




국민대학교
소프트웨어융합대학
소프트웨어학부

C++프로그래밍 프로젝트

프로젝트 명	Snake Game Project
팀 명	유선종, 양성민
문서 제목	결과보고서

Version	1.2
Date	2021-JUN-05

팀원	유선종 (팀장)
	양성민

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05


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 "유선중, 양성민"의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 "유선중, 양성민"의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

문서 정보 / 수정 내역


Filename	최종보고서-Snake Game Project.doc
원안작성자	유선중
수정작업자	양성민

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2021-06-17	유선중	1.0	최초 작성	개발 내용 및 부록 작성
2021-06-18	양성민	1.1	내용 수정	수정된 내용 추가
2021-06-19	유선중,양성민	1.2	최종 수정	최종 수정 및 검토

 국민대학교 컴퓨터공학부 C++ 기말 프로젝 트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

목 차

1	개요	4
2	개발 내용 및 결과물	5
2.1	목표	5
2.2	개발 내용 및 결과물	7
2.2.1	개발 내용	7
2.2.2	시스템 구조 및 설계도	10
2.2.3	활용/개발된 기술	18
2.2.4	현실적 제한 요소 및 그 해결 방안	19
2.2.5	결과물 목록	19
3	자기평가	21
4	참고 문헌	21
5	부록	22
5.1	사용자 매뉴얼	22
5.2	설치 방법	23

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

1 개요

평가기준 (10점)

프로젝트를 완성하기 위해 사용한 개발 방법을 기술하세요.

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

간단하게 구성된 Snake를 방향키를 조작하여 Item을 획득하거나 Gate를 사용하면서 주어진 Mission을 수행하는 게임으로 ncurses, vector, cstdlib, ctime, cmath 등을 사용했고, 특히 ncurses는 터미널 환경에서 \$sudo apt-get update와 \$sudo apt-get install libncurses5-dev libncursesw5-dev를 입력해 다운받아야 코드에서 사용이 가능하고 나머지는 코드에서 #include를 통해 사용이 가능하다.

전체적인 Map과 Score Board, Mission Board 등 사용자 UI는 ncurses를 사용해 구현하였고, vector를 사용해 Snake의 상태 및 동작 관리를 구현하였다. 또한 cstdlib를 사용해 Item과 Gate를 무작위 위치에서 생성시키기 위하여 rand()를 사용하기 위해 사용했고, Item의 무작위 생성 주기인 5초를 설정하기 위하여 ctime을 사용하였다.

Game을 원활하게 진행하기 위하여 8개의 파일이 필요한데 Map 정보를 구현하는 map.h, map.cpp가 있고, Snake의 상태 및 동작을 구현하는 snake.h와 snake.cpp, 전체적인 게임의 동작들과 기능을 구현하는 manage.h와 manage.cpp, 마지막으로 모든 기능의 최상위 함수인 Run() 함수를 호출하는 main.cpp와 헤더파일과 소스파일을 전부 컴파일해주는 Makefile로 이루어져 있다.

8개의 파일을 다운받고 다운받은 디렉토리에서 make를 진행한 후, ./main.out을 입력하면 Gmae을 시작할 수 있다. 모든 미션을 클리어하면 "Mission Complete"가 출력되고, 현재 진행 방향의 반대 키를 누르거나, Snake의 길이가 3보다 작아지거나, Snake의 머리가 몸통에 닿거나, Snake의 머리가 벽에 닿는 상황으로 게임이 중간에 종료된다면 "Game Over"가 출력되면서 게임이 종료된다.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

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

2.1.1 게임 규칙

- 규칙1

1. Snake 는 진행 방향의 반대 방향으로 이동할 수 없다.
2. Snake 는 자신의 Body 와 벽(Wall)을 통과할 수 없다.
3. Head 방향 이동은 일정 시간(틱)에 의해 이동한다.

- 규칙 2


1. Snake 의 이동 방향에 Item 이 놓여 있을 수 있다.
 - 1.1 Growth Item 을 획득하면 몸의 길이가 1 증가한다.
 - 1.2 Poison Item 을 획득하면 몸의 길이가 1 감소한다.
 - 1.2.1 몸의 길이가 3보다 작아지면 게임을 종료한다.
2. Growth Item 과 Poison Item 의 출현
 - 2.1 Snake Body 가 있지 않은 임의의 위치에 출현한다.
 - 2.2 출현 후 일정시간이 지나면 사라지고 다른 위치에 나타난다.
 - 2.3 동시에 출현 할 수 있는 Item 의 개수는 3개로 제한한다.

- 규칙3

1. Gate 는 두 개가 한 쌍이다.
2. Gate 는 겹치지 않는다.
3. Gate 는 임의의 위치에 있는 벽에서 나타난다.
4. Gate 에 Snake 가 진입 중인 경우 Gate 는 사라지지 않는다.
5. Gate 에 Snake 가 진입 중인 경우 다른 위치에 Gate 가 나타나지 않는다.

- 규칙4

1. Gate 가 나타나는 벽이 Map 경계에 있을 때

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

1.1 항상 Map 의 안쪽 방향으로 진출한다.

예) 상단 벽 -> 아래 방향

2. Gate 가 나타나는 벽이 Map 경계 안 쪽에 있는 벽에 있을 때

2.1 상하좌우를 탐색하여 벽이 없는 공간으로 빠져 나온다.

- 규칙5

1. Wall - Gate 로 변할 수 있다.

2. Immune Wall - Gate 로 변할 수 있다.

3. 모든 Wall - Snake 가 통과할 수 없다.

4. Snake Head 와 충돌 시 Game Over

5. Gate 의 출현 방법 결정

5.1 게임 시작 후 일정 시간(약 5초) 후 출현하도록 한다.

- 규칙6

1. 점수 계산

1.1 게임 중 몸의 최대 길이 계산

B: (현재 길이) / (최대 길이)

1.2 게임 중 획득한 Growth Item 의 수

1.3 게임 중 획득한 Poison Item 의 수

1.4 게임 중 Gate 사용 횟수

1.5 게임 시간 - seconds 로 계산한다.

2. 게임 방법

2.1 주어진 미션을 달성한다.

3. 미션

3.1 좌측의 각 점수 항목 별로 목표치 도달 시 게임을 종료한다.

