


C++프로그래밍

프로젝트

| | |
|--------|-------------------|
| 프로젝트 명 | <i>Snake Game</i> |
| 팀 명 | <i>3분반 8조</i> |
| 문서 제목 | 결과보고서 |

| | |
|---------|------------|
| Version | 1.1 |
| Date | 2020-06-20 |

| | |
|----|-------------------|
| 팀원 | 20161943 최장혁 (팀장) |
| | 20163159 조성래 |
| | 20161934 인재휘 |

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |


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 **"8조"**의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 **"8조"**의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

문서 정보 / 수정 내역


| | |
|-----------------|---------------------|
| Filename | 최종보고서-SnakeGame.doc |
| 원안작성자 | 최장혁, 인재휘, 조성래 |
| 수정작업자 | 최장혁, 인재휘, 조성래 |

| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
|------------|-------|----------|----------|---------------------------------|
| 2020-06-19 | 최장혁 | 1.0 | 최초 작성 | 개발 내용 및 결과물 및 1, 2단계 내용 작성 |
| 2020-06-19 | 조성래 | 1.0 | 최초 작성 | 결과 목록, 참고 문헌, 부록 및 4, 5단계 내용 작성 |
| 2020-06-19 | 인재휘 | 1.0 | 최초 작성 | 개요 및 3단계 내용 작성 |
| 2020-06-20 | 인재휘 | 1.1 | 문서 정리 | 문서 정리 내용 통합 |
| | | | | |
| | | | | |
| | | | | |

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

목 차

| | | |
|-------|---------------------------|------------------------|
| 1 | 개요 | 4 |
| 2 | 개발 내용 및 결과물 | 5 |
| 2.1 | 목표 | 5 |
| 2.2 | 개발 내용 및 결과물 | 8 |
| 2.2.1 | 개발 내용 | 8 |
| 2.2.2 | 시스템 구조 및 설계도 | 11 |
| 2.2.3 | 활용/개발된 기술 | 37 |
| 2.2.4 | 현실적 제한 요소 및 그 해결 방안 | 39 |
| 2.2.5 | 결과물 목록 | 40 |
| 3 | 자기평가 | 41 |
| 4 | 참고 문헌 | 42 |
| 5 | 부록 | 42 |
| 5.1 | 사용자 매뉴얼 | 43 |
| 5.2 | 설치 방법 | 오류! 책갈피가 정의되어 있지 않습니다. |

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

1 개요

평가기준 (10점)

프로젝트를 완성하기 위해 사용한 개발 방법을 기술하세요.

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

이 게임을 만들기 위해 우리는 먼저 외부 라이브러리 <ncurses.h>를 사용하였습니다.

Ncurses는 먼저 우분투 기준으로 터미널에 들어가서 다음과 같이 입력합니다.

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

그 후에 프로그램 소스가 snake.cpp, main.cpp이기에 \$g++ -o snake 소스 - 파일 - 이름들 -lncurses를 써서 컴파일 했습니다.

뱀의 몸을 그리기 위해 우리는 <vector>를 사용했습니다.

사용자로부터 게임을 시작할 때 Y와 N을 입력 받기 위해 <iostream>을 사용했습니다.


게이트와 음식들의 좌표를 설정할 때 임의의 값을 주기 위해 <ctime>안에 있는 srand()함수를 이용하였습니다.

위의 3개의 라이브러리는 STL(표준 템플릿 라이브러리)를 설치하면 사용이 가능합니다.

음식들을 랜덤 한 위치에 생성하기 위해 우리는 <cstdlib>를 사용하였습니다.

PlayGame()의 while문이 한 번 돌 때마다 아주 잠깐씩 멈추는 것을 이용하기 위해 <unistd.h>안에 있는 usleep()함수를 사용하였습니다.

위의 2개의 라이브러리는 C 표준 라이브러리를 설치하면 사용이 가능합니다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

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 |

본 보고서는 C++ 프로그래밍 언어로 ncurses 라이브러리를 사용하여 Snake Game을 구현한 내용을 다룬다. 내용의 구성은 다음 5단계로 이루어지며, 각 단계는 다음의 내용을 가진다.

1단계 목표 : Ncurses Library를 사용하여 2차원 배열로 된 Snake Map을 Game 화면으로 표시하는 프로그램을 완성한다. 주의해야 할 점으로 **Wall**과 **Immune Wall**을 잘 구분해야 한다.

2단계 목표 : 1단계의 맵 위에 Snake를 표시하고, 화살표를 입력 받아 Snake가 움직이도록 프로그램을 완성한다. 여기서 Snake는 모든 Wall을 통과할 수 없고, Snake Head가 벽 또는 자신의 몸에 닿는다면 게임은 끝난다.

3단계 목표 : 2단계 프로그램에서, Map 위에 Growth Item과 Poison Item을 출현하도록 수정한다. 주의해야 할 점으로는 Map Data에서 Growth Item과 Poison Item은 구별되어야 한다.

음식의 조건

- Growth Item 을 먹으면 몸 길이가 1 증가한다,
- Poison Item 을 먹으면 몸 길이가 1 감소한다.
- 몸 길이가 3 보다 작아지면 게임은 실패한다.
- 음식의 개수는 최대 3 개로 제한하며, 출현 후 일정시간이 지나면 음식은 사라지고 다른 위치에 나타나야 한다.

4단계 목표 : Map의 Wall의임의의위치에 한 쌍의 **Gate**가 출현할 수 있도록 변경하고, **각 Gate에 Snake가 통과할 수 있도록** 수정한다

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

Wall(Immune Wall 포함)과 Gate 를 Map 배열에 표현할 때 값을 결정한다.

- 화면상에 표현시, Gate 는 Wall 과 구분될 수 있도록 한다.
- Color 값을 사용하여 구분하자

게이트의 조건

- Gate 는 두 개가 한 쌍이다.
- Gate 는 겹치지 않는다.
- Gate 는 임의의 위치에 있는 벽에서 나타난다.
- Gate 에 Snake 가 진입하면, 다른 Gate 로 진출한다.
- Gate 에 Snake 가 진입중인 경우 Gate 는 사라지지않고 다른위치에 Gate 가 나타나지않는다..
- Gate 는 한번에 한 쌍만 나타난다.
- Gate 는 fruit 들의 첫 reset 시 등장한다. (출현 방법)

게이트 이용시 진출 방향

- 항상 Map 의 안쪽 방향으로 진출한다.
- 상단 벽 -> 아래 방향
- 하단 벽 -> 위 방향
- 좌측 벽 -> 오른쪽 방향
- 우측 벽 -> 왼쪽 방향

5 단계 목표 : 우측에 **게임 점수를 표시하는 화면**을 구성한다.

게임 점수는 다음과 같이 측정 한다.

- 점수 계산 – fruit 50 점 poison -50 점
- 게임중 몸의 최대길이 계산 $B: (현재\ 길이)/(최대\ 길이)$
- 게임중 획득한 Growth Item 의 수 : +
- 게임중 획득한 Poison Item 의 수 : -
- 게임중 Gate 사용 횟수 : G

게임 방법 : 주어진 미션 달성하기, 미션 전부 클리어 시 다음 stage 이동
>미션 달성갯수 판단을 위한 변수를 두어 stage 이동 조건으로 사용한다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

Mission 의 각 스테이지 별 설정

1. 구성한 Map 의 정의에 고정 값을 주거나, (선택)
2. 매 게임마다 임의의 값을 주는 방식으로 처리

Mission 의 진행도를 시각화한다.

구현 목표 예시)


B: 10 (목표달성여부)

+: 5 ()

-: 2 ()

G: 1 (v)

*)조건에 맞는 목표를 달성시 해당 Mission () 안에 v 표시가 들어가 달성 여부를 구분,
V 표시시 달성한 미션의 개수를 지칭하는 변수의 값 +1 추가

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

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.

1단계 – Wall 과 Immune Wall 의 구현을 목표로 하였고, 다음과 같이 구현을 하였다.

외부 라이브러리 사용

1. Ncurses Library 사용을 위해 `#include <ncurses.h>`를 사용

Map 의 생성

1. 2개의 정수형 변수 `maxheight`와 `maxwidth`를 사용하여 스크린의 최대 폭, 높이를 설정
2. for 문을 통해 상, 하, 좌, 우의 벽에 지정한 문자를 입력하여 Wall을 구현
3. Immune Wall 위치에는 Wall과는 구별되는 다른 문자를 입력

2단계 – Snake 의 구현을 목표로 삼았으며, 뱀 출력 함수, 뱀의 움직임 함수, 뱀 충돌 함수의 구현과 게임의 시작 함수의 구현을 목표로 하였다.


Snake 구현

1. Vector를 사용하여 Snake를 구현하고자 `#include <vector>`를 선언
2. DrawSnake함수를 통해 뱀을 화면에 출력

Snake 움직임 구현

1. MoveSnake함수를 통해 뱀의 방향을 결정하며, 진행 방향과 반대 방향의 입력을 받았는지도 확인

Snake 충돌 판단

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

1. FatalCollision함수를 통해 머리 부분이 벽 또는 자신의 몸과 충돌하였는지를 판단

게임 시작

1. PlayGame함수를 통해 게임을 진행시킨다. 벽 또는 몸과의 충돌, 역방향 움직임 입력 유무를 확인하며 게임을 계속 진행시키거나 멈춘다.

3단계 - 과일, 독, 랜덤의 생성 및 과일, 독, 랜덤을 먹었을 때를 목표로 두었습니다.

