**C++프로그래밍**

**프로젝트**

|  |  |
| --- | --- |
| 프로젝트 명 | 스네이크 게임 프로젝트 |
| 팀 명 | 팀 Q |
| 문서 제목 | 결과보고서 |

|  |  |
| --- | --- |
| **Version** | 1.0 |
| **Date** | 2021-Jun-19 |

|  |  |
| --- | --- |
| **팀원** | 임지원 (팀장) |
|  |
|  |
|  |
|  |

|  |
| --- |
| **CONFIDENTIALITY/SECURITY WARNING**  이 문서에 포함되어 있는 정보는 국민대학교 소프트웨어융합대학 소프트웨어학부 및 소프트웨어학부 개설 교과목 C++프로그래밍 수강 학생 중 프로젝트 “스네이크 게임 프로젝트”를 수행하는 팀 Q의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 Q의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다. |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | 최종보고서-스네이크게임.doc |
| **원안작성자** | 임지원 |
| **수정작업자** | 임지원 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2021-06-19 | 임지원 | 1.0 | 최초 작성 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**목 차**

[1 개요 4](#_Toc43103653)

[2 개발 내용 및 결과물 5](#_Toc43103654)

[2.1 목표 5](#_Toc43103655)

[2.2 개발 내용 및 결과물 6](#_Toc43103656)

[2.2.1 개발 내용 6](#_Toc43103657)

[2.2.2 시스템 구조 및 설계도 6](#_Toc43103658)

[2.2.3 활용/개발된 기술 6](#_Toc43103659)

[2.2.4 현실적 제한 요소 및 그 해결 방안 6](#_Toc43103660)

[2.2.5 결과물 목록 7](#_Toc43103661)

[3 자기평가 8](#_Toc43103662)

[4 참고 문헌 8](#_Toc43103663)

[5 부록 8](#_Toc43103664)

[5.1 사용자 매뉴얼 8](#_Toc43103665)

[5.2 설치 방법 8](#_Toc43103666)

# 개요

Ncurses

Snake game(main)

본 프로젝트는 위의 구조와 같이 NCurses 라이브러리를 사용하여 Snake Game을 구현한 2021년 국민대학교 소프트웨어학부 C++ 강의 기말 프로젝트이다. 본 프로젝트의 Snake Game은 map, snake, item, gate, 점수 표현(score) 즉, 크게 5가지의 구성 요소로 나뉜 구조로 이루어져 있다.

Score

gate

item(growth, poison)

Snake(Head+Body)

map

Snake Game은 먼저 4가지의 스테이지가 존재한다. 각 스테이지 별로 다른 형태의 map이 사용되고, 점수 표현을 제외한 모든 것들은 전부 map 안에서 구현된다. Head와 여러 개의 Body로 이루어진 snake가 map의 빈 공간에서 진행 방향을 입력 받아 움직이게 된다. 또한 map의 빈 공간에 item인 growth와 poison이 나타나게 되는데snake의 head 부분이 각각을 지날 때 body를 증가시키고 감소시키는 역할을 하게 된다. gate는 map의 wall 부분에서 생성되는데 이를 snake가 통과할 수 있다. map 외부에 있는 Score은 현재 얻은 점수들과 각 스테이지 별로 달성해야 하는 목표를 나타내 다음 스테이지로 넘어가기 위한 도움을 준다. 게임 오버 처리를 당하게 되면 스테이지를 중간에서부터 재시작할 수 없으며, 스테이지 4까지 모두 클리어하면 Snake Game을 최종 클리어할 수 있게 된다.

Ncurses 라이브러리를 사용하여 입출력을 표현해 본 프로젝트를 구현하였기 때문에 Ncurses 설치가 필요하다.

(Ncurses 설치 : sudo apt-get install libncurses5-dev libncursesw5-dev)

# 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 |

**1단계 목표**

1. NCUrses Library 함수들을 사용하여 2차원 배열로 된 Snake Map을 Game 화면으로 표시하는 프로그램을 완성한다. Map은 21x21을 최소 크기로 한다.
2. Wall과 Immune Wall을 구분한다.