2.1.2 1 단계 개발 목표

1. Ncurses Library 함수들을 사용하여 Snake Map 을 Game 화면으로 표시하는 프로그램을 완성한다.

2. Wall 과 Immune Wall 을 잘 구분한다.

2.1.3 2단계 개발 목표

1. 1단계의 Map 위에 Snake 를 표시하고, 화살표(방향키)를 입력 받아 Snake 가 움직이도록 프로그램을 완성한다.

2. Snake 는 규칙1을 준수한다.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

2.1.4 3 단계 개발 목표

1. 2단계 프로그램에서 Map 위에 Growth Item 과 Poison Item 을 출현하도록 한다.
2. 게임 규칙 2를 준수한다.
3. Growth Item 과 Poison Item 을 Map 에 표현할 때 값을 정한다.
 - 3.1 화면 상에 표현 시, 색이나 기호를 달리하여 구분하도록 한다.

2.1.5 4 단계 개발 목표

1. 3단계 프로그램에서, Map 의 Wall 의 임의의 위치에 한 쌍의 Gate 가 출현하도록 변경하고 각 Gate 에 Snake 가 통과할 수 있도록 한다.
2. 게임 규칙 3, 4, 5를 준수한다.
3. Wall(Immune Wall 포함)과 Gate 를 Map 에 표현할 때 값을 정한다.
 - 3.1 화면 상에 표현 시, Gate 는 Wall 과 구분될 수 있도록 한다.
 - 3.2 Map 에서 Gate 는 7과 같이 하여, 다른 요소와 구분할 수 있도록 한다.

2.1.6 5단계 개발 목표

1. 4단계 프로그램에서, 우측에 게임 점수를 표시하는 화면을 구성한다.
2. 게임 점수는 게임 규칙 6을 준수한다.
3. Mission
 - 3.1 구성된 Map 의 정의에 고정 값을 주거나
 - 3.2 매 게임마다 임의의 값을 주는 방식으로 처리한다.
4. Mission 을 달성하면 다음 Map 으로 진행하도록 프로그램을 완성한다.
 - 4.1 Stage 는 최소 4개로 구성하고, 각 Stage 의 Map 은 서로 달라야 한다.

2.2 개발 내용 및 결과물

2.2.1 개발 내용

작성요령 (10점)

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

1. 1단계 개발 내용

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

Map 구현을 위한 Map 클래스를 구현하였다. 클래스에는 게임 맵을 화면 상에 표현하기 위해 각각 윈도우 별로 함수들을 정의하였고 윈도우마다 wborder() 함수를 통해 Wall에 해당하는 영역은 문자 'X'로 Immune Wall에 해당하는 영역은 문자 '+' 로 표현하였다. 추가로, 맵의 Width와 Height를 동시에 처리하는 구조체를 선언하여 이후 단계 개발 목표인 Snake의 생성과 Item의 출현, Gate의 출현 프로그램 개발에 활용하였고 stage(level)가 진행됨에 따라 맵 경계 안에 Wall가 출현하도록 하는 함수를 또한 정의하였다.

2. 2단계 개발 내용


Snake 구현을 위한 Snake 클래스를 구현하였다. Snake를 게임 맵 내부에 출력하기 위해 map.h 파일을 include하여 게임 맵의 정보를 이용할 수 있도록 하였다. 생성자에서 keypad() 함수를 통해 게임 시작 시 사용자의 입력을 받아 Snake가 이동할 수 있도록 하였고 nodelay() 함수를 이용하여 사용자의 입력 없이도 Snake가 움직일 수 있도록 하였다. Snake의 이동은 moveSnake() 함수에서 사용자의 입력과 Snake의 현재 이동 방향에 따라 Snake가 이동하도록 하였다. 사용자의 입력이 Snake의 진행 방향과 반대인 경우, 즉시 게임이 종료되도록 하였다.

추가로, status 변수를 활용하여 Snake가 Item을 획득한 상태인지 아닌지 구분하도록 하였고 이에 따라 Snake의 전반적인 이동을 조정할 수 있게 하였다. 게임 운영을 위한 Manage 클래스를 정의하고 내부에 Wall과 Gate를 판단하는 함수와 Snake의 Body와 Tail을 판단하는 함수를 정의하여 규칙1을 준수하도록 하였다.

3. 3단계 개발 내용

Manage 클래스에 Item의 생성과 획득을 처리하는 함수를 정의하였다. 이를 위해 Snake에 대한 정보와 게임 맵 내부에 Item 출현을 위해 게임 맵에 대한 정보가 필요했고 이를 위해 snake.h 파일(map.h 파일 포함)을 include하여 Snake 정보와 맵 정보를 이용하고 임의의 좌표에 item이 출현하고 획득할 수 있도록 하였다. 해당 기능은 Growth Item 함수와 Poison Item 함수를 정의하여 규칙2에 맞게 Item이 출현 되도록 하였다. 임의의 좌표에 Item을 출현하도록 하기 위해 ctime 라이브러리와 cstdlib라이브러리의 rand() 함수를 이용하였다. Item의 출현은 게임을 시작하자마자 생성될 수 있게 Manage 클래스의 생성자에 해당 함수를 호출하도록 하였고 ctime 라이브러리를 이용하여 이후에도 5초마다 item이 다른 위치에 다시 생성되도록 하였다.

다음으로, 3단계 개발 목표의 3 번째 목표를 수행하기 위해 함수를 정의하여 Snake의 Item 획득

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

득 여부를 판단하였다. Snake가 어떠한 Item을 획득했는지 에따라 status를 변경하여 Snake의 길이를 1씩 증감시켰다. 또한, 게임 규칙2의 Snake의 길이가 3보다 작아지는 경우를 충족하기 위해 같은 매 프레임마다Snake의 size를 판단하도록 처리하였다. 마지막으로 각 Item을 획득한 경우, 곧바로 새로운 위치에 Item이 출현하도록 각 Item에 해당하는 함수가 실행되도록 하였다.


4. 4단계 개발 내용

Manage 클래스에 Gate 생성과 통과를 처리하는 함수를 정의하였다. Gate의 출현은 방식은 규칙 3과 5를 준수하면서 Item 출현과 동일한 방식으로 하되 조건을 달리하여 Immune Wall을 제외한 맵 경계에 해당하는 Wall과 경계 안에 해당하는 Wall에 출현하도록 하였다. Gate는 Wall과 같은 문자 'X'를 사용하여 표현하였고 구분을 위해 색상을 노란색으로 설정하였다. Gate의 출현은 게임 시작 후 일정 시간 후에 생성되도록 gate_time 변수를 이용하여 처리하였고 이후 stage(level)가 진행될 때마다 새로운 임의의 위치에 Gate가 다시 출현하도록 설정하였다.