과일, 독, 랜덤의 생성 및 충돌

1. 과일 - 과일 생성함수(PositionFruit)추가
2. 과일 - 과일 충돌함수(GetsFruit)추가
3. 독 - 독 생성함수(putPoison)추가
4. 독 - 독 충돌함수(getPoison)추가
5. 랜덤 - 랜덤 생성함수(putRandom)추가
6. 랜덤 - 랜덤 충돌함수(getRandom)추가

위의 함수들을 이용해서 과일, 독, 랜덤을 생성하도록 하고 뱀의 머리와 충돌했을 시 먹은 것으로 하였다.

과일, 독, 랜덤의 재생성


먼저 과일, 독, 랜덤의 유지시간을 쥔 변수 foodtime, poisontime, randomtime을 추가하였다.

그리고 뱀이 움직일 때마다 foodtime, poisontime, randomtime은 증가하고 그 값이 일정량이 넘어가면 기존에 있던 음식은 사라지고 새로운 음식이 생성된다.

4단계 - 게이트의 생성 및 게이트의 작동 의 두 가지 목표로 나누었고 각각의 목표에 대하여 더 하위 단계를 나눠 구현하였다.

게이트의 생성

1. 게이트 생성함수(makeGate와 makeGate2)의 추가
2. Wall과 구분되는 게이트의 색 지정 - white

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

게이트의 작동

1. 게이트의 작동 시점 (게이트의 좌표==스테이크 머리의 좌표) 판단
2. 머리 좌표의 이동 및 게이트 진입방향에 따른 출력 게이트에의 출력 방향 분리
3. 진입 방향에 따른 게이트 출력의 맵 구성에 따른 예외 처리

5단계 – 우측사이드에 **현재 점수와 달성해야할 미션판을 출력**하기 위해 각각의 생성 함수를 설정하고 **미션 달성시의 스테이지 이동 및 게임 종료조건**을 추가.

점수판의 출력

1. move함수와 printw함수를 사용하여 위에서부터 변동되는 점수사항을 표시
2. 내부에 미션판을 출력시키는 함수를 포함
3. playGame 루프의 MoveSnake함수에서 매번 PrintScore를 출력하여 상황을 최신화 유지

미션판의 출력

1. 각 맵마다 설정한 변수 map의 값에 따라 미션의 내용이 달라진다.
2. 각 미션에 대해 if와 else를 사용하여 달성시의 (v) 출력과 미달성 시의 () 출력 두 가지를 두고 달성시 미션 종류에 맞는 mission(1~4)까지의 값에 1이 입력.

스테이지의 이동 및 승리조건

1. playGame함수에 mission(1~4)를 모두 합한 값인 mission변수가 4가 되면 direction을 'p'로 변경하여 다음 스테이지로의 이동함수 nextStage()를 실행
2. playGame함수에 map의 변수가 4일 때 direction을 'f'로 변경하여 최종적으로 게임을 클리어 하였음을 출력

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

2.2.2 시스템 구조 및 설계도

작성요령 (30 점)

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.

1 단계 – 맵의 생성

- 20161943 최장혁 작성

- 1 단계에서 사용된 소스코드는 오른쪽과 같다.


maxheight 와 maxwidth 에 최대 폭과 높이를 저장하였다. 이 변수들과 for 문을 사용하여 상, 하, 좌, 우 벽 부분으로 move() 명령어로 커서를 해당 위치로 이동시킨 후, addch() 명령어를 통해 Wall 부분에는 (char)64 의 문자를, Immune Wall 부분에는 (char)219 의 문자를 삽입하여 벽을 구현하였다. 실행 결과는 아래와 같다.

```
#include <iostream>
#include <ncurses.h>
#include <cstdlib>
#include <unistd.h>
using namespace std;

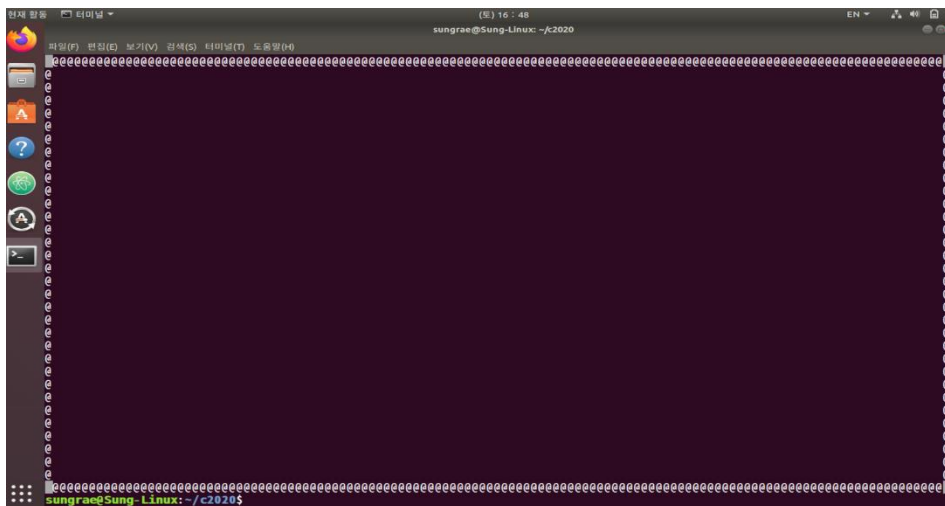
int main(){
    int maxheight, maxwidth;
    char edgechar=(char)64;
    srand(time(NULL));
    int UserInput = getch();
    refresh();
    endwin();
    clear();
    initscr();
    nodelay(stdscr, TRUE);
    keypad(stdscr, true);
    noecho();
    curs_set(0);
    refresh();
    getch();
    getmaxyx(stdscr, maxheight, maxwidth);
    maxheight-=1;
    for(int i=0; i<maxwidth; i++){
        move(0,i);
        addch(edgechar);
    }
    for(int i=0; i<maxwidth; i++){
        move(maxheight-1, i);
        addch(edgechar);
    }
    for(int i=0; i<maxheight; i++){
        move(i,0);
        if(i==maxheight-1 or i==0){
            addch((char)219);
        }else{
            addch(edgechar);
        }
    }
    for(int i=0; i<maxheight; i++){
        move(i, maxwidth-1);
        if(i==maxheight-1 or i==0){
            addch((char)219);
        }else{
            addch(edgechar);
        }
    }

    refresh();
    getch();

    return 0;
}
```

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

➤ 다음은 실행 화면이다.



2 단계 – Snake 생성, Snake 움직이기

➤ 본 단계를 구현하기에 앞서, SnakeGame.h 의 코드를 구현하였다.
SnakeGame.h 는 다음과 같다.

```
#include <iostream>
#include <vector>
#include <ncurses.h>
#include <cstdlib>
#include <ctime>
#ifndef SNAKEGAME_H
#define SNAKEGAME_H

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

class SnakeGame
{
private:
    int score, del, delay, maxwidth, maxheight;    /**
    char direction, headchar, partchar, edgechar;

    std::vector<CharPosition> snake; // 뱀 몸통담을 벡터

    void InitGameWindow();
    void DrawWindow();
    void DrawSnake();
    bool FatalCollision();
    void MoveSnake();
public:
    SnakeGame();
    ~SnakeGame();
    void PlayGame();
};

#endif
```

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

➤ 뱀을 구성하는 핵심 요소들은 다음과 같다.

- ① **char direction:** 뱀의 움직임을 저장하는 변수. 만약 왼쪽으로 이동한다면 'l', 오른쪽으로 이동한다면 'r', 위로 이동한다면 'u', 아래로 이동한다면 'd'를 저장한다.
- ② **void DrawSnake():** 뱀의 몸을 그리는 함수이다.
- ③ **bool FatalCollision():** 벽 또는 자신의 몸과 머리가 충돌하는지를 확인하는 함수이다.
- ④ **Void MoveSnake():** 뱀이 움직이는 방향을 결정하는 함수이다. 또한, 진행 방향과 정 반대 방향으로의 이동을 입력 받는지도 확인한다.
- ⑤ **Void PlayGame():** 게임을 진행시킨다. 벽과의 충돌, 역방향 움직임의 유무를 이용하여 게임을 계속 진행시키거나 멈춘다.

해당 함수들은 SnakeGame.cpp 에서 구현이 되어 있다.