- Map Data에서 Wall은 1, Immune Wall은 2과 같이 구분할 수 있도록 한다.

**2단계 목표**

1. 1단계의 map 위에 Snake를 표시하고, 방향기를 입력 받아 Snake가 움직이도록 프로그램을 완성한다.

- Map Data에서 Head는 3, Body는 4과 같이 구분할 수 있도록 한다.

**3단계 목표**

1. 2단계 프로그램에서, Map 위에 Growth Item과 Poison Item을 출현하도록 수정한다.
2. Growth Item과 Poison Item을 Map 배열에 표현할 때 값을 정한다.

- 화면 상에 표현 시, 색이나 기호를 달리하여 구분할 수 있도록 한다.

- Map Data에서 Growth Item은 5, Poison Item은 6과 같이 구분할 수 있도록 한다.

**4단계 목표**

1. 3단계 프로그램에서, Map의 Wall의 임의의 위치에 한 쌍의 Gate가 출현할 수 있도록 변경하고, 각 Gate에 Snake가 통과할 수 있도록 수정한다.
2. Wall(Immune Wall 포함)과 Gate를 Map 배열에 표현할 때 값을 결정한다.

- 화면 상에 표현 시, Gate는 Wall과 구분될 수 있도록 한다.

- Map Data에서는 Gate는 7과 같이 구분할 수 있도록한다.

**5단계 목표**

1. 4단계 프로그램에서, 우측에 게임 점수를 표시하는 화면을 구성한다.
2. Mission - 매 게임마다 임의의 값을 준다.
3. Mission을 달성하면 다음 Map으로 진행하도록 프로그램을 완성한다. Stage는 최소 4개로 구성하고, 각 Stage의 Map은 서로 달라야 한다.

## 개발 내용 및 결과물

### 개발 내용

**1단계 개발 내용**

각 Stage 별로 다른 게임 화면 맵을 나타내기 위해 2차원 배열 형태의 4가지의 map을 구현했다. 이 배열을 통해 map의 형태가 x, y 즉, 화면상으로 보이는 좌표대로 편리하게 저장되어 있음을 알 수 있다. 현재 stage가 몇 번인지를 통해 해당하는 map을 show 함수에서 ncurses의 printw을 이용하여 화면상에 표시하였다.

**2단계 개발 내용**

Snake의 Head 및 Body를 나타내고자 Snake 클래스를 구현했다. Snake 클래스에서는 Head와 Body들의 각각의 좌표를 저장한다. Ncurses의 keypad(stdscr, true)와 getch()를 통해 사용자로부터 진행 방향 입력을 받고, move 함수에서 입력 받은 방향에 의해 Snake의 좌표 값이 이동되어 Snake가 이동할 수 있도록 하였다. 이때 Snake가 map 밖으로 나가게 되거나 자신의 Body를 통과하는 경우 게임 오버로 종료되도록 구현하였다. nodelay(stdscr, true)을 이용해 입력을 계속 받지 않아도 계속해서 게임이 진행될 수 있도록 하였고, snake\_s라는 변수와 usleep()을 이용해 snake의 속도를 조절했다.

**3단계 개발 내용**

Snake의 Head 부분이 Growth Item과 Poison Item에 닿아 Item을 먹게 되었을 때 각각 Body 추가되고, 감소되도록 구현하였다. Poison Item을 먹었을 때는 감소된 결과에서 Snake의 길이가 3보다 작아졌는지를 체크해 게임 오버 처리하도록 구현했다. get\_growth, get\_poison 함수를 만들어 각각 나타날 item의 x, y의 좌표를 map의 wall이 아닌 빈 공간에서 랜덤으로 생길 수 있도록 구현하였다. move 함수에서 Item을 먹게 되면 먹은 Item은 사라지고 새로 Item이 나타나고, 일정 시간이 지나면 Item이 사라지고 또 새로 Item이 나타나도록 구현하였다. init\_pair(), attron(), attroff()으로 Wall, Item, Snake들을 각기 다른 색으로 표현해 구분할 수 있도록 하였다.