Snake가 Gate를 통과하도록 처리하는 passGate()함수를 정의하였다. 해당 함수는 매 프레임마다 호출되지만 Snake의 Head 좌표와 Gate의 좌표의 일치 여부를 판단하는 조건문에 따라 함수 실행 여부를 결정한다. 함수가 실행되면 Snake 클래스의 moveSnake() 함수와 동일한 방식으로 진행하되 어떤 Gate로 들어왔는지에 따라 Snake가 나오는 Gate를 구분이 필요했기 때문에 각 Gate의 좌표를 이용하였다. 이후, checkOutputGate() 함수를 호출하여 다른 Gate에서 Snake가 출현하도록 처리하였다. 이때, Snake의 출현 위치와 진행 방향은 규칙 4를 준수하여 Gate의 위치에 따라 정하였다.

Gate가 Map의 경계 벽에 해당하는 경우, Map 안쪽 방향에 맞게 Snake가 출현하도록 하고 Snake 클래스의 setDir() 함수로 Snake의 진행 방향을 변경하였다. Snake가 나오는 Gate가 Map의 경계 안쪽 벽에 해당하는 경우는 "경계 안쪽에 생성된 여러 개의 벽에 Gate가 출현하는 경우"와 "경계 안쪽에 생성된 하나의 벽에 Gate가 출현하는 경우"로 구분하여 개발하였다.

"경계 안쪽에 생성된 여러 개의 벽에 Gate가 출현하는 경우"의 경우, 기존의 진행 방향으로 설정하면 벽에 곧바로 충돌하는 경우가 발생하기 때문에 이를 처리하기 위해 현재 Gate의 위치에서 상하좌우를 탐색하여 벽이 아닌 공간으로 Snake가 나오도록 하였다. "경계 안쪽에 생성된 하나의 벽에 Gate가 출현하는 경우"의 경우, 어느 방향으로 Snake가 나오더라도 벽에 곧바로 충돌하는 경우가 발생하지 않기 때문에 기존의 진행 방향을 유지한 채로 Snake가 나오도록 하였다. Gate의 통과 여부를 확인하고 그에 따라 Snake의 이동을 처리하기 위해 매 프레임마다 checkWallGate()

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

함수를 통해 Gate통과를 확인하고 이에 따라 moveSnake()함수와 passGate() 함수를 구분하여 실행하도록 처리하였다.

5. 5단계 개발 내용

Map 클래스에 멤버 함수로 scoreWindow()와 missionWindow() 함수를 만들어 윈도우를 만들고 그 안에 Manage 클래스의 멤버 함수로 setScore()와 setMssion() 함수로 mvprintw() 함수를 사용해 Score Board와 Mission Board를 구성하였다. Stage는 Manage 클래스의 멤버 함수인 Run() 함수에서 switch를 사용해 4단계로 구분하였고, level이 5가 되면 Map 클래스의 MissionComplete() 함수에서 newwin()을 사용해 새로운 윈도우 창을 만들어서 Mission Complete 문구가 출력되도록 하였다. 1단계부터 4단계까지 level이 증가할수록 setDelay() 함수의 인자 크기를 줄여서 1단계는 틱이 0.1초, 2단계는 0.07초, 3단계는 0.05초, 4단계는 0.03초로 구성하여 난이도를 나눴다. 또한 1, 2단계는 Map이 동일하지만 3단계부터는 가운데에 벽이 생기도록 했고, 4단계는 3단계에서 추가됐던 벽을 다른 방식으로 연장시키도록 했다.

Score에 필요한 변수들은 currentLen, growNum, poisonNum, useGate가 있는데 Mission에 사용되는 변수를 만들기 위해 Score 변수 이름 앞에 m을 붙여 총 8개의 멤버 변수를 사용하여 미션 성공 여부를 체크하였다. 게임이 중간에 종료되면 overWindow() 함수를 사용해 newwin()으로 새로운 윈도우 창을 생성하여 mvwprintw() 함수를 사용해 "Game Over" 문구를 출력하도록 했고, 그 밑 부분에 마찬가지로 mvprintw() 함수를 사용하여 currenLen과 growNum, poisonNum, useGate를 출력하도록 했고 마지막으로 time_t 변수를 사용해 실행 시작 시간과 끝 시간을 저장해 difftime() 메소드의 인자로 넣어서 총 Play time을 구해 출력하도록 하였다.

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.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

1. 1 단계 시스템 구조 및 설계

1.1 내용

Map 클래스에는 게임 맵 윈도우와 Score 윈도우, 게임 종료 윈도우, 미션 윈도우, 미션 완료 윈도우를 화면 상에 표현하기 위해 Ncurses 라이브러리를 이용하여 각각 WINDOW 포인터 타입의 변수로 선언하였다. 해당 변수들은 Map 클래스에 정의된 gameMap() 함수와 scoreWindow() 함수, missionWindow() 함수, overWindow() 함수 등에서 newwin() 함수를 통해 메모리를 동적할당 받아 정의하였다. 각 윈도우의 위치와 크기를 조정하기 위해 먼저 resize _term() 함수로 전체 터미널의 size를 35x80의 크기로 고정하였고 터미널 내부의 좌표 (row, col)를 이용하여 개발을 진행하였다.

게임 맵 윈도우에서 터미널 상 시작 위치를 (2, 2) 즉, 2행 2열의 좌표로 설정하여 Height는 30으로 Width는 50으로 고정하여 게임 맵의 크기를 구성하였다. 게임 맵의 경계는 wborder() 함수를 통해 Wall에 해당하는 영역은 문자 'X'로 Immune Wall에 해당하는 영역은 문자 '+' 로 하여 게임 맵의 UI를 구성하였다. 이와 동일한 방식으로 Score 윈도우는 (2, 55)를 시작 위치로 하고 Height는 10, Width 15로 하여 Score 윈도우의 UI를 구성하였다. 이와 동일한 방식으로 게임 종료 윈도우와 미션 윈도우, 미션 완료 윈도우를 정의하고 UI를 구성하였다.