➤ 이제 SnakeGame.cpp 에 구현되어 있는 알고리즘들을 살펴본다.

```
SnakeGame::SnakeGame()
{
    // 변수 초기화
    headchar = (char)229;
    partchar = 'x';
    edgechar = (char)64;

    //del은 딜레이 1000000이 1초 50만은 0.5초
    del = 100000;
    delay = 5000000;
    direction='l';
    srand(time(NULL));
    InitGameWindow();
    //maxwidth- 15한 이유는 점수를 오른쪽에 출력해야하는데 그 자리만들라고
    maxwidth-=20;
    DrawWindow();
    DrawSnake();
    refresh();
}

SnakeGame::~SnakeGame()
{
    nodelay(stdscr, false);
    getch();

    endwin();
}
```

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

- 먼저 SnakeGame()이다. 뱀의 머리 문자를 (char)229 로, 몸 부분을 'x'로 초기화 하였으며, 벽 부분은 (char)64 를 사용했음을 알 수 있다.

```
// draw snake's body
void SnakeGame::DrawSnake()
{
    for (int i = 0; i < 5; i++)
    {
        snake.push_back(CharPosition(20+i, 5));
    }

    //길이가 몇이든 길이만큼 표시해야함
    for (int i = 0; i < snake.size(); i++)
    {
        move(snake[i].y, snake[i].x);
        addch(partchar);
    }
    return;
}
```

- DrawSnake() 부분이다. Snake 의 i 번째 좌표마다 addch(partchar)를 함으로써 'x'가 스크린에 출력된다.

```
// set game over situations 종료할거면 true를 return
bool SnakeGame::FatalCollision()
{
    // 벽에 부딪힐경우
    if (snake[0].x == 0 || snake[0].x == maxwidth-1 || snake[0].y == 0 || snake[0].y == maxheight-2)
    {
        return true;
    }

    //자기몸에 부딪친경우
    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;
}
```

- FatalCollision() 함수이다. Snake 머리인 snake[0]의 x 좌표가 0 이면 왼쪽 벽과의 충돌을, maxwidth-1 은 오른쪽 벽과의 충돌을 의미하며, y 좌표가 0 은 천장과, maxheight-2 는 바닥과의 충돌을 의미한다. 만약 충돌이 일어난다면, true 를 반환한다. 두번째 for 문에서는 자기 몸과의 충돌을 판단한다. 만약

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

충돌이 일어난다면, true 를 반환한다. 벽에도, 몸에도 충돌하지 않았으면 함수는 false 를 반환한다.

```
void SnakeGame::MoveSnake()
{
    //뱀의 방향을 결정한다,
    int KeyPressed = getch();

    switch(KeyPressed)
    {
        case KEY_LEFT:
            if (direction != 'r')
            { direction = 'l'; break;}
            else if(direction == 'r')
            { direction = 'q'; break;}
        case KEY_RIGHT:
            if (direction != 'l')
            { direction = 'r'; break;}
            else if(direction == 'l')
            { direction = 'q'; break;}
        case KEY_UP:
            if (direction != 'd')
            { direction = 'u'; break;}
            else if(direction == 'd')
            { direction = 'q'; break;}
        case KEY_DOWN:
            if (direction != 'u')
            { direction = 'd'; break;}
            else if(direction == 'u')
            { direction = 'q'; break;}
        case KEY_BACKSPACE:
            direction = 'q';
            break;
    }
}
```

```
move(snake[snake.size()-1].y, snake[snake.size()-1].x);
printw(" ");
refresh();
snake.pop_back();

// 이동하는 방향에다가 문자를 넣어라
if (direction == 'l')
{ snake.insert(snake.begin(), CharPosition(snake[0].x-1, snake[0].y)); }
else if (direction == 'r')
{ snake.insert(snake.begin(), CharPosition(snake[0].x+1, snake[0].y)); }
else if (direction == 'u')
{ snake.insert(snake.begin(), CharPosition(snake[0].x, snake[0].y-1)); }
else if (direction == 'd')
{ snake.insert(snake.begin(), CharPosition(snake[0].x, snake[0].y+1)); }

//snake의 head부분씩 지정
move(snake[0].y, snake[0].x);
addch(headchar); // 새로운 머리를 지정

move(snake[1].y, snake[1].x);
addch(partchar); // 이전 머리의 위치에 몸통 문자를 출력

refresh();
return;
}
```

➤ MoveSnake() 함수이다. MoveSnake()함수는 방향을 받는 부분과, 받은 방향으로 움직이는 부분으로 나뉜다.

- ① 사용자 입력: 우선 keypressed 변수에 getch()로 사용자의 입력을 받는다. 사용자가 왼쪽, 오른쪽, 위, 아래 방향키를 입력한 것에 따라 각각 direction 변수에 'l', 'r', 'u', 'd'를 초기화 시킨다. 또한, 입력받은 방향이 현재 움직이는 방향의 반대 방향이면 direction 에 'q'를 초기화 한 후 break 한다.
- ② 입력 방향으로 이동: 입력받은 방향으로 머리의 좌표를 1 칸 이동시킨다. 예를 들어, 사용자가 왼쪽 방향키를 입력하여 direction 이 'l'로 초기화 되었다면, x-1 칸을 이동한 좌표에 snake.insert()를 실행시킨다.

위 실행이 끝나면, 새롭게 머리가 된 좌표에 headchar 문자를 출력해주고, 처음 머리가 있던 자리는 몸통이 되었으니 partchar 문자를 출력한다.

위 클래스의 내용들은 main.cpp 를 통해 실행한다.

마지막으로 main.cpp 를 알아본다.

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```
#include "SnakeGame.h"

int maxheight, maxwidth;
void PlayGame();
int IsUserReady();
int AskUserToPlayAgain();
void ClearCentre();
int UserInput();

int main ()
{
    if (IsUserReady() == 'y')
    do {
        {

            SnakeGame NewSnake;

            NewSnake.PlayGame();

        }
    }
    while (AskUserToPlayAgain() == 'y');
    return 0;
}
```

```
// print start menu
int IsUserReady()
{ //메시지 위치 배치먼저
    ClearCentre(3, 2.5);
    printf("Snake Game, Are you ready? press Y or N");
    return UserInput();
}

//종료시 출력
int AskUserToPlayAgain()
{
    ClearCentre(2.5, 2.5);
    printf("Do you want to play again? press Y or N");
    return UserInput();
}
```


- main 함수에서 IsUserReady()가 'y'임을 확인한다면 PlayGame()을 통해 게임을 시작하는 것을 알 수 있다. 게임이 종료되어 do 문을 벗어난다면, AskUserToPlayAgain()에서 다시 유저에게 입력을 받고, 그것이 y 라면 게임을 다시 실행시킨다.

```
//화면초기화, 커서 중앙 배치
void ClearCentre(float x, float y)
{
    clear();
    initscr();
    noecho();
    getmaxyx(stdscr, maxheight, maxwidth);
    move((maxheight/y), (maxwidth/x));
}

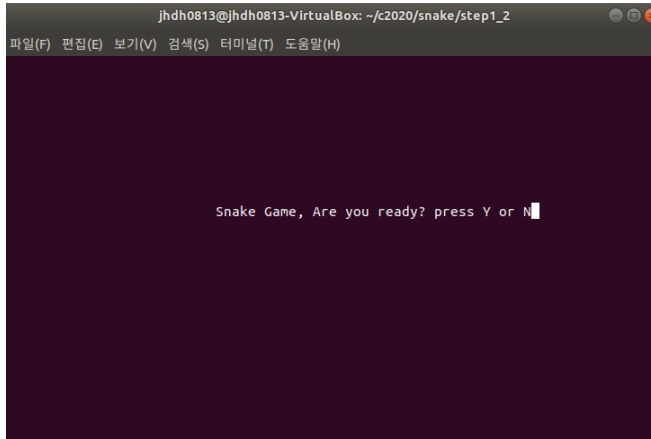
//유저입력받기
int UserInput()
{
    int UserInput = getch();
    refresh();
    endwin();
    clear();

    return UserInput;
}
```

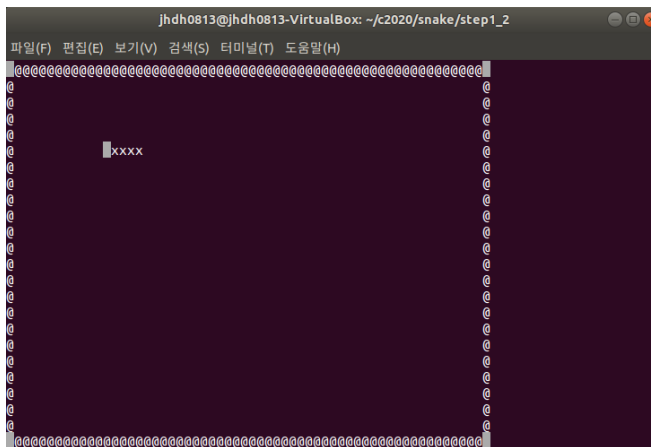
- 화면 초기화, 커서 중앙 배치, 유저에게 입력을 받는 함수는 다음과 같다.

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

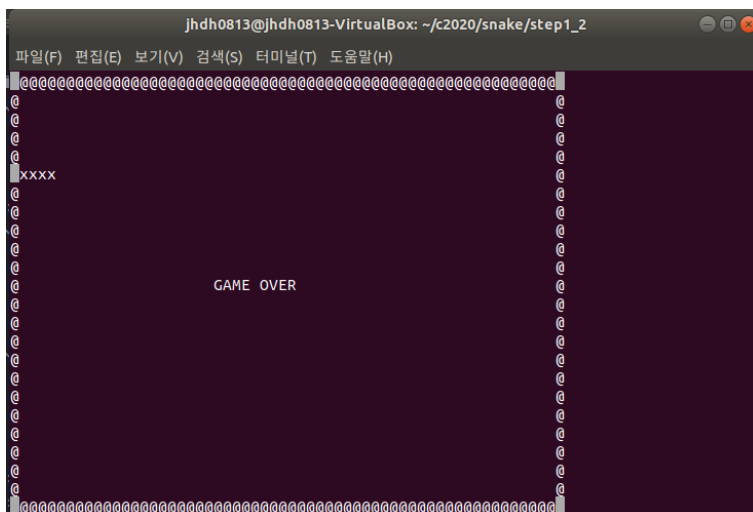
➤ 다음은 실행 화면이다.




➤ 사용자가 y를 입력한다면 게임은 다음과 같이 실행된다.



➤ 만약 충돌이 일어난다면 다음과 같이 게임은 종료된다.



| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

3 단계 – Growth Item, Posion Item 생성

– 20161934 인제휘 작성

```

void SnakeGame::PositionFruit()
{
    while(1)
    {
        int tmpx = rand()%maxwidth+1; // 1부터 랜덤 0을 피하기 위해서 1~maxwidth까지
        int tmpy = rand()%maxheight+1;

