C++프로그래밍
프로젝트

프로젝트 명	Snake game
문서 제목	결과 보고서

팀원	박정재 (20150359)
	최유찬 (20192000)

목차

1 개발 내역

수정 날짜	대표수정자	Revision	추가/수정 항목	내 용
2020/5/28	박정재	1.0	내용 추가	main 함수 구현
2020/5/31	최유찬	1.1	내용 추가	map 배열 구성, 맵과 요소들을 Game 화면에 나타내는 show 함수 구현
2020/6/2	박정재	1.3	내용 추가	키 입력과 이동을 구현하는 input 함수와 move 함수 구현
2020/6/3	박정재	1.4	내용 추가, 수정	move 함수 수정, Game Rule #1 실패 조건 추가
2020/6/3	최유찬	1.5	내용 수정	show함수에서 화면이 표시되지 않는 문제 해결
2020/6/8	최유찬	1.6	내용 추가	Growth Item과 Poison Item 만드는 함수 구현
2020/6/13	최유찬	1.7	내용 추가	Gate 만드는 함수 구현
2020/6/15	박정재	1.8	내용 추가	Item 획득 효과 구현(몸 길이 변화), Gate에 Snake가 통과하는 기능 구현
2020/6/16	박정재	1.9	내용 추가+수정	게임 화면 색 입히기, 아이템 함수 호출 문제 수정, 아이템 일정 시간 이후 소멸 기능 추가
2020/6/16	최유찬	2.0	내용 추가	Score / Mission Board 생성 및 Item 획득과 Gate 통과 횟수 체크하는 변수 설정
2020/6/16	박정재	2.1	내용 추가	Stage 2, 3, 4 추가, 각 미션 목표 랜덤으로 할당되도록 구현
2020/6/16	박정재	2.2	내용 추가	각 스테이지에서 미션을 모두 수행하였을 때 실행되는 함수 구현

Github 주소: https://github.com/3ltigr0/NcursesSnake

2 개발 내용 및 결과물

2.1 목표

1단계: Ncurses Library 함수들을 사용하여 2차원 배열로 된 Snake Map을 Game 화면으로 표시하는 프로그램을 완성한다.

2단계: 1단계 맵 위에 Snake를 표시하고, 화살표를 입력받아 Snake가 움직이도록 프로그램을 완성한다. Snake는 규칙 #1을 준수한다.

3단계: 2단계 프로그램에서, Map위에 Growth Item과 Poison Item을 출현하도록 수정한다. 게임규칙 #2를 준수한다.

4단계: 3단계 프로그램에서, Map의 Wall의 임의의 위치에 한 쌍의 Gate가 출현할 수 있도록 변경하고, 각 Gate에 Snake가 통과할 수 있도록 수정한다. 게임 규칙 #3, #4, #5를 준수한다.

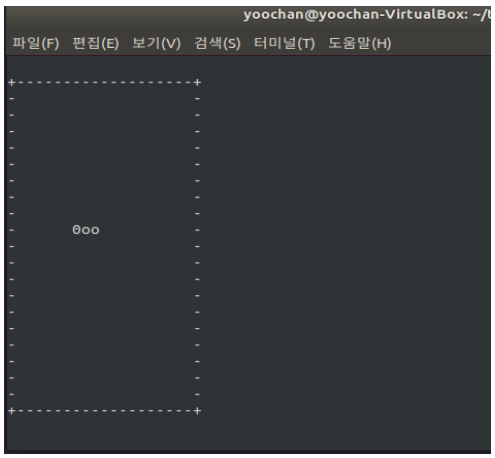
5단계: 4단계 프로그램에서, 우측에 게임 점수를 표시하는 화면을 구성한다. 게임 점수는 게임 규칙 #6을 준수한다. Mission 을 달성하면 다음 Map 으로 진행하도록 프로그램을 완성한다.

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

2.2 개발 내용 및 결과물

2.2.1 개발 내용

- 모든 개발 내용은 2020 C++ 프로젝트.pdf에 명시된 단계에 맞춰 이루어졌으며, 규칙도 pdf의 내용을 준수하였다.