추가로, 맵의 Width와 Height를 저장하는 map_loc 구조체를 선언하여 이후 단계 개발 목표인 Snake의 생성과 Item의 출현, Gate의 출현 프로그램 개발에 활용하였고 stage(level)가 진행됨에 따라 맵 경계 안에 Wall가 출현하도록 하는 makeWall() 함수를 정의하였다. 최종적으로 Map 클래스의 생성자에 각 함수들을 호출하여 게임 시작과 동시에 게임 맵과 Score 윈도우, 게임 종료 윈도우 등이 화면에 출력되도록 하였다.

1.2 개발을 위해 수정한 파일 및 클래스 / 함수

- 수정한 파일


map.h: Map 구성을 위해 필요한 라이브러리와 클래스 및 함수들로 선언되어 있다.

map.cpp: Map.h에서 선언한 함수들을 정의한다.

- 수정한 클래스/구조체

Map class: 멤버 변수로 WINDOW* 타입의 게임 맵, Score window, 게임 종료 윈도우 등 각 종 윈도우들을 UI로 구성하기 위해 필요한 변수들을 선언하고 해당 변수들을 정의하는 함수들을 선언하고 정의한다.

map_loc 구조체: 게임 맵의 Height와 Width 정보를 저장하기 위한 구조체

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

- 수정한 함수

Map() 생성자: 터미널 크기를 초기화 하고 각 윈도우마다 사용할 COLOR_PAIR를 정의한다. 게임 시작과 동시에 UI 구성을 위해 각 함수들을 호출한다.

~MAP() 소멸자: 윈도우 구성을 위해 동적할당 받은 변수들에 할당된 메모리를 반납하는 delwin() 함수를 호출한다.

gameMap(): 게임 맵 구성을 위한 함수

scoreWindow(): Score 윈도우 구성을 위한 함수

overWindow(): 게임 종료 윈도우 구성을 위한 함수

MissionComplete(): 미션 완료 윈도우 구성을 위한 함수


makeWall(int stage=3): stage가 진행될 때마다 게임 맵 안에 wall을 생성하는 함수

2. 2 단계 시스템 구조 및 설계

2.1 내용

Snake를 표현하기 위해서 문자 'O'로 Snake의 형태를 표현하였고 게임 맵 상에서 Snake의 좌표를 나타내는 구조체를 C++ STL 컨테이너인 vector에 저장하여 Snake를 구성하였다. 이를 위해 게임 맵 map.h 파일을 include하여 게임 맵의 정보를 이용할 수 있도록 하였다. 생성자에서 keypad() 함수를 통해 게임 시작 시 사용자의 입력을 받아 Snake가 이동할 수 있도록 하였고 nodelay() 함수를 true로 설정하여 사용자의 입력 없이도 Snake가 움직일 수 있도록 하였다. Snake의 이동은 moveSnake() 함수를 정의하여 사용자의 입력과 Snake의 현재 이동 방향에 따라 처리하였다.

사용자의 입력은 getch()함수를 통해 받고 이를 switch-case문을 활용하여 "KEY_LEFT"에 해당하는지 "KEY_UP"에 해당하는지 등 각 case를 구분하여 Snake 클래스의 멤버 dir 변수의 값을 변경하였다. 이때, 현재 진행 방향이 사용자의 입력과 반대인 경우 dir 변수의 값을 'Q'로 변경하여 게임 종료 조건에 해당하도록 하였다. 또한, status 변수를 멤버로 정의하여 Snake가 Item을 획득한 상태인지 아닌지 구분하도록 하였고 이에 따라 Snake의 형태와 이동을 조정할 수 있게 하였다. status가 0인 경우는 Item을 먹지 않은 상태로 일반적인 진행 상황을 의미하며 이때 Snake를 나타내는 vector의 원소를 pop_back하고 vector의 첫 번째 원소에 다음 이동 방향에 맞는 좌표를 insert하는 식으로 Snake를 구성하였다. 화면 상에서 Snake의 표현은 Ncurses 라이브러리의

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

move() 함수를 통해 Snake vector의 첫 번째 원소 즉 Snake의 Head에 해당하는 좌표로 이동하고 addch('O')를 통해 해당 좌표에 문자 'O'를 삽입하고 refresh()를 통해 화면에 출력 되도록 한다.

위의 개발한 기능과 함께 게임 운영을 위한 Manage 클래스를 정의하여 규칙1을 수행하도록 하였다. 해당 클래스에서 Snake.h파일을 include하여 Snake의 정보를 가져오고 이를 Manage 클래스의 멤버 함수 checkWallGate() 함수에서 Snake의 Head가 Wall과 Gate에 해당하는지 판단하도록 하였고 같은 클래스의 멤버 함수 checkBody() 함수를 통해 Snake의 Head가 Body와 Tail에 해당하는지 판단하도록 하여 규칙1을 준수하도록 하였다. 최종적으로 Manage 클래스의 멤버 Run() 함수를 정의하여 while 문과 Ncurses 라이브러리의 usleep() 함수를 통해 매 프레임마다 checkWallGate()함수와 checkBody() 함수를 실행하여 규칙 1을 준수하도록 하였다.

2.2 개발을 위해 수정한 파일 및 클래스 / 함수

- 수정한 파일

snake.h: Snake 구성을 위해 필요한 라이브러리와 클래스 및 함수들로 선언한다.

snake.cpp: Snake.h 파일에서 선언한 함수들을 정의한다.

manage.h: 게임 운영에 필요한 클래스 및 함수를 선언한다.

manage.cpp: manage.h 파일에서 선언한 함수들을 정의한다.

- 수정한 클래스/구조체

Snake class: 멤버 변수로 Snake를 구성하기 위한 vector를 선언하고 Snake의 모양을 나타내는 char 타입의 snake_shape 변수를 선언한다. Snake의 이동 방향을 나타내는 dir 변수와 상태를 나타내는 status 변수를 선언한다.

Snake의 이동을 처리하는 함수, 이동 방향을 설정하는 함수, Snake의 상태를 설정하는 함수 등을 선언하고 정의한다.


snake_loc 구조체: 게임 맵 상에서 Snake의 좌표(row, col) 정보를 저장하기 위한 구조체

Manage class: Snake 이동 시 벽(Wall)과 Gate의 충돌 Snake Body와 Tail의 충돌을 확인하는 함수를 멤버 함수로 선언한다.