        // 뱀 위에는 나타나지 않는다
        for (int i = 0; i < snake.size(); i++)
        {
            if (snake[i].x == tmpx && snake[i].y == tmpy)
            {
                continue; // 뱀 위에 나타나는거였으면 밑에 무시하고 다시 루프돌림
            }
        }
        // 뱀에 있으면 넘어간다 **
        if (tmpx >= maxwidth-2 || tmpy >= maxheight-3)
        {
            continue;
        }
        // 2,3,4스테이지의 뱀까지 넘어가게 포함시킨다
        if(map==1){
            if((tmpx==maxwidth/2 and (tmpy>=0 and tmpy<maxheight/2-2)) or
              (tmpx==maxwidth/2 and (tmpy<=maxheight and tmpy>maxheight/2+2))){
                continue;
            }
        }
        else if(map==2){
            if((tmpx>=maxwidth/4+5 and tmpx<=(maxwidth*3)/4-6 and (tmpy>=maxheight/4+2 and
              tmpy<=(maxheight*3)/4-3)) {
                continue;
            }
        }
        else if(map==3){
            if(((tmpx==maxwidth/5 and (tmpy>=maxheight/4 and tmpy<=maxheight/2)) or
              ((tmpy==maxheight/4) and (tmpx>maxwidth/5 and tmpx<maxwidth)) or
              ((tmpy==(maxheight*3)/4) and (tmpx>0 and tmpx<=(maxwidth*4)/5)) or
              ((tmpx==(maxwidth*4)/5) and (tmpy>=maxheight/2 and tmpy<=(maxheight*3)/4))){
                continue;
            }
        }
        // 좌표설정되면 맵상에 표시
        fruit.x = tmpx;
        fruit.y = tmpy;
        break;
    }
    move(fruit.y, fruit.x);
    attron(COLOR_PAIR(1));
    addch(fruitchar);
    attroff(COLOR_PAIR(1));
    refresh();
}

```

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

➤ Void SnakeGame::PositionFruit()함수이다.

tmpx, tmpy 에 랜덤 한 수를 대입해서 그 수의 위치가 라운드의 벽 위나 뱀의 몸 위에 없다면 그 좌표에 과일을 나타나게 하였다.


```
void SnakeGame::putPoison()
{
    while(1)
    {
        int tmpx = rand()%maxwidth+1;
        int tmpy = rand()%maxheight+1;
        for(int i=0; i<snake.size(); i++){
            if((snake[i].x == tmpx && snake[i].y ==tmpy) || (fruit.x == tmpx && fruit.y==tmpy))
                continue;
        }
        if(tmpx >= maxwidth-2 || tmpy >= maxheight-3){
            continue;
        }

        if(map==1){
            if((tmpx==maxwidth/2 and tmpy>=0 and tmpy<maxheight/2-2) or
                (tmpx==maxwidth/2 and (tmpy<=maxheight and tmpy>maxheight/2+2))){
                continue;
            }
        }else if(map==2){
            if((tmpx>=maxwidth/4+5 and tmpx<=(maxwidth*3)/4-6) and (tmpy>=maxheight/4+2 and
                tmpy<=(maxheight*3)/4-3)){
                continue;
            }
        }else if(map==3){
            if(((tmpx==maxwidth/5) and (tmpy>=maxheight/4 and tmpy<=maxheight/2)) or
                ((tmpy==maxheight/4) and (tmpx>maxwidth/5 and tmpx<maxwidth)) or
                ((tmpy==(maxheight*3)/4) and (tmpx>0 and tmpx<=(maxwidth*4)/5)) or
                ((tmpx==(maxwidth*4)/5) and (tmpy>=maxheight/2 and tmpy<=(maxheight*3)/4))){
                continue;
            }
        }

        poison.x = tmpx;
        poison.y = tmpy;
        break;
    }
    move(poison.y, poison.x);

    attron(COLOR_PAIR(2));
    addch(bad);
    attroff(COLOR_PAIR(2));
    refresh();
}
```

➤ Void SnakeGame::putPoison()함수로 과일을 두는 방식과 비슷하며 이 때는 독을 배치한다.

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

void SnakeGame::putRandom()
{
    while(1)
    {
        int tmpx = rand()%maxwidth+1;
        int tmpy = rand()%maxheight+1;
        for(int i=0; i<snake.size(); i++){
            if((snake[i].x == tmpx && snake[i].y == tmpy) || (fruit.x == tmpx && fruit.y == tmpy) || (poison.x == tmpx && poison.y == tmpy))
                continue;
        }
        if(tmpx >= maxwidth-2 || tmpy >= maxheight-3){
            continue;
        }

        if(map==1){
            if((tmpx==maxwidth/2 and (tmpy>=0 and tmpy<maxheight/2-2)) or
               (tmpx==maxwidth/2 and (tmpy<=maxheight and tmpy>maxheight/2+2))){
                continue;
            }
        }else if(map==2){
            if((tmpx>=maxwidth/4+5 and tmpx<=(maxwidth*3)/4-6 and (tmpy>=maxheight/4+2 and
               tmpy<=(maxheight*3)/4-3)){
                continue;
            }
        }else if(map==3){
            if(((tmpx==maxwidth/5 and (tmpy>=maxheight/4 and tmpy<=maxheight/2)) or
               ((tmpy==maxheight/4 and (tmpx>maxwidth/5 and tmpx<maxwidth)) or
               ((tmpy==maxheight*3)/4 and (tmpx>0 and tmpx<=(maxwidth*4)/5)) or
               ((tmpx==maxwidth*4)/5 and (tmpy>=maxheight/2 and tmpy<=(maxheight*3)/4))){
                continue;
            }
        }


        random.x = tmpx;
        random.y = tmpy;
        break;
    }
    move(random.y, random.x);

    if(randnum == 0)
    {
        start_color();
        attron(COLOR_PAIR(1));
        addch(fruitchar);
        attroff(COLOR_PAIR(1));
    }
    else
    {
        start_color();
        attron(COLOR_PAIR(2));
        addch(bad);
        attroff(COLOR_PAIR(2));
    }
    refresh();
}
//

```

➤ Void SnakeGame::putRandom()함수이다.

이 함수는 과일이나 독을 둘 때와 비슷하며 뒷부분에 임의로 구한 수가 짝수이면 과일을 홀수이면 독을 배치하도록 만들었다.

| | | | |
|---|-------------------------|----------------------------|------------|
|  | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

//뱀상의 fruit먹었을때
bool SnakeGame::GetsFruit()
{
    //뱀좌표 snake[0].x , snake[0].y가 fruit의 좌표와 동일하면 먹는상황임
    if (snake[0].x == fruit.x && snake[0].y == fruit.y)
    {
        foodtime = 0;
        //PositionFruit() fruit을 먹었으니 랜덤한 위치에 재생성
        PositionFruit();
        score +=50;
        fruitnum++;
        PrintScore();

        // if score is a multiple of 100, increase snake speed
        if ((score%100) == 0)
        {
            del -= 4000;
        }
        return bEatsFruit = true;
    }// ****
    else if(foodtime%100==0){
        move(fruit.y,fruit.x);
        printw(" ");
        //fruitnum--;
        PositionFruit();
    }

    else
    {
        return bEatsFruit = false;
    }

    return bEatsFruit;
}


```

➤ bool SnakeGame::GetsFruit()함수이다.

뱀의 머리 좌표가 과일의 좌표와 일치할 때 뱀이 과일을 먹은 상황이 된다.

먹었을 때 점수와 스피드를 올리며 기존에 있던 과일이 사라지고 새로운 음식이 맵에 등장하게 만들었다. 그리고 이 함수의 return 값이 True 가 된다.

또한 foodtime 이 일정 시간이 되면 기존에 있던 food 를 없애고 재생성 한다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

bool SnakeGame::getPoison()
{
    if(snake[0].x==poison.x && snake[0].y == poison.y)
    {
        poisonsime = 0;
        get_poison = true;
        putPoison();
        if(snake.size()==4){
            direction='q';
        }
        score-=50;
        if(score<0){ //score는 음수가 안나오게 한다
            score=0;
        }
        poisonnum++;
        PrintScore();
    }
    /**
    else if(poisonsime%100==0){
        move(poison.y,poison.x);
        printw(" ");
        putPoison();
    }
    else{
        return get_poison = false;
    }
    return get_poison;
}

```

➤ **bool SnakeGame::getPoison()**함수이다.

이 또한 뱀의 머리 좌표가 독의 좌표와 일치할 때 독을 먹는 상황이 된다.

독을 먹었을 때 점수를 깎으며 기존에 있던 독은 사라지고 새로운 독이 맵상에 등장하게 된다. 그리고 이 함수의 return 값이 True가 된다.