**4단계 개발 내용**

get\_gate 함수를 만들어 Immune wall이 아닌 그냥 wall에만 gate가 생기도록 하고, gate 2개의 x, y 좌표가 랜덤으로 정해져 나타날 수 있도록 구현하였다. move 함수에서 gate를 통과하게 되는 경우 snake가 들어가 입구가 된 gate를 제외하고 남은 다른 gate를 출구로 취급하여 입구에서 출구로 snake가 나올 수 있도록 하였다.

**5단계 개발 내용**

set\_stage에서 랜덤으로 미션에 임의의 값을 주도록 구현하였다. show에서 현재 상태와 미션, 미션 성공 여부를 map의 오른쪽에 구현하였다. item을 먹거나 gate를 통과할 때마다 점수 변동이 생긴 것이 이를 통해 나타난다. mission을 달성하면 다음 stage로 넘어가게 되고, 4스테이지를 넘어가게 되면 모든 스테이지를 깼다는 결과가 화면 상에 나타나도록 구현하였다.

### 시스템 구조 및 설계도

**1단계**

int map[4][21][21]이라는 3차원 배열을 이용하여 각 스테이지 별로 2차원 배열로 이루어진 map을 사용하였다. 기본적인 map은 빈 공간 0, wall 1, Immune wall 2로 나타낸다. void show()에서 map이 빈 공간인지 wall인지 파악하여 Ncruses의 printw()로 나타낸다.

**2단계**

Snake 클래스에서 Snake의 Head와 각 Body의 좌표를 각각 저장하고 이를 하나의 list로 묶는다. Ncurses의 keypad(stdscr, true)와 getch()를 통해 사용자로부터 입력 받은 진행 방향을 토대로, move 함수에서는 head의 좌표를 받아와 값을 증감해 새로 이동한 좌표를 얻어 맨앞에 새로 추가하고, 맨뒤에 있는 값을 하나 삭제해 snake가 이동할 수 있도록 하였다.

**3단계**

move 함수에서는 이동 전 tail의 좌표를 따로 저장하고, Snake가 Growth Item을 먹었을 경우 삭제되었던 tail 좌표를 다시 추가해 body의 길이가 1 증가하도록 하고, Poison Item을 먹었을 겨우에는 현재 상태에서 snake의 마지막 좌표를 제거해 body의 길이가 1 감소하도록 하였다. 이때 각각 item이 새로 생기도록 get\_growth(), get\_poison()을 다시 실행하였다. growth\_s, poison\_s 라는 변수를 만들어 각각의 생성 속도를 조절하였다. Ncurses의 init\_pair(), attron(), attroff()을 사용하여 화면 상에서 snake, item, wall을 색으로 쉽게 구분할 수 있도록 하였다.

**4단계**

get\_gate 함수에서 Immune wall이 아닌 그냥 wall에만 2개의 gate가 만들어진다. move 함수에서 snake가 한 gate를 통과하게 되면 다른 gate는 출구가 되어 이를 통해 다시 snake가 나오게 되는데 gate가 가장자리에 있을 때, 가장자리가 아닌 경우 진입 방향과 일치하는 방향, 시계 방향으로 회전하는 방향, 역시계 방향으로 회전하는 방향, 반대 방향의 우선 순위로 snake가 빠져나올 수 있도록 구현하였다. 이때 Snake의 몸이 다 통과할 때까지 gate가 새로 변경되지 않도록 현재 snake의 길이를 카운트해 카운트 값을 snake의 좌표가 바뀔 때마다 -1해 0이 될 때 새 gate가 생기도록 구현하였다.

**5단계**

set\_stage 함수에서 각 스테이지가 시작할 때마다 랜덤으로 미션을 지정하고, show 함수에서 Score Board, Mission Board를 나타냈다. body.size(), max\_length, item\_g, item\_p, use\_g으로 점수를 goal\_b, goal\_gr, goal\_p, goal\_g로 미션 상황을 나타내고자 하였고 이를 mvprintw(), printw()로 표현하였다.