- 수정한 함수

Snake() 생성자: nodelay()를 true로 설정하여 사용자의 입력 대기를 하지 않도록 설정한다. 게임시작과 동시에 Snake가 화면에 나타나도록 Snake vector의 원소들을 추가하고 move(), addch(), refresh() 함수를 통해 Snake를 화면에 출력한다.

~Snake() 소멸자: 생성자에서 nodelay()를 통해 사용자의 입력을 대기하지 않도록

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

설정한 것을 대기하는 상태로 설정한다.(인자 변경:true→ false)

moveSnake(): 사용자의 입력과 item 획득 여부에 따라 Snake의 이동을 조정하는 함수

getSnakePos(): snake_loc 구조체 타입으로 Snake의 Head 좌표를 return하는 함수

setDir(), getDir(): Snake의 이동 방향을 설정하고 return하는 함수

setStatus(): item 획득 여부에 따라 Snake의 status를 설정하는 함수

manage()생성자: usleep()함수의 인자로 사용되는 멤버 변수 delay를 100000으로 초기화하여 0.1초마다 프레임이 진행되도록 한다.

checkWallNGate(): Snake 이동 시 벽과 Gate에 충돌을 판단하는 함수

checkBody(): Snake 이동 시 Snake의 Body와 Tail에 충돌하는지 판단하는 함수

Run(): 매 프레임마다 Snake가 이동할 때마다 벽과 Gate, Body, Tail을 확인하는


함수를 호출하여 게임을 진행시키는 함수

3. 3 단계 시스템 구조 및 설계

3.1 내용

Growth Item은 문자 'G'로 Poison Item은 'P'로 형태를 나타내고 색상은 COLOR_PAIR를 이용하여 각각 초록색과 빨간색으로 설정하여 구분하였다. Manage 클래스에 Item의 생성과 획득을 처리하는 함수를 정의하였다. 이를 위해 Snake에 대한 정보와 게임 맵 내부에 Item 출현을 위해 게임 맵에 대한 정보가 필요했고 snake.h 파일(map.h 파일 포함)을 include하여 Snake 정보와 맵 정보를 이용하고 임의의 좌표에 item이 출현하고 item을 획득할 수 있도록 하였다. 해당 기능은 makeGrowItem() 함수와 makePoisonItem() 함수를 통해 규칙2에 맞게 Growth Item과 Poison Item이 출현 되도록 하였다. 임의의 좌표에 Item을 출현하도록 하기 위해 ctime라이브러리와 cstdlib 라이브러리의 rand() 함수를 이용하였다. Item의 출현은 게임을 시작하자마자 생성될 수 있게 Manage 클래스의 생성자에 해당 함수를 호출하도록 하였고 ctime 라이브러리를 이용하여 이후에도 5초마다 item이 다른 임의의 위치에 다시 생성되도록 하였다.

다음으로, 3단계 개발 목표의 3번째 목표를 수행하기 위해 takeItem() 함수를 정의하여 Snake의 Item 획득 여부를 판단하였다. Snake가 Growth Item을 획득한 경우 Snake 클래스에 정의된 setStatus() 함수를 호출하여 Snake의 status값을 1로 변경하여 Snake vector의 마지막 원소를 pop하지 않도록 처리하여 Snake의 길이를 1증가시켰고 Poison Item을 획득한 경우 위와 동일한 방식으로 status를 2로 변경하여 item을 획득하지 않은 상태와 동일하게 pop을 진행하되 vector의 첫

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

번째 원소에 다음 위치에 대한 좌표를 insert하지 않는 식으로 처리하여 Snake의 길이를 1 감소시켰다. 게임 규칙2의 Snake의 길이가 3보다 작아지는 경우는 같은 클래스의 멤버 Run() 함수에서 매 프레임마다Snake의 size를 판단하도록 처리하였다. 마지막으로 새로운 위치에 Item이 곧바로 출현되도록 makeGrowItem() 함수와 makePoisonItem() 함수가 실행되도록 하였다.

3.2 개발을 위해 수정한 파일 및 클래스 / 함수

- 수정한 파일

manage.h: Item 출현과 획득을 위한 클래스 및 함수들로 선언하였다.

manage.cpp: manage.h 파일에서 선언한 함수들을 정의한다.

- 수정한 클래스/구조체

Manage class: Growth Item과 Poison Item을 생성하는 함수들을 선언하고 각 Item을 Snake가 획득했는지 판단하는 함수를 선언한다.

- 수정한 함수

manage() 생성자: 벽 판단 멤버 변수 ckWall와 Gate 판단 멤버 변수 ckGate, char 타입의 변수 growth_item변수와 poison_item 변수를 각각 초기화하고 게임 시작과 동시에 Item이 출현하도록 makeGrowItem()함수와 makePoisonItem() 함수를 실행한다.

makeGrowItem(): Growth item을 게임 맵 내부 임의의 위치에 생성한다.

makePoisonItem(): Poison item을 게임 맵 내부 임의의 위치에 생성한다.


takeItem(): Snake의 Item 획득 여부를 Item의 좌표를 통해 판단하고 Snake의 status를 변경한다.

Run(): 매 프레임마다 일정 시간 간격으로 makeGrowItem함수와 makePoison함수를 실행하여 Item이 새로운 위치에 출현하도록 하고 takeItem() 함수를 실행하여 Snake의 Item 획득 여부를 판단한다.

4. 4 단계 시스템 구조 및 설계

4.1 내용

Manage 클래스에 Gate 생성과 Snake의 Gate 통과를 처리하는 함수를 정의하였다. Gate의 출현은 makeGate() 함수를 통해 구현하였고 출현 방식은 규칙 3과 5를 준수하면서 Item 출현과동일한 방식으로 조건을 달리하여 Immune Wall을 제외한 게임 맵의 경계에 해당하는 Wall와 경계안


 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

에 해당하는 Wall에 출현하도록 하였다. Gate는 Wall과 같은 문자 'X'를 사용하여 표현하였고 구분을 위해 COLOR_PAIR를 이용하여 색상을 노란색으로 설정하였다. Gate의 출현은 게임 시작 후 일정 시간 후에 생성되도록 gate_time 변수를 이용하여 처리하였고 이후 Run() 함수에서 stage(level)가 진행될 때마다 새로운 임의의 위치에 Gate가 다시 출현하도록 설정하였다. Snake가 Gate를 통과하도록 처리하는 passGate() 함수를 정의하였다. 해당 함수는 매번 호출되지만 Snake의 Head 좌표와 Gate의 좌표의 일치 여부를 판단하는 조건문에 따라 함수 실행 여부를 결정한다.