또한 poisonsime 이 일정 시간이 되면 기존에 있던 poison 를 없애고 재생성한다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

bool SnakeGame::getRandom()
{
    if(snake[0].x == random.x && snake[0].y == random.y)
    {
        randomtime = 0;
        get_random = true;

        if((snake.size() == 4) && randnum == 1){
            direction = 'q';
        }
        if(randnum == 0){
            score += 50;
            check = 0;
            fruitnum++;
        }
        else{
            score -= 50;
            check = 1;
            poisonnum++;
            if(score < 0){
                score = 0;
            }
        }
        PrintScore();
        randnum = rand()%2;
        putRandom();
    }
    else if(randomtime % 100 == 0){
        move(random.y, random.x);
        printf(" ");
        randnum = rand()%2;
        putRandom();
    }
    else{
        if(randnum == 0){
            check = 0;
        }
        else{
            check = 1;
        }
        return get_random = false;
    }
    return get_random;
}


```

➤ bool SnakeGame::getRandom()함수 이다.

뱀의 머리 좌표가 랜덤의 좌표와 일치할 때 먹은 상황이 된다.

이것을 먹었을 때의 효과는 과일일 땐 과일을 먹었을 때, 독일 땐 독을 먹었을 때의 효과와 같다. 그리고 이 함수의 return 값이 True 가 된다.

또한 randontime 이 일정 시간이 되면 기존에 있던 random 을 없애고 재생성 한다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```
// SnakeGame newGame >> newGame.PlayGame(); 으로 게임 시작
void SnakeGame::PlayGame()
{
    while(1)
    {
        foodtime+=1;
        poisontime+=1;
        randomtime+=1;
        //머리 충돌시, 역방향입력시 게임 종료 가운데에 출력
        if (FatalCollision())
        {
            move((maxheight-2)/2,(maxwidth-5)/2-5);
            printf("GAME OVER");
            break;
        }


        //먹었나안먹었나 bool 판별한 뒤에 MoveSnake()실행
        GetsFruit();
        getPoison();
        getRandom();
        MoveSnake();
        //-----
        if(gate==true){
            gatenum=gatenum+1;
            PrintScore();
            gate=false;
        }
        else
            gate=false;

        if(direction=='p'){

            move((maxheight-2)/2,(maxwidth-5)/2-2);
            printf("Next Stage");
            refresh();
            //미션 충족시 next stage가 중앙에 뜨고 2초 뒤 자동으로 다음 스테이지
            usleep(2000000);
            clear();
            refresh();

            direction='l';
            nextStage();
        }
    }
}
```

- Void SnakeGame::PlayGame() 함수이다.
매번 코드가 돌아갈 때마다 foodtime, poisontime, randomtime을 1씩 증가시켜놓는다.
이로 인해 현재 생성되어 있는 과일, 독, 랜덤의 시간을 잴 수 있으며 각각을 원하는 시간 뒤에 재생성 시킬 수 있다.


| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

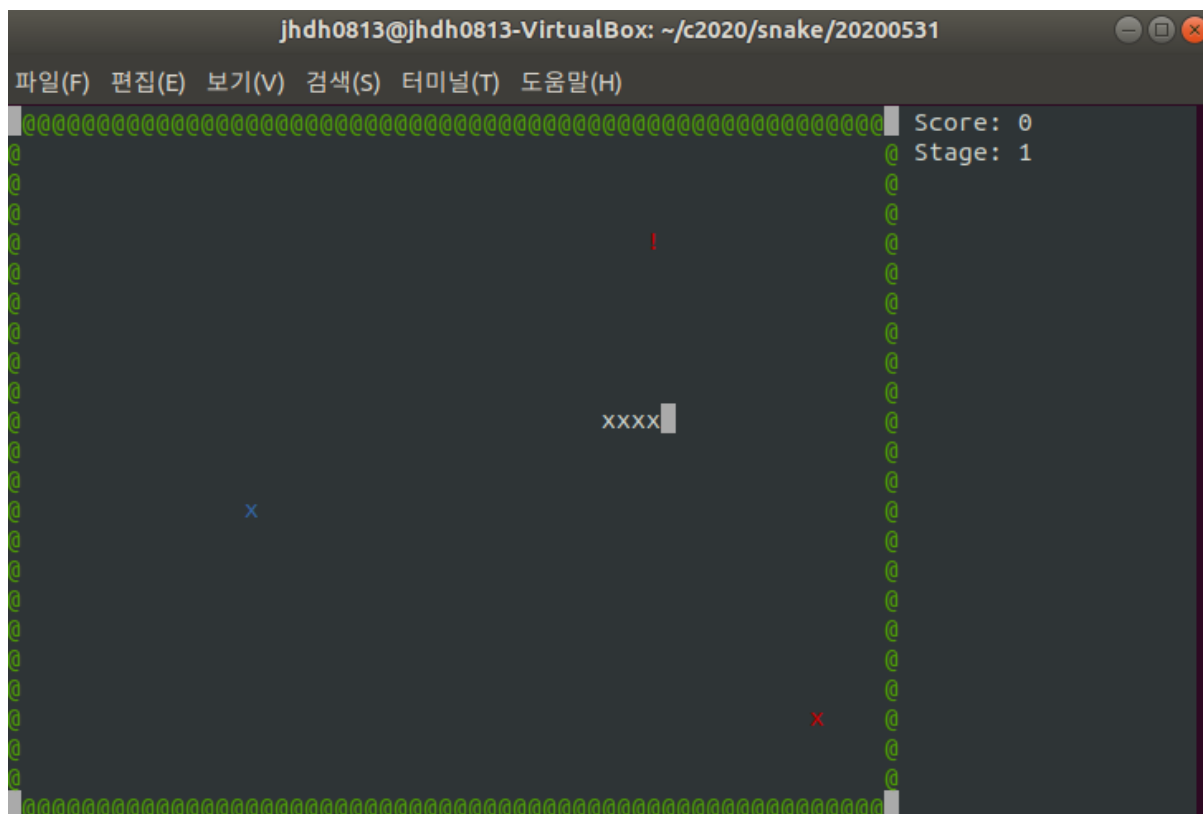
void SnakeGame::MoveSnake()
{
    // Fruit과 pPoison을 둘 다 안먹었을때 size 그대로, 앞으로 전진, 맨뒤제거, 새로운 머리, 예전머리 몸통으로표기
    if(check == 0)
    {
        if(!bEatsFruit && !get_random)
        {
            //snake 몸통 vector의 맨 마지막부분을 지우고 vector의 크기도 1줄임
            move(snake[snake.size()-1].y, snake[snake.size()-1].x); // 맨 뒤 요소로 이동
            printw(" ");
            refresh();
            snake.pop_back();
        }
        if(get_poison)
        {
            move(snake[snake.size()-1].y, snake[snake.size()-1].x);
            printw(" ");
            move(snake[snake.size()-2].y, snake[snake.size()-2].x);
            printw(" ");
            refresh();
            snake.pop_back();
        }
    }
    else
    {
        if(!bEatsFruit)
        {
            //snake 몸통 vector의 맨 마지막부분을 지우고 vector의 크기도 1줄임
            move(snake[snake.size()-1].y, snake[snake.size()-1].x); // 맨 뒤 요소로 이동
            printw(" ");
            refresh();
            snake.pop_back();
        }
        if(get_poison || get_random)
        {
            move(snake[snake.size()-1].y, snake[snake.size()-1].x);
            printw(" ");
            move(snake[snake.size()-2].y, snake[snake.size()-2].x);
            printw(" ");
            refresh();
            snake.pop_back();
        }
    }
}
//note1에 적임시

```

- **Void SnakeGame::MoveSnake()함수에 다음과 같이 추가하였다.**
 랜덤음식이 과일이면 check == 0일 때 충돌 검사하고 독이면 check == 1일때의
 충돌검사를 한다.

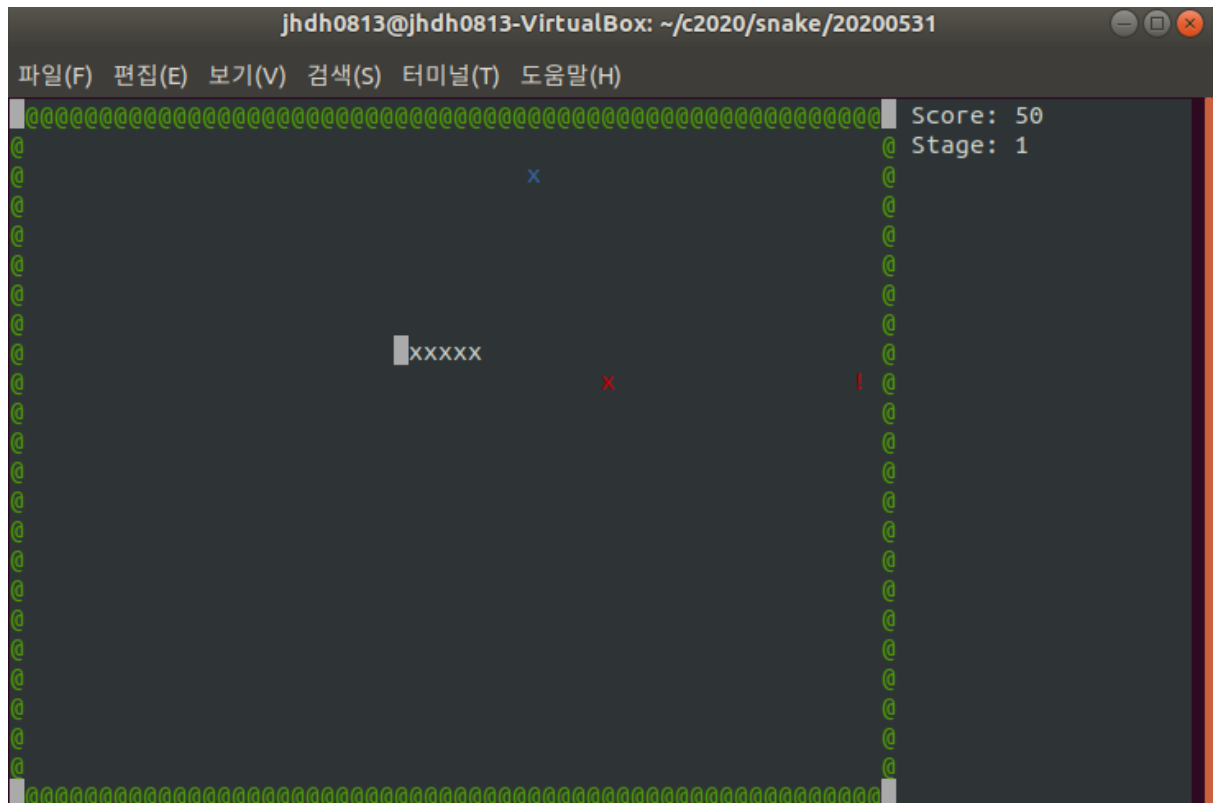
| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

➤ 다음은 실행 화면이다.



➤ 먼저 게임이 시작되면 임의의 위치에 과일, 독, 랜덤이 생성된다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |



- 파란색을 먹게 되면 과일을 먹은 것이 되며 점수가 오르고 뱀의 길이가 증가한다. 그리고 새로운 위치에 과일이 생성된다. 만약 먹었던 것이 과일이 아닌 랜덤이었다면 생성할 때 독이나 과일 중 하나가 랜덤 하게 생성된다.

4 단계 – 게이트의 생성, 게이트의 작동

- 20163159 조성래 작성

구현 함수 : makeGate(), makeGate2()

| | | | |
|---|-------------------------|----------------------------|------------|
|  <div> 국민대학교 컴퓨터공학부 캡스톤 디자인 I </div> | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

게이트의 생성