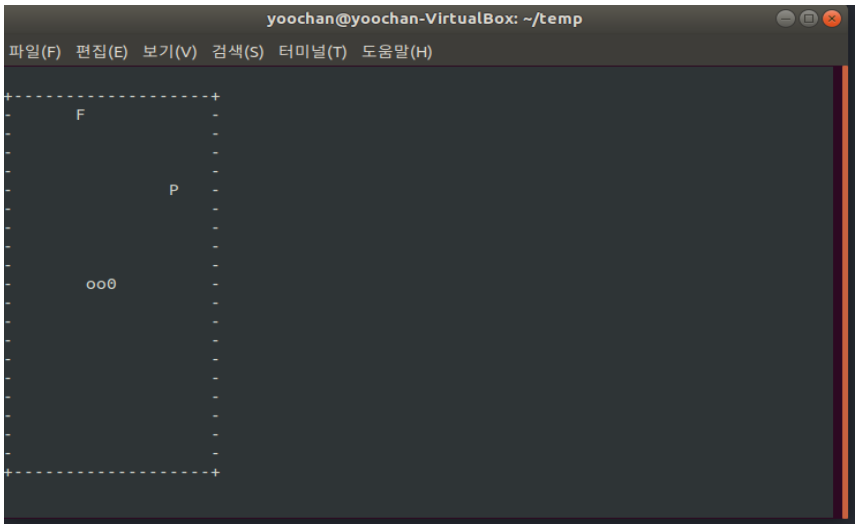
<사진 1. 1단계, 2단계 구현>

1단계: Map의 구현

- 사진 1에서와 같이 뱀이 이동할 수 있는 공간을 공백으로, Immune wall을 "+" 문자로, Wall을 "-" 문자로 표현하여 Map을 게임 화면에 표시하였다.

2단계: Snake 표현 및 조작

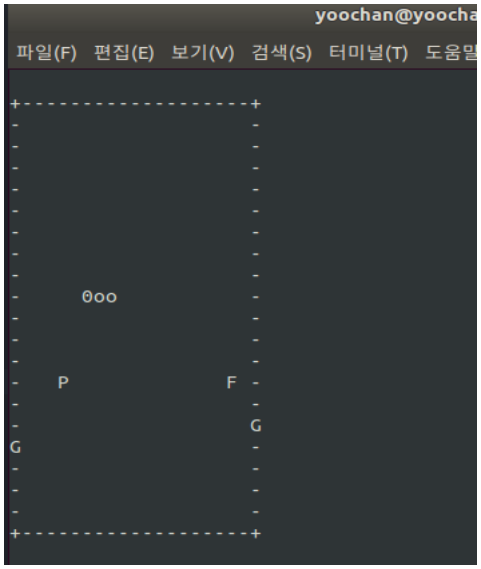
- 사진 1에서와 같이 Snake Head를 "0"으로, Snake body를 "o"로 표시하였고, 이 Snake를 방향키 조작(↑↓←→)을 통해 방향을 입력 받고 0.5초마다 움직이도록 구현하였다. 프로젝트 Rule에 명시된 바와 같이 진행 방향과 반대 방향키를 누르거나 Wall 또는 Snake Body에 부딪히면 게임이 실패하도록 구현하였다.



<사진 2. 3단계 구현>

3단계: Item 요소의 구현

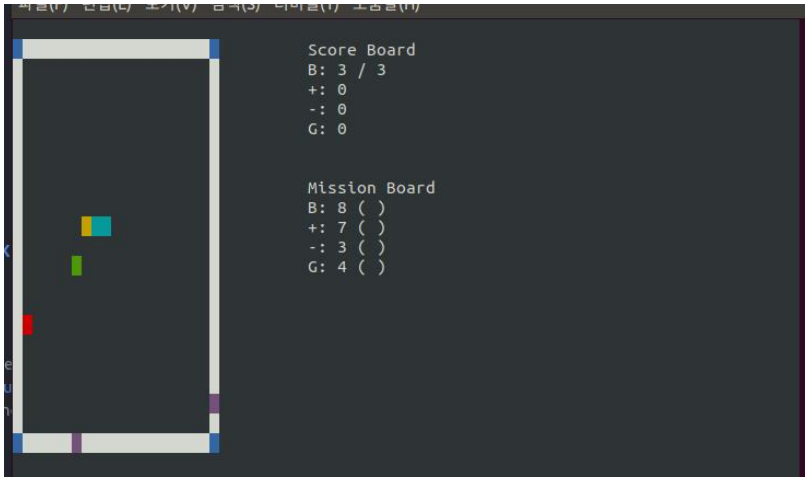
- 사진 2에서와 같이 Map에 Growth Item을 "F"로, Poison Item을 "P"로 표현하여 구현하였다. 이 아이템들은 임의의 위치에 랜덤으로 생성되며, 게임 시작 직후 최초 생성되어 일정 시간(10초)이 지나거나 아이템 획득 시 다른 위치에 다시 생성되도록 하였다. Growth Item을 획득하면 몸의 길이가 1만큼 증가하고, Poison Item을 획득하면 몸의 길이가 1만큼 감소한다.



<사진 3. 4단계 구현>

4단계: Gate 요소의 구현

- 사진 3에서와 같이 Wall에 Gate가 나타나도록 하였다. 이 Gate는 임의의 위치에 랜덤으로 생성되도록 하였으며, 게임 시작 직후 최초 생성된다. 한 게이트를 통과하면 다른 게이트로 Snake가 나오도록 하였고, Snake가 Gate를 완전히 통과하면 Gate가 다른 위치에 다시 생성된다. Gate 진출 방향은 Game Rule #4를 준수하였다.



