및 조작**

* Snake의 y좌표, x좌표를 묶어 data라는 배열에 저장하고 움직이는 것을 목표로 합니다. Snake는 snake.x, snake.y로 좌표로 표하며, snake의 움직임에 따라 data에서도 같이 값이 변하는 것을 목표로 하였습니다. data에서 snake의 움직임은 아이템, 게이트와 같은 것의 위치 및 함수 활동에 사용됩니다. data에서 snake에 머리는 3, 몸통은 4로 표현할 것입니다. Snake는 방향키를 입력하여 해당 방향키에 맞는 방향으로 움직이며, 해당 부분에 mvaddch함수를 이용하여 몸과 머리를 입력함으로써, 뱀의 움직임을 ncurses에 표현할 수 있도록 구상했습니다. 해당 사용자의 입력이 없으면 기존의 진행방향으로 움직이며, 해당 진행 방향과 반대 방향으로는 움직이지 못하게 설정했습니다. snake는 머리 꼬리 몸통 부분을 조건문을 활용하여 해당 부분들을 표현하였으며, 머리와 몸통의 모양은 Coin을 모으면 해당 Coin에 따라 모양이 변하도록 설정했습니다.

**3단계 Item요소의 구현**

* Item에는 food와 poison이 있다. Food는 ‘\*’, poison은 ‘p’로 표현하며 data배열에서는 각각5와 6으로 구분된다. snake가 food와 충돌 시, snake의 size가 1이 증가하도록 설정한다. Snake가 poison과 충돌 시, snake의 size가 1이 감소하도록 설정한다. Food와 poison은 벽이나 snake의 일부분에 생기지 않도록 유의한다.

**4단계 Gate 요소의 구현**

* data배열에서 게이트는 7로 구분되며, Gate 실행 시, g로 표현되도록 했습니다. Gate는 맵에서 한 쌍이 겹치지 않고 data에서 1로 표시된 부분에만 생성할 수 있도록 설정했으며, putgate1,2를 활용하여 서로 다른 위치에 게이트가 생성하도록 목표를 잡았습니다. Snake가 Gate를 지나갈 때는 게이트가 사라지지 않으며 snake가 지나간 후에는 기존의 게이트가 막히고 새로운 게이트가 열리게 되도록 설정했습니다. Gate는 snake의 몸 길이가 5일 때 첫 번째 Gate가 발생하며, 그 이후에 몸 길이가 줄어들었다 다시 5를 넘어서도 게이트가 사라지거나 또 다른 게이트가 발생하지 않도록 설정했습니다.
* Snake는 들어가는 게이트에서의 방향과 나가는 게이트의 주변 상황에 따라 snake의 이동 방향이 다르도록 설정하였습니다. 맵 가장자리에 gate가 생성되면 맵 안 쪽으로 snake가 움직이도록 설정하였으며, 그 이외의 벽에서 gate가 발생하면 주어진 룰에 따라 움직이도록 설정하였습니다. 나오는 gate의 상하좌우의 값을 확인하여 벽이 존재할 경우 주어진 룰에 맞게 움직일수록 조건문을 활용하도록 설정했습니다. Snake가 게이트로 들어가면 다른 게이트의 좌표에서 snake를 출력하도록 설정하였습니다.

**5단계 점수 요소의 구현**

* 게임 화면의 우측에 점수판을 만든다. 점수판에서는 현재 스테이지#, 최종 스테이지#, 게임 진행 시간, 현재 뱀의 길이, 뱀의 최대 길이, 먹이와 독을 먹은 횟수, 게이트를 통과한 횟수, 미션 통과 여부, coin에 의해 결정되는 뱀의 모양 등을 볼 수 있다.

**6단계 Coin 요소의 구현**

* Coin은 snake가 food와 poison과 같이 해당 아이템을 먹었을 때 효과가 나도록 하는 아이템입니다. Coin은 각 코인을 모은 정도에 따라 해당 snake의 머리와 몸통의 모양을 변형시켜줍니다. Coin은 미션과는 별개이며, 해당 코인을 얼마나 모으는가에 따라 snake의 외형을 변형시켜주며, 해당 Coin의 수집정보와 코인에 따른 snake의 변형 모습도 점수판 하단에 표시되어 있습니다. 초기의 snake의 모습은 (+,-)로 표시되며 각 coin에 따라 (+,!) , (@,~), (@,=) (H,B) 형태로 발전하는 형태를 갖추도록 설정했습니다.

## 개발 내용 및 결과물

### **개발 내용**

**1단계 Map의 구현**

* Snake Game의 Map은 총 4개다. createMap함수의 stage인자에 따라 다른 맵이 출력된다. mvaddch를 통해 벽을 만들고 싶은 위치로 가서 Wall은 ‘+’, Immune Wall은 ‘#’ 모양으로 만든다. partClassify함수를 통해 Wall과 Immune Wall의 data값을 각각 1과 2로 구분 짓는다. cntmission을 정적 변수로 선언하여 새로운 snakeclass 객체가 생겨도 mission 달성 시 stage가 바뀔 수 있도록 한다.

**2단계 snake 표현 및 조작**

* Snake의 표현

data배열에서 snake는 머리는 3, 몸통은 4로 구분합니다. 실행화면에서 머리는 head, 몸통은 body로 6단계에서 coin의 양에 따라 head와 body값이 다릅니다. 초기로 설정된 head와 body는 +,-로 구성되어 있습니다. Snake는 사용자의 입력에 다라 snake의 위치가 바뀌며, 사용자의 입력이 없을 시 기존의 진행방향을 유지하여 움직입니다. 현재 진행방향과 반대 방향으로는 움직일 수 없습니다.

* Snake의 조작

화살표를 입력을 받아 그 방향으로 snake의 좌표를 1씩 움직입니다. Snake의 움직임 방향은 변수로 저장하여 gate 부분에서 사용했습니다. 해당 snake의 움직임을 partclassfy를 활용하여 data에 저장하며, mvaddch를 통해 해당 좌표로 움직여 맵에 snake가 표현되도록 했습니다. 그리고 data에서 snake의 head의 좌표를 활용하여 food, poison, Coin과 같은 아이템을 먹는 행위, 벽에 부딪히는 행위를 확인하며 그에 따라 snake의 길이가 길어지거나 짧아지고, 머리와 몸통의 형태가 변화하며, 길이가 3보다 작아지거나 벽에 부딪히면 게임오버가 되도록 했습니다.