```
//게이트 생성 일단 생성 조건은 mission이 3개 클리어 시로 지정
void SnakeGame::makeGate(){
    if(gateadd == false){
        return;
    }else{
        while(1){
            int tmx = rand()%maxwidth;
            int tmy = rand()%maxheight;

            if(map==0){
                //immune wall 위에 있으면 재시도
                if((tmy==0 and (tmx==0 or tmx==maxwidth-1)) or (tmy==maxheight-1 and (tmx==0 or tmx==maxwidth-1))){
                    continue;
                }else if((tmy==0 and (tmx>0 and tmx<=maxwidth-1)) or (tmy==maxheight-1 and (tmx>0 and tmx<=maxwidth-1)) or (tmx==0 and (tmy>0 and tmy<=maxheight-1)) or (tmx==maxwidth-1 and (tmy>0 and tmy<=maxheight-1))){
                    gate1.x=tmx;
                    gate1.y=tmy;
                    break;
                }else{
                    continue;
                }
            }
        }
    }
}
```

- makeGate 함수는 playGate()함수 내에서 실행된다 , 임의의 수를 x,y 좌표로 받아 현재

진행중인 맵(stage)에 맞는 wall 을 고르는 코드, immune wall 은 제외한다. 또한, gate2 의 경우, gate1 의 좌표와는 중복되지 않아야한다.

```
if((tmpx==gate1.x and tmpy==gate1.y) or (tmpx==maxwidth-1 and tmpy==maxheight/4) or (tmpx==0 and tmpy==(maxheight*3)/4)){
    continue; // 찾은 좌표가 gate1의 좌표와 동일하다면 다시 찾아
}else{
    gate2.x=tmpx;
    gate2.y=tmpy;
    break;
}
```

- makeGate 가 먼저 실행되어진 상태이기 때문에 그때 결정난 좌표와의 중복은 continue

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

        direction='p';
    }
}

if(foodtime==100){                                //-----게이트 생성 조건, 일단은 foodtime=100

    makeGate();
    makeGate2();

    gateadd=false;
}

```

- (foodtime==100 이 되는순간 makeGate(), makeGate2()함수가 playGame()함수 내에서 실행된다.)


게이트의 작동

```

//gate1에 진입 시
if(snake[0].y==gate1.y and snake[0].x==gate1.x){
    gate=true;
    if(direction=='l'){
        if(gate2.x==0){                                //맨 왼쪽에 두개 생겼을때.
            direction='r';
            snake.insert(snake.begin(), CharPosition(gate2.x+1, gate2.y));
        }else if(gate2.x==maxheight-1){
            direction='u';
            snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y-1));
        }else if(gate2.y==0){
            direction='d';
            snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y+1));
        }else if(gate2.y==maxheight-1){
            direction='u';
            snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y-1));
        }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y !=maxheight-1))
            and gate2.y==0){                                //map1 일때 중앙부
            direction='d';
            snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y+1));
        }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y !=maxheight-1))
            and gate2.y==maxheight-1){                    //map1 일때 중앙부
            direction='u';
            snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y-1));
        }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y !=maxheight-1))
            and gate2.x==0){                                //map1 일때 중앙부
            direction='r';
            snake.insert(snake.begin(), CharPosition(gate2.x+1, gate2.y));
        }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y !=maxheight-1))

```

- MoveSnake()함수 내에서의 gate 진입 판단 코드
- 스네이크 게임은 playGame()함수로 실행되며, playGame()함수는 fruit,poison,random 3 가지 요소의 습득 여부의 판단과 MoveSnake()함수의 이동 명령 실행의 무한루프 반복을 통해 진행된다. 따라서 MoveSnake()함수 내부에 게이트와의 접촉여부를 판단하거나 MoveSnake()함수 앞에 게이트 접촉여부판단을 하는 함수를 생성해야한다, 이 경우 우리는

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |


MoveSnake()함수 내부에서 판단을 하기로 하였다.

```
// gate2으로 들어갈때의 경우
}else if(snake[0].y==gate2.y and snake[0].x==gate2.x){
    gate=true;
    if(direction=='l'){
        if(gate1.x==0){ //맨 왼쪽일때
            direction='r';
            snake.insert(snake.begin(), CharPosition(gate1.x+1, gate1.y));
        }else if(gate1.y==maxheight-1){
            direction='u';
            snake.insert(snake.begin(), CharPosition(gate1.x, gate1.y+1));
        }
    }
}
```

- 맨 먼저 게이트생성을 두 함수로 나눠 생성하였으므로, gate1, gate2 둘 중 어느곳에 접근을 했는가를 판단.

```
if(direction=='l'){
}
}else if(direction == 'r'){
}
}else if(direction == 'u'){
}
}else if(direction=='d'){
}
```

- 둘 중 어느 게이트로 접근하냐가 판단되었으면 그 내부에서 그렇다면 어느 방향으로 진입했는가를 각각 구분하게됨.

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

if(direction=='l'){
    if(gate2.x==0){ //맨 왼쪽에 두개생겼을때.
        direction='r';
        snake.insert(snake.begin(), CharPosition(gate2.x+1, gate2.y));
    }else if(gate2.x==maxheight-1){
        direction='u';
        snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y-1));
    }else if(gate2.y==0){
        direction='d';
        snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y+1));
    }else if(gate2.y==maxheight-1){
        direction='u';
        snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y-1));
    }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y !=maxheight-1))
        and gate2.y==0){ //map1 일때 중앙부
        direction='d';
        snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y+1));
    }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y !=maxheight-1))
        and gate2.y==maxheight-1){ //map1 일때 중앙부
        direction='u';
        snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y-1));
    }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y != maxheight-1))
        and gate2.x==0){ //map1 일때 중앙부
        direction='r';
        snake.insert(snake.begin(), CharPosition(gate2.x+1, gate2.y));
    }else if((gate1.x==maxwidth/2 and (gate1.y !=0 or gate1.y !=maxheight-1))
        and gate2.x==maxwidth-1){ //map1 일때 중앙부
        direction='l';
        snake.insert(snake.begin(), CharPosition(gate2.x-1, gate2.y));
    }else if((gate1.x==0 and gate2.x==maxwidth/2 and gate2.y==maxheight/2+2) or
        (gate1.x==maxwidth-1 and gate2.x==maxwidth/2 and gate2.y==maxheight/2+2)){
        direction='l';
        snake.insert(snake.begin(), CharPosition(gate2.x-1, gate2.y));
    }
}

```

- 진입방향에 따라 반대편 출력 게이트의 주변 벽의 위치를 고려하여 각각 맵에 의한 모든 출력 가능성들을 예외처리. 스네이크의 이동은 스네이크의 head부의 좌표를 반대편 출력 게이트에서 나와야 할 곳의 좌표로 넣어준다.

Ex)snake.insert(snake.begin(), CharPosition(gate2.x, gate2.y-1));

5단계 - 우측상단에 게임의 상황 (점수판, 미션판, 스테이지)를 표시

- 20163159 조성래 작성

구현 함수 : PrintScore(), PrintMission()

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

점수의 표시

```
// 점수 표시 우측상단에 할거
void SnakeGame::PrintScore()
{
    move(0, maxwidth+5);

    attron(COLOR_PAIR(7));
    printw("Score Board");
    attroff(COLOR_PAIR(7));
    move(1, maxwidth+1);
    printw("B: %d", snake.size()/5);
    move(2, maxwidth+1);
    printw("+: %d Items", fruitnum);
    move(3, maxwidth+1);
    printw("-: %d Items", poisonnum);
    move(4, maxwidth+1);
    printw("G: %d Use", gatenum);
    move(5, maxwidth+1);
    printw("-----");
    move(6, maxwidth+1);
    printw("Score : %d", score);
    move(7, maxwidth+1);
    printw("-----");
}
```

- PrintScore()함수 생성, move함수와 printw함수로 올바른 위치에 이동하여 문장 출력
- 출력될 변수 snake.size()/5 , fruitnum, poisonnum, gatenum, score는
 - Snake.size()/5 = 초기 사이즈 5에 대한 현재 사이즈의 비율
 - Fruitnum = 습득한 fruit의 개수
 - Poisonnum = 습득한 poison의 개수
 - Gatenum = Gate의 사용 횟수
 - Score = 현재 점수

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

printw("*-Mission-*");
attroff(COLOR_PAIR(7));
PrintMission();
move(15, maxwidth+1);
printw("end mission: %d",mission);
move(maxheight-1, maxwidth+1);
printw("stage : %d", map+1);
move(0,0);
attron(COLOR_PAIR(6));
addch((char)33);
attroff(COLOR_PAIR(6));
return;