### 활용/개발된 기술

Ncurses : 텍스트 모드에서 UI를 구현할 수 있는 프레임 워크를 제공한다.

list : snake를 이루는 각 body를 묶기 위해 사용

time : 게임을 실행할 때마다 item, gate의 좌표 변경이 매번 똑같은 순서로 나타나지 않도록 방지하기 위하여 사용, 좌표를 랜덤으로 나타내기 위해 사용

unistd : usleep()을 통해 게임 속도를 조절하기 위해 사용

### 현실적 제한 요소 및 그 해결 방안

stage 별로 맵이 다르기 때문에 생성 위치와 처음 진행 방향까지 모두 무작위일 경우 시작하자마 게임 오버가 될 수 있다는 한계가 있다. 이 때문에 처음 생성 위치와 진행 방향은 임의로 지정하였다.

### 결과물 목록

snakegame.cpp

- Ncurses를 사용하여 만든 snake game project c++ 파일

# 자기평가

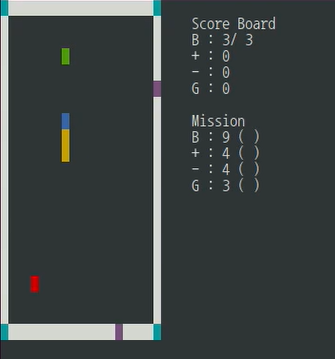
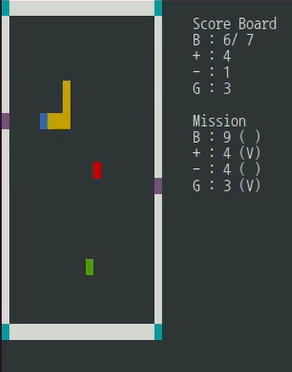
C++언어로 처음 프로젝트를 진행하게 되기도 하였고, 혼자서 snake game 프로젝트를 구상하고, 프로그램을 제작해야 하니 걱정을 많이 하게 되었다. 걱정 이후에는 미리 정리해둔 구상을 바탕으로 책과 구글링을 통해 필요한 라이브러리, 함수들을 정리하고 이를 통해 단계 별로 생각한 결과 값이 나올 때까지 여러 번 컴파일을 거쳤다. 프로젝트에 시간을 좀 더 투자했기 때문에 그래도 제 시간 내에 필요한 기능들을 모두 구현해낼 수 있었다. 시간 상의 문제로 새로운 추가 기능을 많이 넣어서 snake game을 만들지 못 해 아쉬움이 크지만, 제 시간 안으로 기본 기능을 모두 구현해내서 만족스럽고, 이후에 따로 추가 기능을 넣은 snake game을 다시 만들어 내고자 한다.

# 참고 문헌

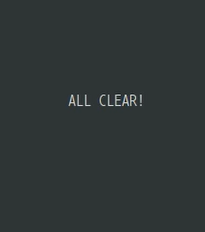
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 번호 | 종류 | 제목 | 출처 | 발행년도 | 저자 | 기타 |
| 1 | 책 | 윤성우의 열혈 C 프로그래밍 |  | 2010 | 윤성우 |  |
|  |  |  |  |  |  |  |

# 부록

## 사용자 매뉴얼

처음 시작 시 stage가 몇 번째인지 나타내며 잠시후 자동으로 게임 화면으로 전환되고, 방향키를 통해 snake를 움직일 수 있게 됩니다. q를 누르면 게임을 강제 종료시킬 수 있습니다.

초록색은 growth로 먹으면 body가 늘어나고 빨간색은 poison으로 body가 줄어들고 보라색은 gate로 한 곳으로 들어가면면 다른 곳으로 나오게 됩니다. 우측의 mission을 성공하면 다음 스테이지로 넘어가게 됩니다.

stage 4까지 성공 시 all clear가 나타나고 종료하게 됩니다. 벽을 통과하거나 자신의 몸을 통과하게 되거나 snake의 body가 3보다 작아지는 경우 게임은 그전에 게임 오버로 종료됩니다.

## 설치 방법

컴파일 : g++ -o sg snakegame.cpp -lncurses

실행 : ./sg