**3단계 Item요소의 구현**

* Food Item의 표현

Food Item은 putFood함수를 통해 맵에 생성된다. Food는 맵 내부에 랜덤한 곳에 위치하도록 rand를 이용하여 x좌표 food.x와 y좌표 food.y를 설정한다. 이 때 먹이의 좌표가 뱀의 좌표와 겹칠 경우, 먹이가 맵 외부로 위치할 경우 다시 먹이의 좌표값을 설정하도록 한다. 이상이 없을 경우 mvaddch를 통해 해당 위치에 먹이 표시가 표현되도록 하고, partClassify함수를 통해 먹이 위치에 해당하는 좌표 값은 data값이 5를 갖도록 한다. collision함수를 통해 뱀이 먹이와 충돌하는지 알 수 있다. 뱀의 머리 좌표가 먹이 좌표와 일치할 경우 getFood는 true값을 가지며, 뱀의 길이는 1 증가하고 10포인트를 얻으며, 먹이가 다시 생성된다. 먹이 생성 시, ctime라이브러리를 이용하여 생성 순간부터 5초가 지난 후까지 뱀과 먹이의 충돌이 없으면 새로운 먹이가 새로운 위치에 등장하게 된다. 이전의 먹이는 사라지게 된다.

* Poison Item의 표현

Poison Item은 putPoison함수를 통해 맵에 생성된다. Poison는 맵 내부에 랜덤한 곳에 위치하도록 rand를 이용하여 x좌표 poison.x와 y좌표 poison.y를 설정한다. 이 때 독과 좌표가 뱀의 좌표와 겹칠 경우, 독이 맵 외부로 위치할 경우 다시 먹이의 좌표값을 설정하도록 한다. 이상이 없을 경우 mvaddch를 통해 해당 위치에 독이 표시가 표현되도록 하고, partClassify함수를 통해 독 위치에 해당하는 좌표 값은 data값이 6를 갖도록 한다. collision함수를 통해 뱀이 독과 충돌하는지 알 수 있다. 뱀의 머리 좌표가 먹이 좌표와 일치할 경우 getPosion는 true값을 가지며, 뱀의 길이는 1 감소하고 -10포인트를 얻으며, 독이 다시 생성된다. 독 생성 시, ctime라이브러리를 이용하여 생성순간부터 5초가 지난 후까지 뱀과 먹이의 충돌이 없으면 새로운 독이 새로운 위치에 등장하게 된다. 이전의 독은 사라지게 된다.

**4단계 Gate 구현**

* Gate의 표현

Gate는 맵에서는 g로 data에서는 7로 표현됩니다. Gate는 putgate1,2를 통해 게이트가 생성되며 rand함수를 사용하여 x,y 값을 랜덤으로 생성하여 게이트의 위치를 정합니다. 해당 게이트는 data에서 1로 구성된 벽에서만 생성됩니다. Mvaddch를 사용하여 맵에 표현하고, partClassfy를 사용하여 data배열 안에 값을 저장합니다. 게이트의 출현은 해당 snake의 길이가 5가 되었을 때 최초의 게이트가 생성되도록 했습니다. 그 외에 몸 길이가 5가 되었다 poison아이템의 영향으로 몸의 길이가 4가 되었을 때 게이트는 소멸되지 않으면, food를 먹어 다시 몸의 길이가 5가 되어도 새로운 게이트가 발생하거나 하지 않습니다. 게이트 게임 내에 한 쌍을 이루며 출현하며, snake가 지나간 후에는 해당 게이트는 mvaddch를 할용하여 gate를 막고, newgate를 사용하여 해당 data를 뒤져 7이 존재하는 경우 게이트를 생성하지 않고 존재하지 않으면 게이트를 생성하도록 작성하여 새로운 게이트가 발생합니다.

* Gate의 작동

Snake가 Gate1,2에 따라 들어간 게이트와 다른 곳에 위치하는 게이트에 snake가 나오도록 했습니다. 들어간 게이트가 1이면 나오는 게이트의 x,y 좌표 값을 사용하여 해당 위치에 snake가 출현하도록 했습니다. 나오는 게이트에 위치에 따라 해당 snake의 진행방향이 변경되도록 했습니다. 조건문을 활용하여 먼저 가장자리에 있는 게이트에서 나올 경우 맵 안쪽으로 snake가 움직이도록 설정했으며, 그 이외의 벽에서 생긴 게이트에서는 시계방향, 역시계방향, 반대 방향 순으로 snake가 진행하도록 설정했습니다. 이 순서를 결정하는 것은 게이트 주변의 값을 조사하여 해당 방향 수선대로 0이외의 값이 존재하는 경우 다음 방향으로 진출하도록 작성했습니다.

**5단계 점수 요소의 구현**

* 스테이지#, 게임 진행 시간, Coin

stage를 정적 변수로 선언하여 게임이 다음 단계로 진행될 때 마다 +1을 해주어 점수판에 나타내주었다. start1~4함수의 시작 부분에 ctime라이브러리를 이용하여 시간을 측정하여 점수판에 게임 진행 시간을 나타내었다. Coin에 의해 변하는 뱀의 모양을 나타내주었다.

* Score Board의 구현

curLength와 maxLength 변수를 만들어 뱀이 먹이와 독을 먹을 때 변수들에 그에 맞게 가감을 주었다. 이 변수들을 이용하여 현재 뱀의 길이, 뱀의 최대 길이를 나타내었다. cntfood와 cntpoison 변수를 만들어 뱀이 먹이와 독을 먹을 때마다 +1을 해주어 먹이와 독을 먹은 횟수를 나타내었다. cntgate 변수를 이용하여 뱀이 게이트와 충돌할 때 +1을 해주어 게이트 통과 횟수를 나타내었다.

* Mission Board의 구현

미션판에 설정해놓은 목표치에 도달했는지 여부를 위의 변수들과 조건문을 통해 알아내어 표기해주었다.

**6단계 Coin 구현**

* Coin 표현

Coin은 food,Poision과 마찬가지로 생성 후에 5초마다 해당 위치를 변경하며 map에 표현한다. Coin은 맵에서는 C로 표시되며 data에서는 8로 표현된다. Coin은 rand함수를 사용하여 랜덤한 x,y를 사용하여 위치를 정하며 c\_time을 활용하여 해당 아이템을 5초 동안 안 먹을 경우 그 다음 랜덤으로 다른 곳에 등장하도록 하였다.