```


- PrintScore()함수의 내부에 미션판을 출력시키는 PrintMission()함수가 포함
PrintScore()함수 실행 -> 점수메뉴 출력 -> PrintMission()함수 실행 -> 미션판 출력

```

// score Board 와 Mission 판을 표시, fruit or poison을 먹거나 gate 사용시 변수가 바뀌고 미션조건 달성시
// Mission 판에 달성 여부가 v 표시로 체크된다, 그리고 Mission 4가지가 전부 v 되어야 다음 스테이지로 넘어감
void SnakeGame::PrintMission()
{
    if(map==0){
        if(snake.size()/5>=1){
            move(10,maxwidth+1);
            printw("B: 1 (v)");
            mission1=1;
        }
        else{
            move(10, maxwidth+1);
            printw("B: 1 ( )");
        }
        if(fruitnum>=2){
            move(11, maxwidth+1);
            printw("+: 2 (v)");
        }
    }
}

```

- 설정된 기본 변수들의 변화에 의해 출력될 문장들이 선택되는 형식,
특히, map 별로 달성되어야 할 미션들이 다르므로 우선 어떤 map인지를 구분하는 if문
이 설정되어 있다., 4개의 미션 각각의 달성시 출력문에 mission1~4를 1로 바꿔주는
코드가 존재하며 이는 이후의 스테이지 전환 조건이 된다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

mission1=1;

move(10, maxwidth+1);
printw("B:  1 ( )");

tnum>=1){
  move(11, maxwidth+1);
  printw("+:  1 (v)");
  mission2=1;

  move(11, maxwidth+1);
  printw("+:  1 ( )");

  onnum>=1){
    move(12, maxwidth+1);
    printw("-:  1 (v)");
    mission3=1;

    move(12, maxwidth+1);
    printw("-:  1 ( )");


    num>=0){
      move(13, maxwidth+1);
      printw("G:  0 (v)");
      mission4=1;

```

→ 각각의 미션을 mission1,2,3,4라고 할 때 달성시 출력문 밑에 그 변수를 1로 선언한다.

→ 조건 달성시의 출력문 ()안에 v 표시가 들어간 출력문이 실행

→ 미션은 모두 4종류이며 각각의 미션은 mission(i)번의 변수를 가지며, 달성시 해당 변수의 값은 1로 지정

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```

mission=mission1+mission2+mission3+mission4;
//조건 달성시(mission=4일때, 미션이 전부 달성시) 증감, 종료
if(mission==4){
    map+=1;
    foodtime=0;
    poisonsime=0;
    randomtime = 0;
    if(map==4){
        direction='f';
    }else{
        direction='p';
    }
}

```

➤ PlayGame() 함수 내의 코드

MoveSnake() 이후 mission변수들의 합을 통해 현 스테이지의 미션들의 달성 여부를 판단한다. 달성시 map은 +1로 이동, 나머지 time변수들은 초기화시킨다.

그 후 만일 map변수가 값이 4가 된다면, 더 이상 진행할 스테이지가 없기 때문에 개별적인 축하 메시지를 실행시킬 direction = 'f' 상황으로 이동, 그렇지 않다면 다음 스테이지로의 이동을 실행시킬 direction = 'p' 상황으로 이동한다.

```

//4스테이지 까지 클리어 했을 시 실행 축하 메시지와 함께 게임 종료
if(direction=='f'){
    move((maxheight-2)/2, (maxwidth-5)/2-10);
    printw("Congratulation!!!, YOU ARE WINNER!!!");
    refresh();
    usleep(2000000);
    break;
}
if (direction=='q') //exit
{
    move((maxheight-2)/2,(maxwidth-5)/2+2);
    printw("GAME OVER");

    break;
}

```

➤ playGame()함수는 direction이 'f'일 시, 게임 클리어 축하 메시지를 출력 한다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```


//스테이지 미션 클리어시 실행, nextStage()함수로 루프 내에서 반복
if(direction=='p'){

    move((maxheight-2)/2,(maxwidth-5)/2-2);
    printw("Next Stage");
    refresh();
    //미션 충족시 next stage가 중앙에 뜨고 2초 뒤 자동으로 다음 스테이지
    usleep(2000000);
    clear();
    refresh();

    direction='l';
    nextStage();
}

```

- playGame()함수는 direction이 'p' 일시 다음 스테이지로의 이동을 출력, 화면을 초기화, 변수를 초기화하는 nextStage() 함수 실행.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

2.2.3 활용/개발된 기술

작성요령 (10 점)

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.

● NCURSES

커서를 움직이거나 윈도우를 생성하고, 색깔을 지정하고, 마우스 관련 함수를 제공
Star_color기능을 통해 각 요소들(Wall, Immune wall, Fruit, Poison, Snakebody, SnakeHead)
의 색깔을 통한 구분을 가능하게 함.

Refresh(), getch(), initscr() 등의 기능으로 대기화면 및, 사용자의 입력을 받아 전환을
할 수 있는 기능을 제공

Getmaxyx()의 기능으로 사용할 화면의 가로, 세로의 길이의 값을 얻어내어
구현하고자 하는 맵의 사이즈 및 좌표값들에의 이용을 통한 출력을 가능하게 함

Keypad(*win, bool) 방향키를 통하여 이동을 조정하게 될 것이니 bool 값의 true로 실행

● 임의의 값을 얻기위한 <ctime> header

#include <ctime> 헤더의 정의로 현재의 윈도우 내에 fruit, poison, random, gate의 좌
표를 찾기위해 사용, ncurses의 기능으로 설정한 maxwidth, maxheight를 활용한


Int a = rand()%maxwidth;

Int b = rand()%maxheight;

코드로 현재 윈도우 내의 임의의 좌표값을 얻어낼 수 있었고, 얻어낸 값을 조건식에 대
입함으로써 올바른 위치의 좌표값을 얻어냄

● Vector STL 사용

- Snake의 몸 요소 하나하나의 좌표값을 설정할 구조체
CharPosition의 설정

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```
//사용자 정의 데이터 구조체
struct CharPosition
{
    int x, y;
    CharPosition(int col, int row);
    CharPosition();
};
```

>x 와 y 값을 행, 열의 번호로 갖는 구조체 CharPosition이 설정되고 Snake의 body 각 요소하나하나에 해당될 좌표값을 가지게 된다.

- 구조체 CharPosition의 값을 요소로 담은 구조체 벡터 Snake vector의 설정


```
std::vector<CharPosition> snake; // 뱀 몸통담을 벡터
```

>snake라는 벡터를 구조체변수로 선언함으로써, snake의 요소 하나하나는 x와 y의 좌표값을 가지게 된다.

- Fruit, poison, random, gate의 좌표값을 담은 구조체의 선언

```
CharPosition fruit; // 구조체 변수 선언
CharPosition poison; // poison 구조체 변수 선언
CharPosition random;
CharPosition gate1; //선언할 두 개의 gate 좌표
CharPosition gate2;
```

>각자의 좌표값이 설정되어야 할 변수들 또한 x,y좌표값을 가지는 구조체로 선언되어야 한다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

2.2.4 현실적 제한 요소 및 그 해결 방안

작성요령 (5 점)


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

3단계 – Growth Item, Posion Item생성

- 처음에 과일과 독, 랜덤이 유지되고 있는 시간을 어떻게 재는지에 대해 처음에는 고민이 많았다. 그래서 코드를 집중해서 보던 중에 뱀이 일정시간마다 움직이는 방식을 어떻게 만들었는지를 다시 한 번 봤고 매번 일정시간마다 움직이니깐 뱀이 몇 번 움직였나 에 따라서 과일, 독 그리고 랜덤이 유지되고 있는 시간을 재면 되겠다 라고 판단했고 그에 따라 코드를 짜보니 음식을 먹지 않았다면 일정한 시간이 지난 뒤 사라지고 재생성되는 것을 확인되었다.
- Step3 개발 단계에서 음식 3개를 맵 상에 출현시키는 것은 쉬웠지만, 음식 각각을 랜덤 하게 독 또는 과일로 지정 시키는 것 은 어려웠다. 이를 해결하기 위해, 과일 1개와 독 1개는 고정으로 출현 시켰으며, 3번째 음식을 랜덤 음식으로 만들어, rand()함수를 사용하여 modulo값이 0 이면 과일, 1 이면 독과 같은 성질을 가지게 설정하였다.