함수가 실행되면 Snake 클래스의 moveSnake() 함수와 동일한 방식으로 진행하되 어떤 Gate로 들어왔는지에 따라 Snake가 나오는 Gate의 구분이 필요했기 때문에 각 Gate의 좌표를 이용하였다. 이후, checkOutputGate() 함수를 통해 다른 Gate에서 Snake가 출현하도록 처리하였다. 이때, Snake의 출현 위치와 진행 방향은 규칙 4를 준수하여 Gate의 위치에 따라 정하였다. Gate가 Map의 경계 벽에 해당하는 경우, Map 안쪽 방향에 맞게 Snake가 출현하도록 Snake의 vector에 Map 안쪽 방향의 좌표를 insert하고 Snake 클래스의 setDir() 함수로 Snake의 진행 방향을 Map의 안쪽 방향으로 변경하였다. 이후, moveSnake() 함수에서와 같은 방식으로 move()와 addch(), refresh()를 이용해 화면 상에 Snake의 형태가 출력 되도록 하였다. Snake가 나오는 Gate가 Map의 경계 안쪽에 있는 벽에 해당하는 경우는 다음과 같이 구분하여 개발하였다.

1. 경계 안쪽에 생성된 여러 개의 벽에 Gate가 출현하는 경우
2. 경계 안쪽에 생성된 하나의 벽에 Gate가 출현하는 경우

1번의 경우, 기존의 진행 방향으로 설정하면 벽에 곧바로 충돌하는 경우가 발생하기 때문에 이를 처리하기 위해 현재 Gate의 위치에서 상하좌우를 탐색하여 벽이 아닌 공간으로 Snake가 나오도록 하였다. 상하좌우 탐색을 위해 int형의 배열 dx와 dy를 선언하여 for문을 통해 최대 4번의 탐색을 진행하고 진행 중 빈 공간이 발견되면 해당 좌표와 방향 정보를 이용해 Snake의 출현을 결정하였다. 2번의 경우, 어느 방향으로 Snake가 나오더라도 벽에 곧바로 충돌하는 경우가 발생하지 않기 때문에 위와 달리 상하좌우 탐색없이 곧 바로 기존의 진행 방향을 유지한 채로 Snake가 나오도록 하였다. Gate의 통과 여부를 확인하고 그에 따라 Snake의 이동을 처리하기 위해 Manage 클래스에 정의된 멤버함수 Run()에서 매 프레임마다 checkWallNGate() 함수를 통해 Gate 통과를 확인하고 확인 결과에 따라 moveSnake() 함수와 passGate() 함수를 구분하여 Snake의 이동이 이루어지도록 처리하였다.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

4.2 개발을 위해 수정한 파일 및 클래스 / 함수

- 수정한 파일

manage.h: Gate 출현과 통과를 위해 Manage 클래스를 수정하였다.

manage.cpp: manage.h 파일에서 수정한 부분을 구현하였다.

- 수정한 클래스/구조체

Manage class: Gate의 출현과 통과를 위해 필요한 변수와 함수를 선언하고 정의하였다.

- 수정한 함수

makeGate(): 게임 맵의 경계 벽과 맵 안쪽 벽에 Gate를 생성하는 함수

passGate(): Snake가 Gate를 지나갈 때 Snake의 이동을 처리하는 함수

checkOutputGate(): Snake가 Gate를 지나갈 때 나오는 Gate의 위치에 따라 Snake의 이동을 처리하는 함수

resetGate(): Level이 올라갈 때마다 기존의 Gate는 벽으로 만들고 새로운 위치에 Gate가 생기도록 설정한다.

Run(): 게임 시작 후 int형 변수 gate_time의 값을 증가시켜 일정 시간


(gate_time == 10)이 지나면 makeGate() 함수를 실행하여 Gate를 생성하고 switch-case문을 통해 level마다 새로운 위치에 Gate가 생성되도록 한다.

5. 5단계 시스템 구조 및 설계

5.1 내용

Score Board와 Mission Board를 각각 다른 WINDOW 포인터로 만들어서 화면에 구성했다. WINDOW 포인터를 Map 클래스의 멤버 변수로 저장하고 멤버 함수인 scoreWindow() 함수와 missionWindow() 함수에 newwin() 메소드와 wbkgd(), watttrn, mvprintw(), wrefresh() 메소드까지 사용하여 구성하였다. 구성된 윈도우 안에는 Manage 클래스의 멤버 함수인 setScore(), setMission() 함수를 통해 setScore()는 현재 자신의 점수를, setMission()에서는 달성해야 할 목표를 추가한다. 각 레벨별로 미션이 달라지도록 하기 위해 Score Board의 각 요소들이 Mission Board의 요소들과 같아지면 Mission Board에서 해당 요소에 대해 "V" 표시가 된다. 모든 요소에 대해서 "V" 표시가 이루어지면 다음 level로 넘어가도록 하였다. Level이 높아지면 높아질수록 Manage 클래스의 Run() 함수에서 게임의 틱을 줄이도록 설정하여 level 1은 0.1초, level 2는 0.07초, level 3는 0.05초, level 4는 0.03초의 틱을 가지므로 점점 Snake의 움직임이 빨라진다. 또한 level 3부터는 Map의 변화가 생기고 변화된 Map에 대해서도 Gate를 생성하도록 하여 더욱 어려워지도록 하였다.

Snake Game이 종료되면 현재 길이, 획득한 Grow Item의 수, 획득한 Poison Item의 수, 게이트를 사용한 횟수, 총 플레이 시간이 출력되며 게임 도중에 끝나면 "Game Over"를, 4단계 미션까지

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

클리어 했다면 "Mission Complete"가 출력된다. 여기서 Play time은 C++ 표준 라이브러리의 ctime 클래스를 include하여 time_t 자료형을 사용해 게임 시작시간과 끝 시간을 변수로 저장하여 끝 시간 - 시작 시간을 int형으로 형변환하여 사용하였다.

5.2 개발을 위해 수정한 파일 및 클래스 / 함수

- 수정한 파일

map.h: Score Board와 Mission Board를 구성하기 위해 수정하였다.

map.cpp: map.h에서 수정된 함수들을 정의하였다.

manage.h: Score Board와 Mission Board의 내용을 구성하기 위해 수정하였다.

manage.cpp: manage.h에서 수정된 함수들을 정의하였다.