* Coin 작동

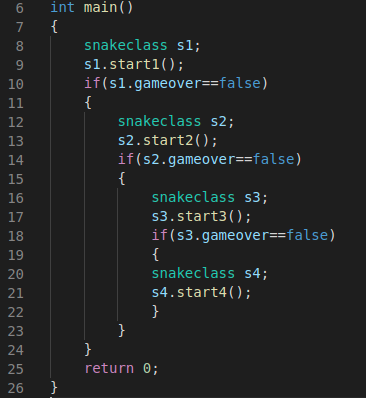
Coin 획득 시 점수판에 있는 Coin의 값이 100씩 변동한다. 해당 코인을 모아 300단위에 따라서 snake의 머리와 몸통의 모양이 다르게 표현된다. 이는 각 코인의 모인 양에 따라 조건문을 활용하여 head와 body의 값에 변화를 주었다.

### 시스템 구조 및 설계도

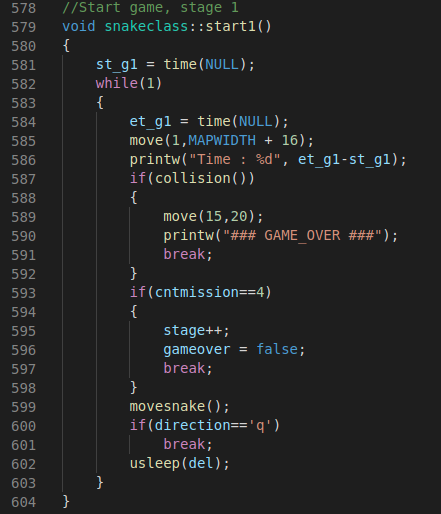
**1단계 Map의 구현**

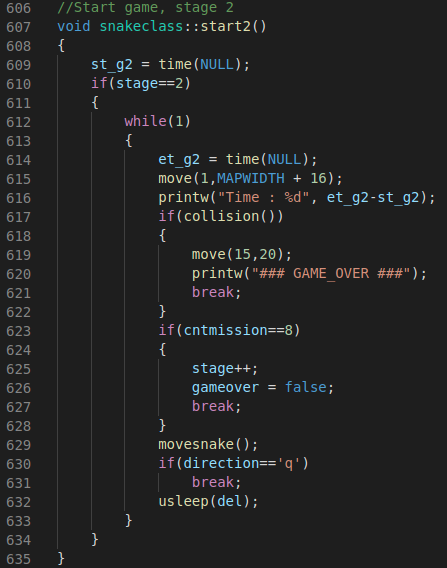


맵의 너비와 높이를 MAPWIDTH와 MAPHEIGHT를 통해 정의해주었다. 위의 코드는 스테이지 1일 때의map이다. mvaddch와 partClassify함수를 통해 wall과 immune wall을 표현하고 구분해주었다. 다른 스테이지도 위와 유사한 방식으로 만들었다.

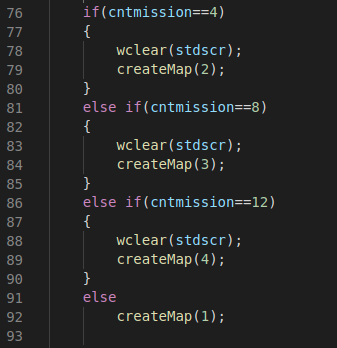


main.cpp에서 snakeclass 객체 생성을 하여 다음 단계로 가게 된다면 새로운 객체를 생성하여 start하는 방식을 사용했다. 정적 변수 cntmission을 통해 목표치에 도달하면 gameover가 false가 되어 다음 단계의 진행이 가능하다.





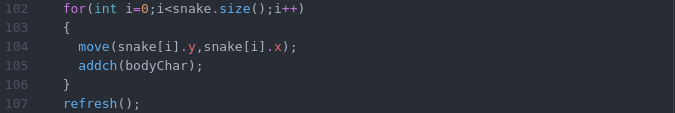
스테이지 1에서는 cntmission이 4일 때 다음 단계로, 스테이지 2에서는 cntmisson이 4 누적되어 8일 때 다음 단계로 넘어가게 된다.



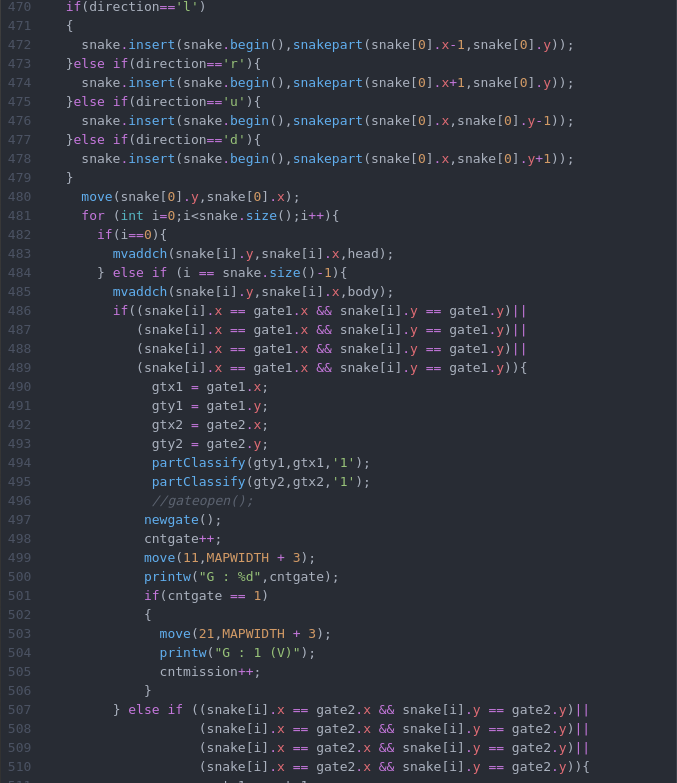
snakeclass 객체가 생성될 때 이 코드를 거치므로 cntmission에 따라 해당하는 createMap함수가 실행된다.

**2단계 snake 표현 및 조작**



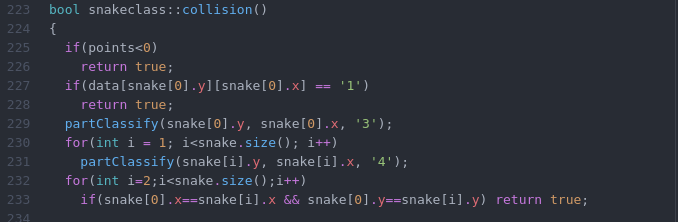








먼저 snakeclass에서 반복문을 활용하여 snakepart를 사용하여 snake의 초기 x,y값을 설정하여 snake를 생성한다. 그 이후 반복문과 move,addch를 사용하여 해당 좌표의 뱀을 출력한다. 그 다음 사용자로부터 getch함수를 통해 snake의 진행방향을 입력 받는다. 입력 받은 방향은 direction이라는 변수에 저장하여 사용한다. Direction에 저장한 값에 따라 insert문을 사용하여 해당 방향으로 snake의 좌푯값을 입력한다. 그 이후에 move함수를 사용하여 해당 좌표로 이동 반복문을 사용하여 머리, 몸통, 꼬리 부분을 나누어 mvaddch를 통해 해당 맵에 출력하고 partClassfy를 통해 data에 값을 저장한다.