4단계 – 게이트의 생성, 게이트의 작동

- 게이트의 생성을 위해서는 우선, 올바른 위치의 좌표를 얻는 작업이 우선되어야 한다. 주어진 윈도우의 좌표 범위 중, immune wall과 snake가 이동할 수 있는 부분을 제외한 곳 중에서 선택되어야 하며, 두번째 생성되어질 게이트의 좌표는 먼저 생성되어진 게이트 의 좌표와 같지 않아야하는 과정이 포함되어야 했다. 따라서, 스테이지 별로 맵을 작성하는 코드를 참조하여 불가능한 좌표에는 continue를 하여 올바른 좌표를 얻어내 게이트를 설정하였고, 두번째 게이트의 좌표 설정코드 바로 앞에 조건식 하나(앞 게이트와의 좌표가 같으면 continue)를 추가하여 두번째 게이트가 겹치지 않도록 구성하여 해결하였다.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

- 게이트의 작동을 위해서는 우선, snake의 게이트와의 접촉 여부를 판단하는 작업이 필요하다, MoveSnake()함수 내에 snake[0].x와 snake[0].y의 값이 설정된 게이트의 좌표와 같은지를 비교하는 작업이 추가되어야 했으나.
게이트의 생성을 두 함수로 나누어 작성하였기에 각각의 게이트에 대한 진입을 구분해야 하는 문제가 발생했고, 이는 단순하게 코드의 반복을 통해 해결하였다.
추가적으로, 작동 원리 중 게이트의 진입과 출력시 방향에 대한 조건이 필요하지만 규칙을 찾을 수 없었고, 결국 각 진입방향 {'l', 'r', 'u', 'd'}에 대한 반대편 출력 게이트의 위치에 따른 케이스를 모두 구분하여 전부 예외 처리 형식으로 작성 할 수 밖에 없었다.
방법을 찾지 못한 만큼, 이를 올바르게 않은 방식으로 해결하기 위해 직접 장시간 테스트 과정을 거치며, 버그를 전부 찾아내고 고치는 식으로 해결하였다.

2.2.5 결과물 목록

작성요령 (5 점)

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

SnakeGame.cpp -> main 함수에서 실행되어질 SnakeGame 클래스의 함수들의 기능을 정의한다.

SnakeGame.h -> 사용될 header 파일들의 선언과 좌표로 사용될 구조체 및 SnakeGame 클래스 및 작동 함수들을 선언

Main.cpp -> 선언된 header 파일을 사용하여 실행할 main 함수를 정의하고 그에 필요한 추가적 함수들을 선언.

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

3 자기평가


작성요령 (5 점)

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.

20161943 최장혁 - 우선 팀 단위로 이정도 규모의 프로젝트를 수행하는게 처음이었으며, 많은 시행착오가 있었다. 초반에 기본 토대가 되는 뱀 게임의 구현을 각자 제작하다 보니 상대방의 코드를 이해하는 것에 많은 시간이 들어갔다. 하지만 서로 주석을 사용함으로써 이를 해결할 수 있었다. 나는 음식과 독을 구현하였으며, 과일은 이미 구현한 상태여서 독을 구현하는 것에는 그렇게 오랜 시간이 걸리지 않았다. 하지만 음식을 3개 까지 출력되도록 구현하는 것과 출력되는 음식이 임의로 과일 또는 독으로 정해져야 하는 부분에서 많은 시간과 노력이 필요했다. 내가 막힌 부분을 조원들이 도와주며 조언을 해주어 힘들게 이를 구현 하였다. 협력을 통해 더 효율적이고, 새로운 해결책을 찾을 수 있는 좋은 경험이었다. 다만 아쉬웠던 점은, 후반부에 어려운 구현 부분으로 갈수록 코딩경험이 적어 개발 부분에 많이 참가하지 못하였다. 계속해서 코딩 공부를 하여 다음에는 더 비중 있는 조원이 되도록 노력해야함을 느꼈다.

20163159 조성래 - 이번 프로젝트에서, 맵의 구현, 및 게이트의 생성, 작동을 담당하는 4단계와 점수와 미션을 출력하는 5단계의 구현을 담당했습니다. 프로젝트를 진행하며 맡게 된 부분을 구현하는 과정에서 생각보다 요구하고 있는 조건이 많고 복잡하던 것을 느꼈으며, 실제로 그것을 구현하는 과정에서 올바르게 달성하지 못했습니다. 비록 올바른 방법이 아니긴 하지만 포기하고 넘겨 버리기엔 책임감이 느껴져 단순히 코드가 길어지고 난잡해지는 방법일지라도 시간을 투자하여 실행해보고, 버그를 일일이 찾아내고 수정하는 과정을 통해 성공시켜내는 과정을 겪으면서 미숙한 실력일지라도 내가 할 수 있는 일이 있다는 것을 느낄 수 있었습니다.

20161934 인재휘 - 항상 혼자 코딩을 했더니 팀 단위로 하나의 코딩을 하는 게 익숙하지 않았다. 그래서 같은 코드를 작성했지만 뭔가 더 효율적으로 짜지 못 했던 거 같고, 같은 모양을 만들었음에도 더 보기에 편하지 못 했던 거 같아 내 자신이 실망스러웠다. 나는 첫 초안 짜기와 맵의 색과 스테이지를 넘기는 법, 그리고 과일과 독이 일정 시간이 지나면 사라지는 부분을 맡았다. 프로젝트를 짜면서 어려웠던 건 각자의 첫 초안 중 누구의 것을 이용해서 다음 단계를 넘어갈 것인가와 매주 대화가 끝났을 때 서로의 것을 이해하는 부분이 힘들었다. 하지만 항상 코드에 주석이 달려있었고 이해가 잘 안 되는 부분을 질문하면 잘 알려주기에 많은 도움이 되었다.

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

4 참고 문헌

| 번호 | 종류 | 제목 | 출처 | 발행년도 | 저자 | 기타 |
|----|--------|---|---|------------|--|--|
| 1 | 웹페이지 | http://tldp.org/HOWTO/NCURSES-Programming-HOWTO/ | Pradeep Padala | 2005-06-20 | ppadala@gmail.com | |
| 2 | 웹페이지 | (Unix C)NCURSES 란? | 전산학도 이야기 - 블로그 | 2009-10-5 | 위디안 | |
| 3 | 웹 pdf | Unix/Linux 화면 조작 (curses.h) | http://blog.naver.com/kamvo.do?Redirect=Log&logNo=60003596240 | 2004-6-29 | 삼겹살조아 | |
| 4 | Github | C++ implementation of a terminal based snake game, using ncurses library | Github | 2017-1-16 | romanedgn | |
| 5 | 웹 페이지 | Ncurses 프로그래밍 | https://psman2.tistory.com/entry/ncurses-프로그래밍 | 2011-5-2 | 윤상배 | dreamyun@yahoo.co.kr |

5 부록

작성요령 (15 점)

프로젝트의 결과물을 사용하기 위한 방법에 대해서 작성하세요.

● Ncurses 준비

■ Ubuntu 기준: 라이브러리를 설치한다.

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


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

● 컴파일 및 실행방법

```
$ g++ -o SnakeGame.cpp main.cpp -lncurses
```

```
$ ./a.out
```

또는

| | | | |
|---|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

```
$ g++ -o (원하는파일명) SnakeGame.cpp main.cpp -lncurses
$ ./(원하는파일명).out
```

5.1 사용자 매뉴얼

프로젝트 실행 후 사용안내, 사용자 매뉴얼

파일 실행 후 기본 방향 'left' 로 이동하는 snake 를 키보드 화살표를 이용하여 조종 합니다.

맵의 곳곳에 임의로 생성될 Fruit(녹색)을 획득하여 몸의 길이를 늘리고 우측 상단의 미션들을 달성합니다. 특히, Poison(빨간색)을 주의하세요. 습득 시 길이가 -1 만큼 줄어듭니다.

맵의 Fruit 과 Poison 은 각각 50 점의 추가점수와 -50 점의 감점을 줍니다.

점수가 100 단위씩 늘어날 때마다 snake 의 속도는 점점 빨라지는 것에 주의합니다.


만일 snake 가 너무 길어져서 자신의 몸통에 부딪히거나, 맵의 벽에 충돌할 경우 Game Over.

마찬가지로 Poison 을 너무 많이 습득하여, Snake 의 길이가 3 이하가 되어도 Game Over

특정 시간이 지나면 맵의 임의의 벽에 게이트가 생성이 됩니다. 게이트는 두개가 한 쌍으로 나타나며 어느 한쪽으로 진입시, 그 반대편으로 snake 가 나올 것입니다.

총 4 stage 의 map 을 돌아다니며 실패조건을 피해 미션을 달성하면 승리입니다.

If you hit the borders of the game window or if the snake collapse into his own body, it's game over

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

5.2 설치 방법.

1. 파일을 다운받고 SnakeGame 의 cpp 파일, h 파일, main.cpp 파일 3 가지가 있는지 확인

```
cho@cho-VirtualBox:~/문서$ ls
SnakeGame.cpp  SnakeGame.h  a.out  main.cpp
cho@cho-VirtualBox:~/문서$
```

2. 받은 파일들을 컴파일하여 out 파일 생성

```
cho@cho-VirtualBox:~/문서$ g++ -o SnakeGame.cpp main.cpp -lncurses
```


- header 파일을 포함한 cpp 파일과 main.cpp 파일을 포함

3. 생성된 a.out 파일을 실행

```
cho@cho-VirtualBox:~/문서$ ls
SnakeGame.cpp  SnakeGame.h  a.out  main.cpp
cho@cho-VirtualBox:~/문서$ ./a.out
```

4. 게임 시작을 알리는 문구. Y 를 통해 시작, N 을 통해 종료

```
Snake Game, Are you ready? press Y or N
```

| | | | |
|--|-------------------------|----------------------------|------------|
|  국민대학교 컴퓨터공학부 캡스톤 디자인 I | 결과보고서 | | |
| | 프로젝트 명 | SnakeGame.cpp with ncurses | |
| | 팀 명 | 3분반 8조 | |
| | Confidential Restricted | Version 1.1 | 2020-06-20 |

5. 최종 실행시 화면