- 수정한 클래스/구조체

Map class: WINDOW 포인터인 swin, mwin을 멤버 변수로 추가하였고 Mission과 Level을 체크하는데 사용되는 멤버 함수를 추가하였다.

Manage class: Score Board와 Mission Board를 구성하는 멤버 변수 및 멤버 함수를 추가하였다.

- 수정한 함수

setScore(): Score Board의 내용을 추가하는 함수

setMission(): Mission Board의 내용을 추가하는 함수

checkLevel(): 현재 진행도인 Level을 체크하여 각각 다른 난이도를 주도록 하는 함수

setDelay(): 난이도를 조정하는 방법 중 하나인 게임의 속도를 조절하는 함수

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.

1. Ncurses 라이브러리

터미널 상에서 게임화면 UI를 구성하는 라이브러리로 move() 함수를 적절히 활용하여 각 단계 별 프로그램을 수월하게 개발할 수 있었다.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

2. C++ STL Vector container

Snake 를 구성하는데 사용한 라이브러리로 Vector 컨테이너의 내장 함수인 insert() 함수와 pop_back()함수를 활용해 Snake 구성을 수월하게 할 수 있었다.

3. cstdlib 라이브러리

Item 이나 Gate 의 임의의 위치에 출현시키기 위해 rand() 함수를 활용하였다.

4. ctime 라이브러리

ctsdlib 라이브러리의 rand() 함수와 혼용하여 사용함으로써 랜덤 추출 시마다 새로운 값을 얻을 수 있었고 이를 통해 다양한 위치에서 Item 과 Gate 의 출현이 가능했다. Item 의 출현 시간을 일정 시간 간격으로 출현하도록 조절하기 위해 활용하였다.

5. cmath 라이브러리

점수 계산을 위해 사용하였다.

6. 고안한 알고리즘

Snake 의 구성을 위해 게임 맵의 좌표를 이용하였고 좌표 정보를 한 번에 처리할 수 있도록 구조체를 선언하여 관리하였다. 더불어 vector 컨테이너를 활용하여 Snake 의 형태와 이동을 수월하게 관리할 수 있었다.

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.

1. Ncurses 라이브러리는 Linux 환경에서 활용할 수 있다는 점에서 다른 OS 에서 활용 시 Virtual Box 와 같은 가상 환경에서 Linux 를 설치하여 이용해야 한다는 점에서 게임 사용자 입장에서 환경 설정에 어려움이 있을 것으로 생각된다.
2. Ncurses 라이브러리는 화면 UI 를 구성하는 라이브러리이다 보니 UI 구성 이외의 기능을 구현할 때 UI 로 인해 디버깅이 어려웠다. 이를 위해 기능 구현 시 완벽하게 프로그램을 설계하려 하였고 그 결과 디버깅 상황을 최소화하였다.

2.2.5 결과물 목록

작성요령 (5 점)

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

 국민대학교 컴퓨터공학부 C++ 기말 프로젝 트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

1. map.h 파일

Map 구성을 위해 필요한 클래스 및 함수, 구조체가 선언되어 있다.

2. map.cpp 파일

map.h 파일에서 선언한 변수와 함수들이 각각 정의되어 있다.

3. snake.h 파일

Snake 를 구성하고 이를 화면에 표현하기 위한 클래스 및 함수, 구조체가 선언되어 있다.

4. snake.cpp 파일

snake.h 파일에서 선언한 변수와 함수들이 각각 정의되어 있다.

5. manage.h 파일

게임 규칙 등 게임의 전반적인 운영을 위한 클래스 및 함수들이 선언되어 있다.

6. manage.cpp 파일


manage.h 파일에서 선언한 함수들이 각각 정의되어 있다.

7. main.cpp 파일

manage.h 파일을 include 하여 Manage 클래스의 멤버 Run() 함수를 실행함으로써 게임 실행을 담당한다.

8. Makefile

각 cpp 파일을 컴파일하기 위한 명령어가 입력되어 있다.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

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.

유선중 : 전체적인 프로젝트 구조 및 설계를 진행하여 ncurses 를 사용한 Snake Game 을 구현했다. 처음에는 ncurses 에 대한 개념이 부족해서 방향을 잡기 힘들었지만 사용법을 공부하면서 수월하게 프로젝트를 진행할 수 있었다. 전체적인 계획을 세워서 체계적으로 프로젝트를 진행했으면 더 좋은 결과물을 만들 수 있었을 것이라는 생각이 들어서 아쉬움이 남긴 하지만 만족스러운 결과물을 만들었다고 생각한다.

양성민 : Snake Game에 필요한 몇 가지 기능을 개발했다. 기능에 필요한 ncurses Library 가 익숙하지 않아 어려움을 느꼈지만 프로젝트를 진행할수록 익숙해져서 이후에는 편하게 개발을 진행했다. 프로젝트 설명만으로는 부족한 부분이 있어서 각 단계별로 어떤 식으로 개발할지 생각을 했어야 했는데 그 부분을 생각하지 않고 바로 개발을 진행해서 수정하는 상황이 많았다. 하지만 결과물을 만들어내서 좋은 경험을 했다고 생각한다.

4 참고 문헌

번호	종류	제목	출처	발행년도	저자	기타
1	PDF	C++ 강의 Ncurses 라이브러리 자료	국민대		임은진	
2	블로그	Ncurses 프로그래밍 - 투명링크	https://anythink.tistory.com/entry/Linux-NCURSES-%ED%94%84%EB%A1%9C%EA%B7%B8%EB%9E%98%EB%B0%8D	2013 년	블로거 “투명링크”	
3	블로그	Ncurses 라이브러리 관련 링크	https://neverapple88.tistory.com/28	2016sus	블로거 “neverapple88”	