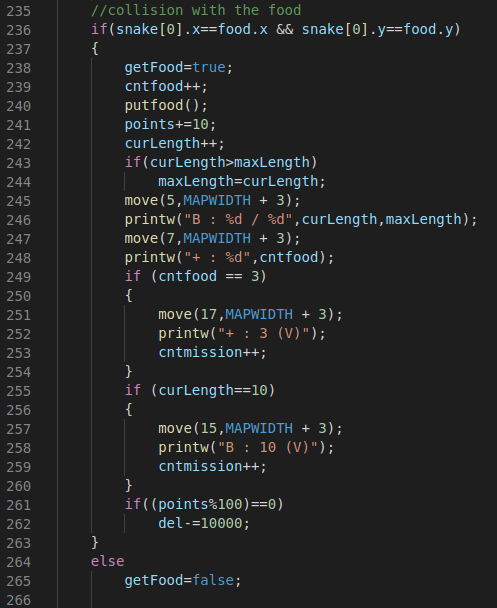
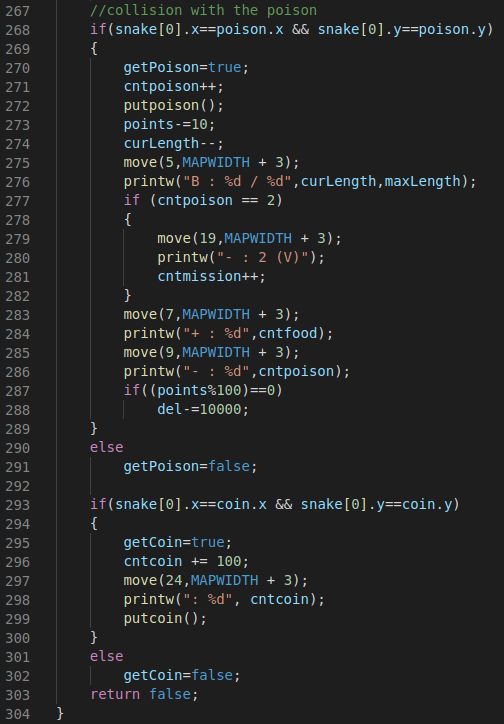


Collision부분에서 snake의 몸의 길이가 3보다 작거나, 벽으로 된 부분에 부딪히면 죽는 부분을 표현하였습니다.

**3단계 Item 요소의 구현**



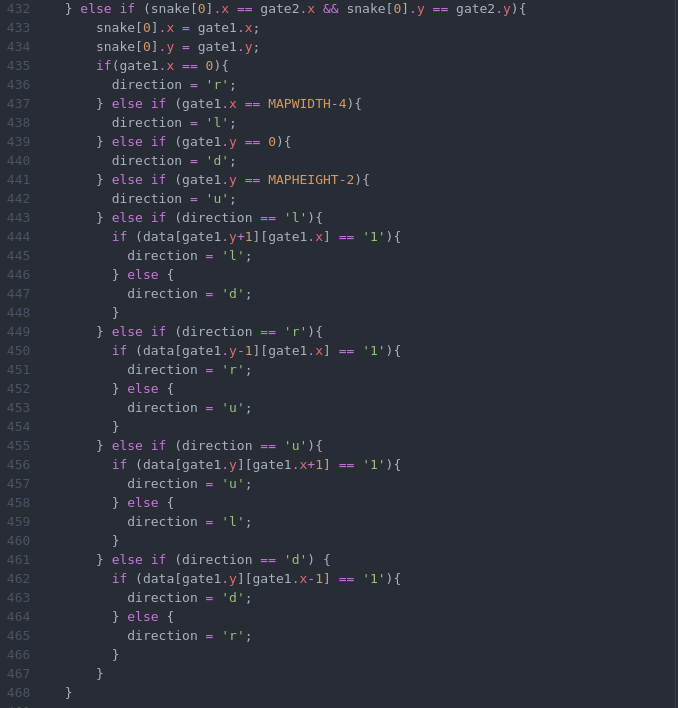
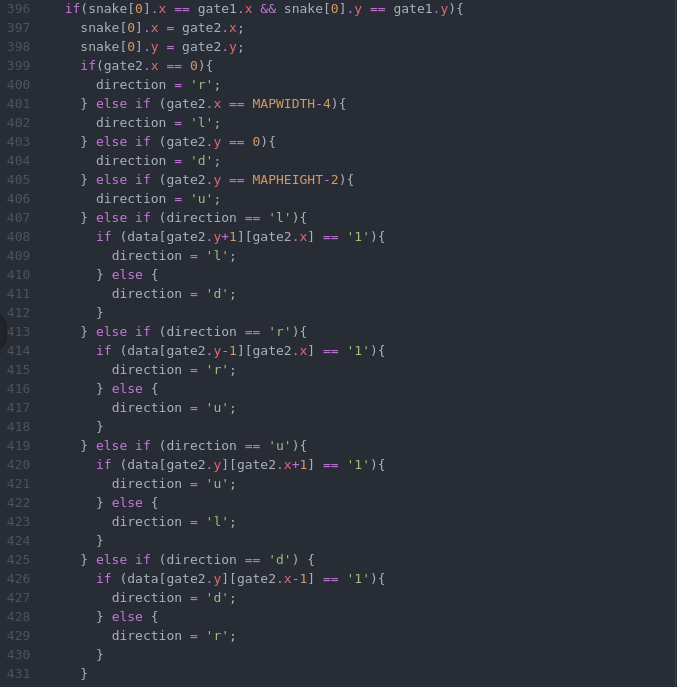
putfood와 putpoison 함수는 유사하다. 두 함수 모두 시작 시에 시간을 측정하기 시작한다. 그 다음 먹이와 독의 좌표값이 맵의 내부에, 벽에 겹쳐지지 않게, 뱀과 겹쳐지지 않도록 랜덤한 값을 설정해준다. 값이 결정되면 해당 좌표에 먹이와 독을 나타내는 기호를 mvaddch를 통해 나타내고 partClassify를 통해 data값을 구분해준다.

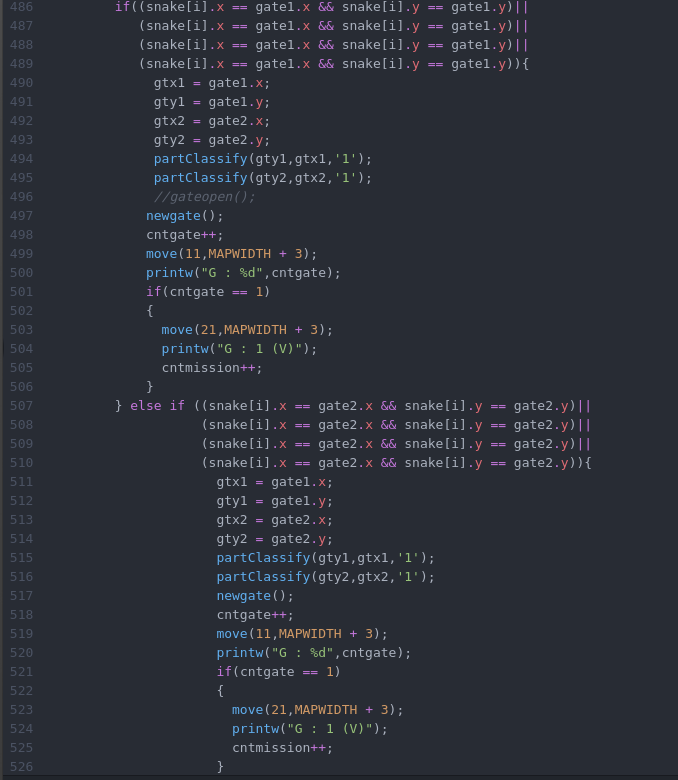


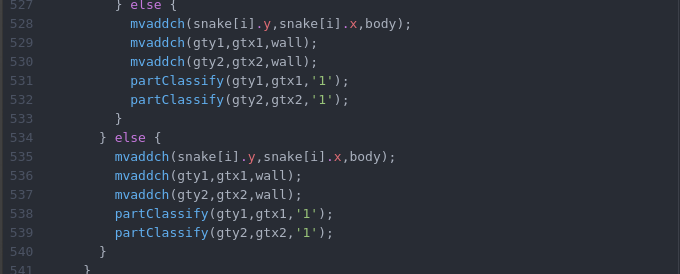
먹이와 독이 뱀의 머리와 충돌했을 시 수행하는 코드도 유사하다. 아이템을 먹으면 getFood와 getPoison이 true 값이 된다. 포인트는 +10, -10이 되며 현재 뱀의 길이도 늘어나고 줄어들게 된다. 변경된 사항들은 점수판에 새롭게 기재된다.