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

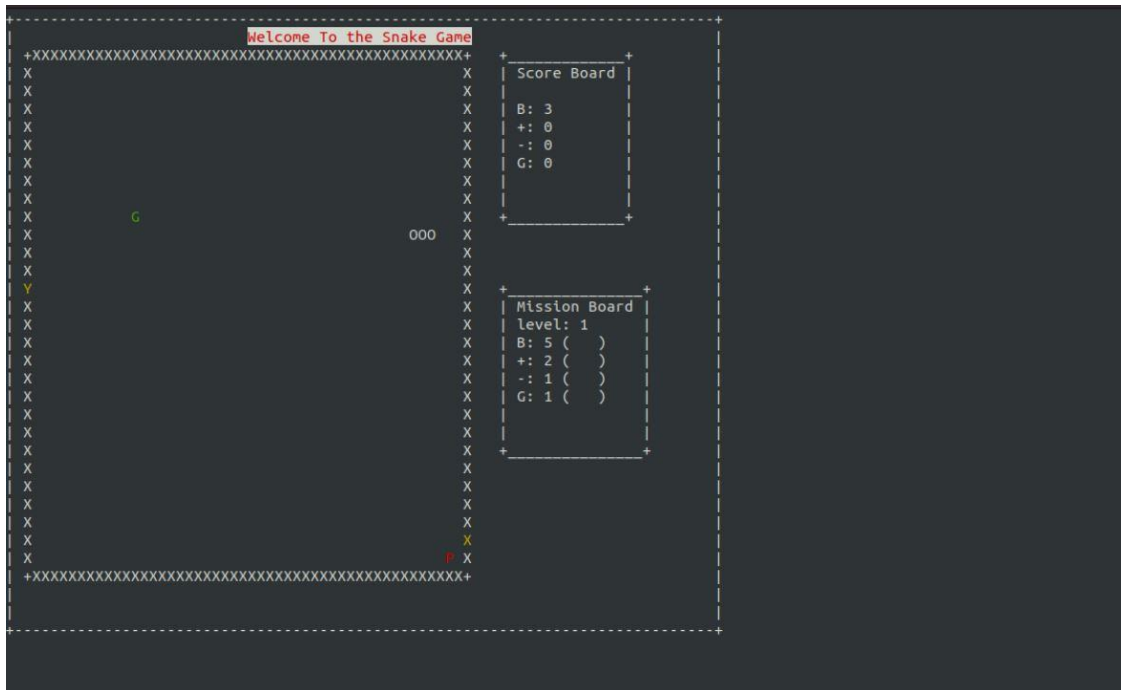
5 부록

작성요령 (15 점)

프로젝트의 결과물을 사용하기 위한 방법에 대해서 작성하세요.


5.1 사용자 매뉴얼

1. 게임 시작 화면

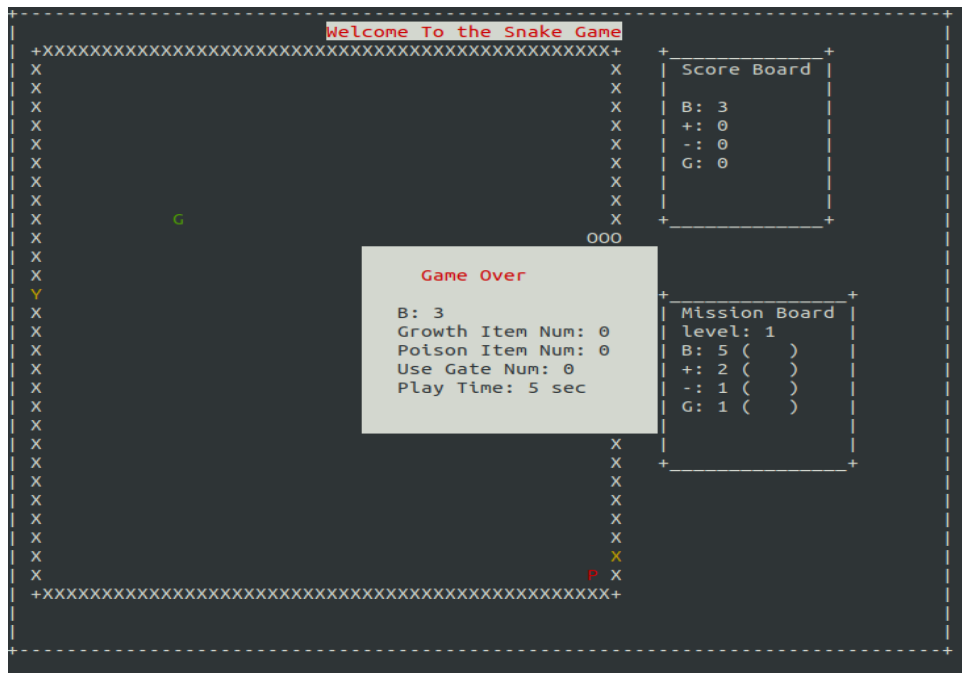


[그림 1]

게임을 실행하면 위와 같이 "OOO" 형태로 Snake 의 형태가 나오고 키보드의 방향키 입력을 통해 Snake 의 움직임을 조종할 수 있다. 처음 시작 시 level 은 1로 설정되어 있고 우측 하단에 Mission Board 에 나와 있는 대로 점수를 획득하면 level 이 1씩 증가하게 된다. level 이 증가할 때마다 게임의 속도가 빨라지고 맵 중앙에 벽이 생기기도 하며 level 4까지 완성되어 있다.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

2. 게임 종료 화면



[그림 2]

Snake Game 이 종료되는 상황은 다음과 같다.

1. Snake 의 머리가 벽에 부딪힐 때 => Game Over
2. 현재 진행 방향의 반대 방향으로 조작했을 때 => Game Over
3. Snake 의 머리가 몸통에 부딪힐 때 => Game Over
4. Snake 의 길이가 3 보다 작을 때 => Game Over
5. 모든 미션을 Clear 했을 때 => Mission Complete


5.2 설치 방법

1. <https://github.com/SeonJongYoo/SnakeGameProject-Cpp> URL 에 접속하여 코드를 다운 받는다.
2. 다운 받은 코드가 있는 디렉토리에서 터미널 창에 접속한다.
3. 터미널 환경에서

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

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

명령을 입력하여 ncurses 설치를 진행한다.

 국민대학교 컴퓨터공학부 C++ 기말 프로젝 트	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	유선중, 양성민	
	Confidential Restricted	Version 1.2	2021-JUN-05

4. Make 명령을 통해 기존 파일의 Makefile 을 사용하여 컴파일을 진행한다.
5. ./main.out 명령어로 Snake Game 을 시작한다.

5.3 데모 동영상 링크

1. 미션 성공 영상 링크
<https://youtu.be/lo2tqKEnZc0>
2. 미션 실패 영상 링크 – Snake 진행 방향의 반대 방향의 키를 입력한 경우
<https://youtu.be/DUTPah4Axtw>