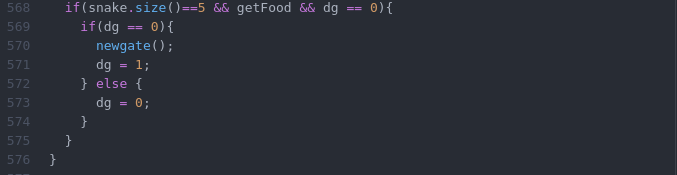
**4단계 Gate 구현**





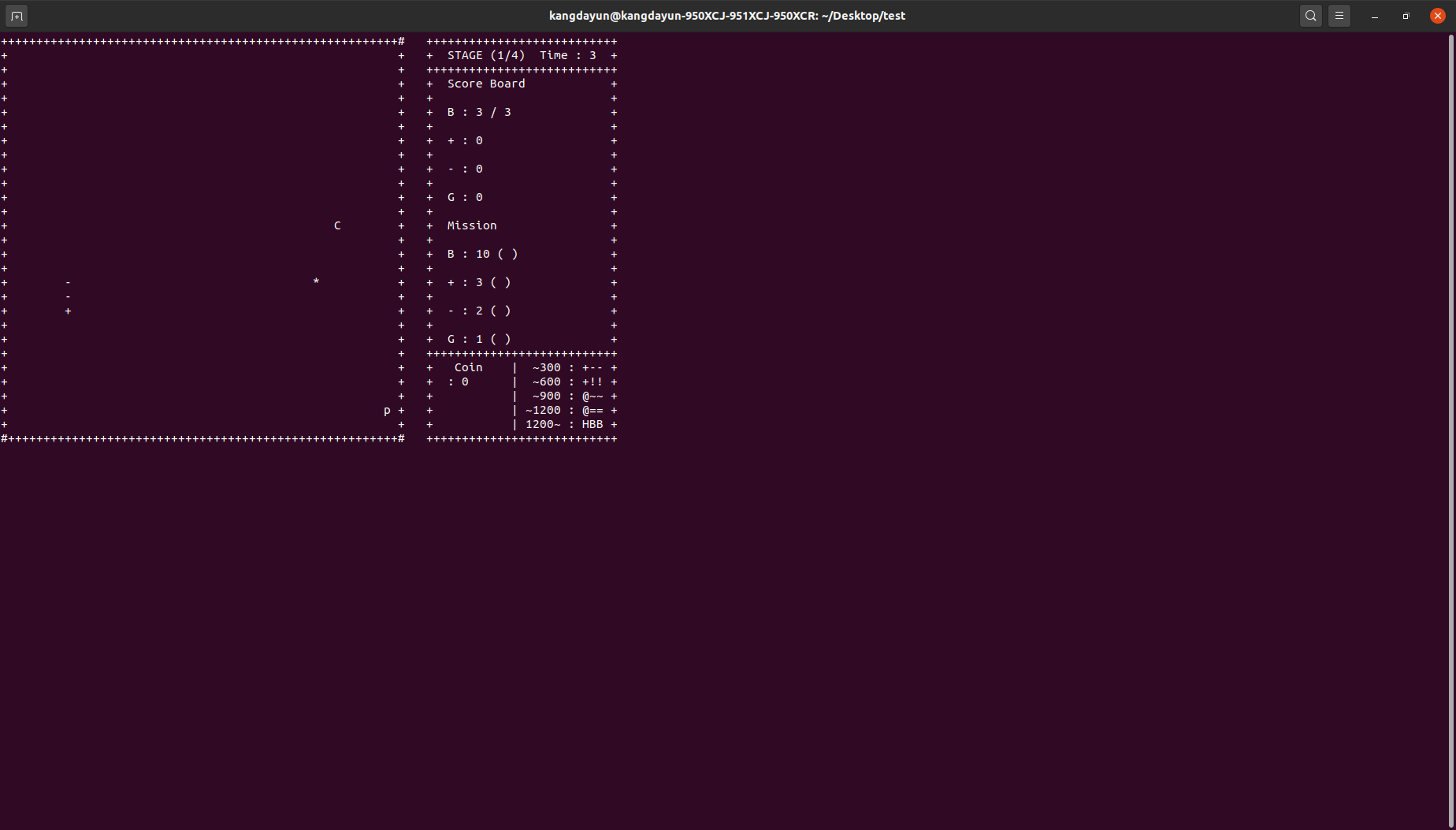






Gate는 putgate1,2를 통해 맨 처음 생성됩니다. putgate에서 rand함수를 사용하여 gate1,2의 x,y값을 랜덤으로 부여합니다. 두 게이트는 data의 값이 1인 부분에만 생성이 되도록 설정되어 있습니다. 게이트는 맵 상에서 구분이 되지 않으므로 들어가는 곳과 나가는 곳 또한 구분이 되지 않습니다. 그래서 해당 게이트로 들어올 때 나갈 때를 조건문을 통해 두 가지 상황에 대해 만들었습니다. Snake가 게이트로 들어가면 들어가지 않은 나가는 게이트로 좌표 값을 변경하여 snake가 나오도록 했습니다. Snake가 나올 때 나오는 게이트의 상황에 따라 snake에 진행 방향이 달라지도록 했습니다. 먼저 가장자리 게이트에서 snake가 등장할 경우 맵 안쪽으로 들어가도록 진행 방향을 설정했으며, 그 이외의 벽인 경우 조건문을 활용하여 시계방향, 반시계방향, 역방향 순으로 진행 시 1이 존재 할 경우 다음 방향으로 snake가 진행 하도록 설정했습니다. 해당 게이트를 snake가 지나가게 되면 mvaddch와 partclassfy를 사용하여 그 전 게이트가 닫히고, newgate함수를 사용하여 새로운 게이트가 열리도록 작성하였습니다. 그 다음으로 gate는 snake의 크기가 5일때 맨 처음 발생하도록 설정하였습니다. 그 이외에 snake가 poison아이템을 먹어 몸길이가 5이하가 된다고 해서 게이트가 소멸되지 않고, 다시 food를 먹어 몸 길이가 5가 되도 다른 부분에 게이트가 생기지 않도록 설정하였습니다.

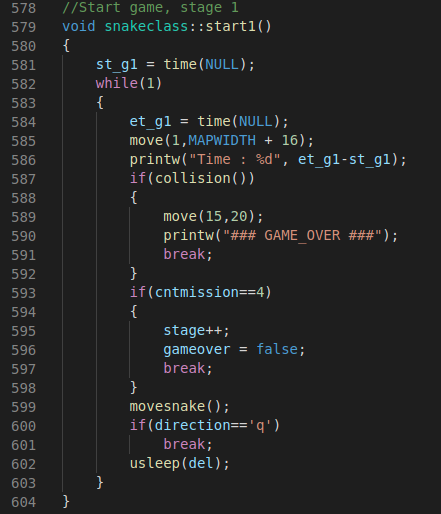
**5단계 점수 요소의 구현**



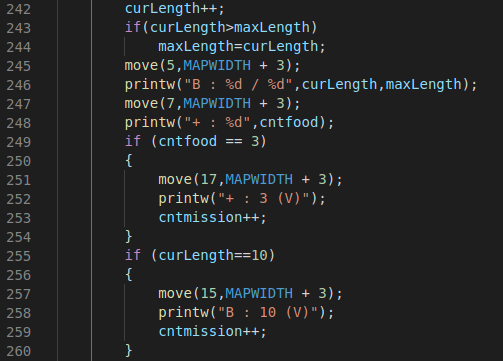
게임 실행 화면의 우측에 Score Board를 볼 수 있다.



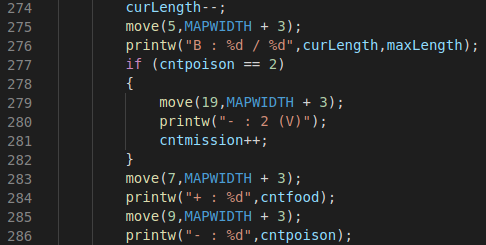
스코어보드의 너비와 높이를 각각 SBWIDTH와 SBHEIGHT로 값을 정의해주었다. move, addch, printw를 이용하여 각각의 항목들을 원하는 위치에 배치하여 스코어보드를 표현해주었다.



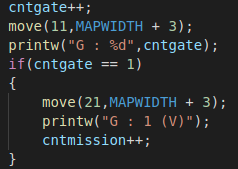
게임에 소요된 시간은 start함수에서 ctime라이브러리 함수를 이용하여 측정하였다.



먹이를 먹었을 경우 달라지는 뱀의 길이와 먹이를 먹은 횟수를 업데이트 한다.



독을 먹었을 경우 달라지는 뱀의 길이와 독을 먹은 횟수를 업데이트 한다.



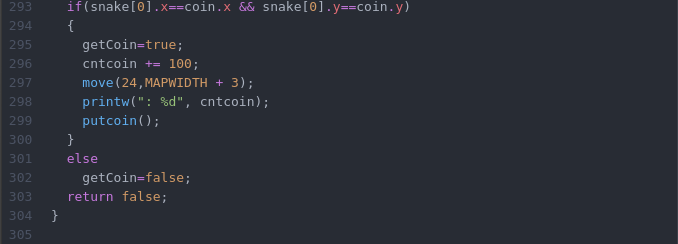
뱀이 게이트 통과 시, 게이트 통과 횟수를 업데이트 한다.



뱀이 코인을 먹었을 경우, 코인으로 얻은 점수를 업데이트 한다.

**6단계 Coin 구현**







Coin은 putCoin을 활용하여 맵에 등장한다. Food와 Poison과 같이 5초 마다 해당 위치가 변동되며, Coin을 먹을시 Coindml 값이 100씩 올라가며, 300단위로 snake의 머리와 몸통의 모양이 변화한다. Coin은 미션과는 상관없는 snake를 꾸미기 위해 존재하는 단계이다.

### 활용/개발된 기술

Snake-game 프로젝트에서 사용된 라이브러리는 ncurses와 ctime 등이 있다. ncurses라이브러리를 통해 Game을 실행하였으며, mvaddch를 사용하여 cli환경에 게임에서 오브젝트들의 결과를 출력해보았습니다. 그리고 ctime을 활용하여 food와 Poison, Coin을 사용자가 5초 동안 획득을 못 할 경우 다른 곳에 배치되도록 설정하는데 사용되었습니다.

### 현실적 제한 요소 및 그 해결 방안

1단계에서 다음 스테이지로 맵이 바뀌는 것을 같은 snakeclass객체 내에서 바꾸려다보니깐 맵이 겹쳐지고 전 스테이지에 있던 item들이 남아있었습니다. 클래스의 새로운 객체 생성과 정적 변수 cntmission과 stage를 선언함으로써 해결할 수 있었습니다.

2단계에서 snake를 움직일 때 해당 snake가 방향 전환 시 해당 snake의 모습이 여러 개로 분해되면서 이상한 형태를 띄었다. 당시 snake의 움직임을 좌표값을 게산하여 다음 칸으로 이동하는 형식이었는데 많은 오류가 발생하였다. 그래서 insert방식과 반복문, mvaddch를 사용하여 해당 문제를 해결하였다. insert문을 사용하여 해당 snake의 좌표값을 이동시키고, 반복문을 사용하여 snake를 mvaddch로 좌표값으로 이동시켜 머리와 몸을 출력하여 해결하였습니다.

3단계에서 item과 snake가 충돌하지 않을 시, 5초마다 재생성하기 위해 처음에 sleep함수를 사용했다. sleep함수는 전체 코드의 실행을 멈추었다가 실행되는 함수였기 때문에 사용이 불가능했고, 용도와 맞지 않았다. 방법을 찾아보다가 ctime 라이브러리의 time(NULL)을 이용하여 처음 아이템을 놓는 순간부터 시간을 재어 5초를 초과되면 아이템이 회수되고 재생성되는 미션을 해결할 수 있었습니다.

4단계에서 Gate를 만들 때 snake가 지나 간 후에 gate가 벽으로 변하지 않고 뚫린 채로 snake가 지나 갈 수 있는 통로가 생기는 문제가 발생하였다. 그래서 snake가 지나간 후에 gate에 mvaddch를 활용하여 해당 게이트 부분을 벽으로 만들었다. 그 이후에 다시 게이트를 생성하는 newgate부분을 추가하니 다시 같은 문제가 발생하였다. 이 문제를 해결하기 위해 여러 연구를 해보다가 조건문에 else부분에도 게이트를 막는 mvaddch부분을 추가하여 해당 문제를 해결하였다.

### 결과물 목록

|  |  |
| --- | --- |
| **파일** | **설명** |
| Main.cpp | 게임이 실행되는 메인 파일 |
| Map.cpp | 맵을 구성하는 파일 |
| Scoreboard.cpp | 스코어보드를 구성하는 파일 |
| Snake.cpp | 뱀의 움직임 아이템 생성 등 전반적인 파일 |
| Snake.h | snake파일의 헤더 파일 |
| Make | make명령어를 입력하면 게임이 시작됨 |

# 자기평가

**강다윤(20185280)**

역할:

Snake-game 프로젝트에서 1,3,5단계를 구현하였습니다. 전반적인 진행 방향과 일정을 정리하였습니다. 유광열 학우분과 함께 프로젝트를 진행하면서 개선해야 할 코드나 해결해야 할 문제에 대해 의논하였습니다.

프로젝트 수행 시 어려운 점:

C++의 Ncurses와 비슷한 python의 Pyqt5를 다룰 때에는 구글에 정리되어있는 자료들이 많아서 원하는 함수들을 수월하게 찾을 수 있었는데 그에 반해 Ncurses는 전체적으로 정보가 많이 없어서 비교적 힘들었습니다.

프로젝트 운영에 개선이 필요한 점:

구현한 코드를 서로 mail을 통해 주고 받았는데 github와 같은 협업 툴을 이용해보지 못해서 아쉬웠습니다. Github 이용 경험이 적고, 팀 프로젝트 경험이 없어서 선뜻 사용해보지 못했습니다. 좀 더 효율적인 협업과 안정적인 코드 관리를 위하여 github에 대해 공부를 하여 경험을 쌓을 필요가 있다고 느꼈습니다.

도움이 되었던 점:

프로젝트를 수행함으로써 수업에서 배웠던 C++ 이론들을 전반적으로 활용해볼 수 있는 기회가 되었습니다. 헤더파일의 역할에 대해서도 알았고, 특히 정적 변수의 쓰임새와 클래스의 객체 생성에 대해서도 직접 부딪히고 나니 깊게 이해할 수 있게 되었습니다. Ncurses에 대해 전반적으로 알게 되었습니다. 팀프로젝트를 하면서 같이 협업한 학우님의 코드를 보고 이해하고 배울 수 있는 기회였습니다.

팀원 평가:

유광열 학우님께서 열정적이고 의욕적인 모습들을 많이 보여주셔서 프로젝트를 끝까지 수행할 수 있었던 것 같습니다. 프로젝트 시작 전, 학우님께서 이 프로젝트가 재미있을 거 같아서 끝까지 잘 해보고 싶다고 말씀하신 것이 제게도 원동력이 되었습니다. 프로젝트 1단계를 구현할 때, map을 외부에서 가져오는 코드를 짜서 가져오셨는데 그 후 단계에 어려움이 있어 map을 내부에서 제작하는 방향으로 가면 어떨지 물어본 적이 있었습니다. 어쩌면 아쉬웠을 수도 있는 상황이었을텐데, 제 제안을 흔쾌히 받아주어 고마웠습니다. 학우님을 통해서 협업에 대한 마음가짐과 자세에 대해 배울 수 있었습니다.

**유광열(20192780)**

역할:

Snake-game 프로젝트에서 2,4,6단계를 집중적으로 구현했으며, 그 외에 단계는 팀장인 강다윤님의 코드를 돌려보면서 오류 및 다른 아이디어에 대해 생각하는 역할을 하였다. Zoom회의를 열어 회의를 진행하였으며, 전반적인 프로젝트 일정 및 프로젝트에서 부족한 부분을 체크하는 역할을 하였다.

프로젝트 수행 시 어려운 점:

타과생이어서 python언어 외에 처음으로 다른 언어로 프로젝트를 진행하는 거라 많은 어려움이 있었다. 일단 기존에 배운 C++을 아직 능숙하게 다루지 못해서 코드를 생각한 것을 직접 구현하는데 많은 어려움이 있었다. 논리적으로는 맞는 것 같은데 실제 결과가 다르게 나오는 것이 있었다. 그리고 C++에서 더 나아가 ncurses를 추가적으로 하면서 ncurses에 대해서 찾아보는데 자료도 적고 영어로 된 자료가 많아 해석하는데 많은 시간이 들었다.

프로젝트 운영에 개선이 필요한 점:

아직 C++을 능숙하게 못 다루고 알고리즘과 같은 부분을 많이 못 다루어 봤기 때문에 더 효율적이고 보기 좋게 코드를 작성 할 수 있을 거 같은데 반복문과 조건문과 같은 기본적인 문법만 갖고 프로그램을 작성하여 아쉬움이 있다. 그리고 변수명을 짓는데 정확한 의미를 부여하지 않은 변수명을 사용하여 오류가 난 적이 있어서 정확한 변수명을 작성하는 것이 필요하다고 느꼈다.

도움이 되었던 점:

도움이 되었던 점은 실제로 수업에서 배운 것을 프로젝트를 진행하면서 적용 할 수 있었고, 해당 프로젝트를 해결하기 위해 구글링을 하면서 많은 것을 배울 수 있었다. 그리고 해당 문제를 해결하기 위해 더 나은 방법을 찾기 위해 알고리즘에 대해 생각하고 연구하면서 알고리즘 적인 사고 방식이 늘어난 거 같다.

팀원 평가:

요번에 같이 팀플을 하게 된 다윤님은 여태껏 한 팀플 중에 손 뽑을 정도로 좋은 팀원

을 만났다고 생각한다. 프로젝트를 진행하면서 정말 열심히 하시는 것이 느껴질 정도로 맡으신 부분을 완벽하게 해오셨다. 서로 못 구현하는 단계들이 있었는데, 부족한 부분을 서로 메꾸면서 합이 잘 맞았던거 같다. 팀플이 많아서 항상 회의를 자정에 시작해서 새벽에 맞췄는데 감사하게도 그 시간 대에 회의를 해 주신 것이 너무 감사하다. 그리고 프로젝트의 전반적인 흐름을 이끌면서 팀장의 역할을 충분히 하셨다. 그리고 지금까지 다 해온 단계도 몇 번의 검증을 하시면서 프로젝트의 부족한 부분들을 채워 주셨다. 최고의 조원이었다!!

**기여도**

main.cpp : 강다윤

snake.cpp : 강다윤(50%), 유광열(50%)

map.cpp : 강다윤

scoreboard.cpp : 강다윤

snake.h : 강다윤(50%), 유광열(50%)

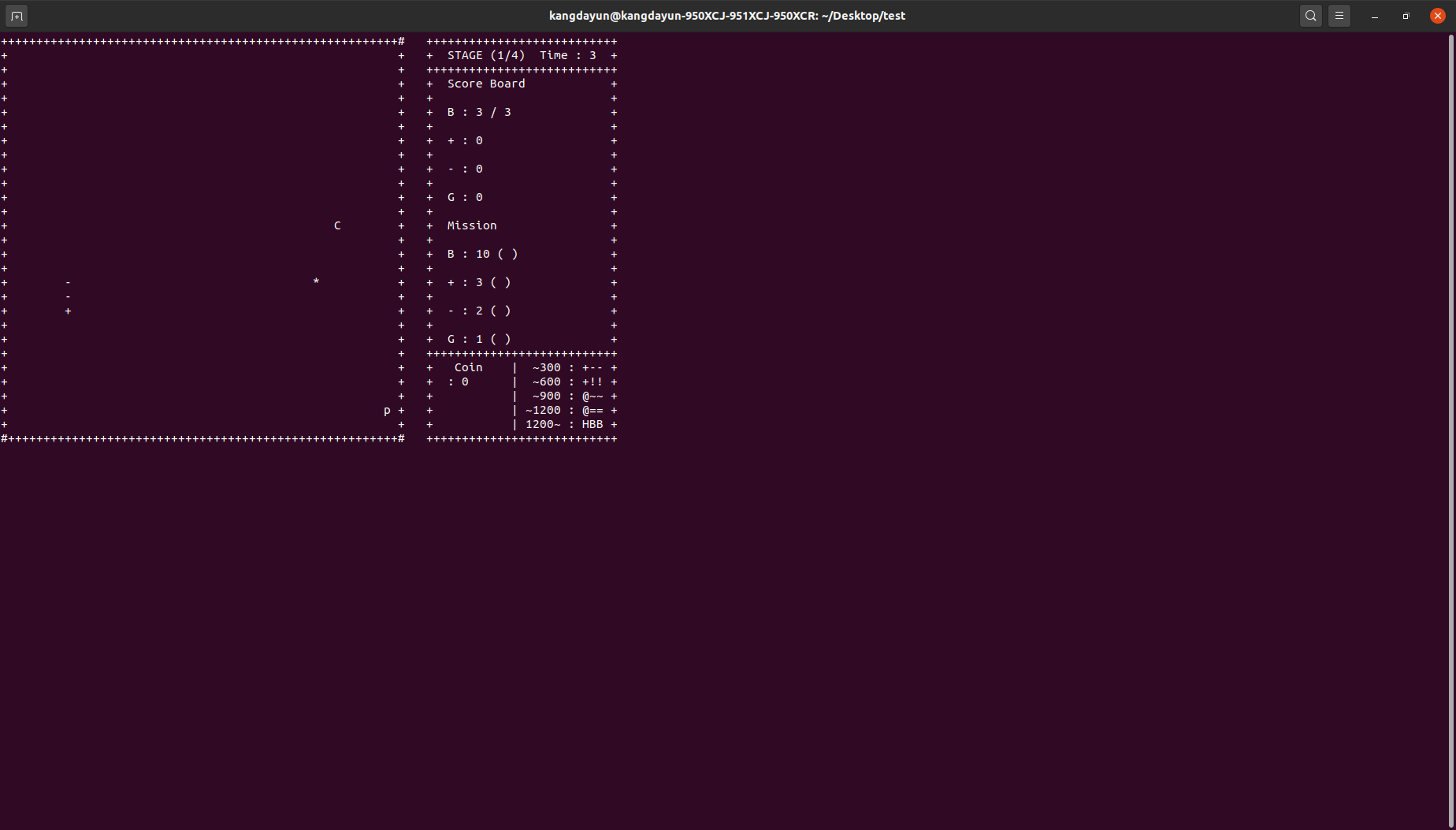
Makefile : 강다윤

# 참고 문헌

***https://modoocode.com/114***

# 부록

## 사용자 매뉴얼



**1. terminal에서 make 명령어 입력 시, 게임이 실행됩니다.**

**2. '\*'과 충돌하면 snake 길이가 1 증가하고 먹이를 먹은 횟수가 1만큼 증가합니다.**

**3. 'p'와 충돌하면 snake 길이가 1 감소하고 독을 먹은 횟수가 1만큼 증가합니다.**

**4. 'C'와 반복하여 충돌하여 일정 점수를 얻게 되면 해당 점수 대의 snake shape을 갖게 됩니다.**

**5. 모든 미션 완수 시, 되면 다음 스테이지로 넘어갑니다.**

## 설치 방법

**terminal에서 make 명령어를 입력하면 게임이 실행됩니다.**