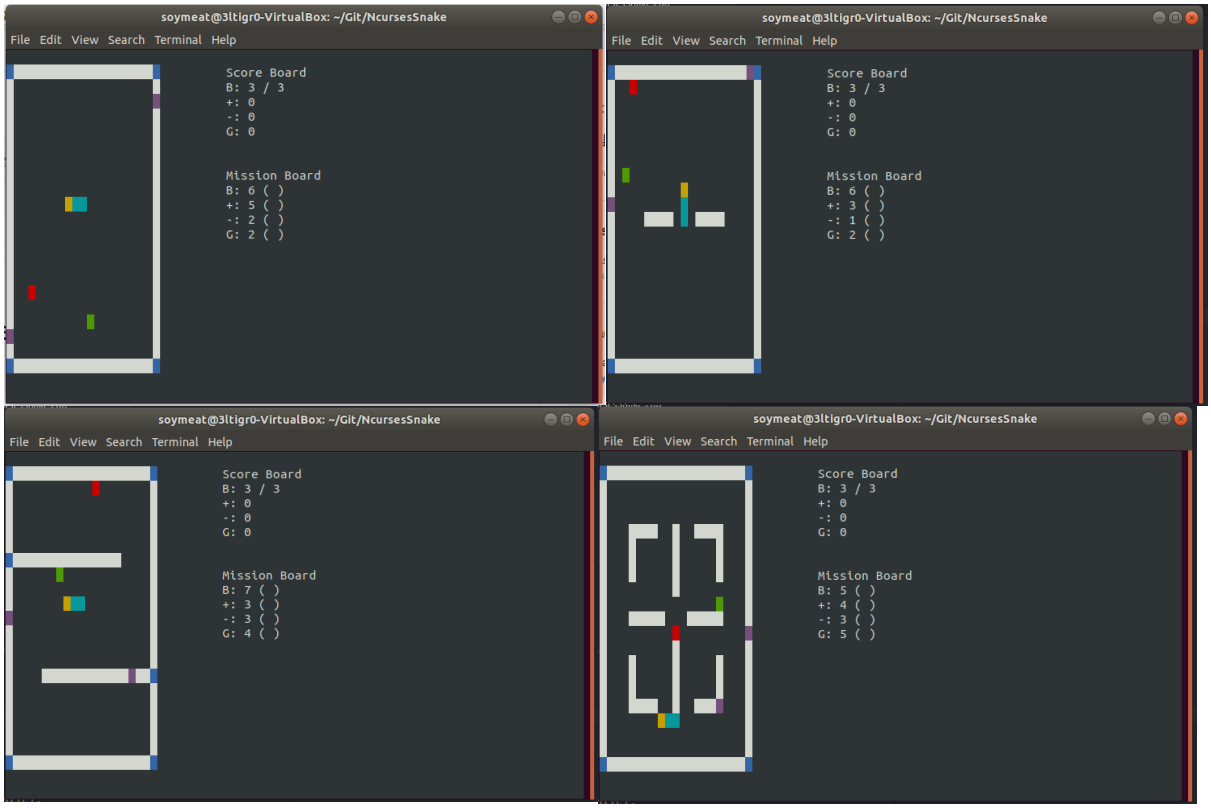
<사진 4. 5단계 구현>

5단계: 점수 요소의 구현

- 게임 화면의 각 요소들이 색으로 표현되도록 수정하였다. 흰색은 Wall, 파란색은 Immune Wall, 빨간색은 Poison Item, 초록색은 Growth Item, 보라색은 Gate, 노란색과 청록색은 각각 Snake Head와 Snake Body이다.

- 사진 4에서와 같이 Score Board와 Mission Board를 보여주도록 하였다. B는 현재 Snake의 길이 / 가장 길었던 Snake의 길이이다. +는 Growth Item의 획득 개수, -는 Poison Item의 획득 개수, G는 Gate 통과 횟수이다. Mission Board의 목표들은 랜덤으로 배정되어 점수가 미션 목표에 도달시

V 표시를 하여 알려준다.



<사진 5. Stage 1 – 4>

- 4개의 목표를 모두 도달하면 다음 Stage로 이동한다. Stage 별 Map의 형태는 사진 5와 같다. (왼쪽 위 Stage 1, 오른쪽 위 Stage 2, 왼쪽 아래 Stage 3, 오른쪽 아래 Stage 4)

2.2.2 시스템 구조 및 설계도

```
135  int main()
136  {
137      reset();
138      while (!fail)
139      {
140          timer();
141          input();
142          show();
143          usleep(tick);
144      }
145      if(clearall == false)
146      {
147          clear();
148          mvprintw(winy/2, (winx-11)/2, "You Failed!");
149          refresh();
150          usleep(2000000);
151      }
152      endwin(); // fail 시 종료
153      return 0;
154  }
```

<사진 6. 게임을 실행하는 반복문을 포함한 main 함수>

(1 단계) Ncurses Library 함수들을 사용하여 2 차원 배열로 된 Snake Map 을 Game 화면으로 표시하는 프로그램을 완성한다.

Ncurses Library를 사용하여 게임을 제작하므로 reset 함수를 만들어 Ncurses Library를 initialize 하는 함수들을 호출하고 main 실행 시 reset 함수를 호출하여 Ncurses Library를 initialize하도록 해주었다. reset 함수에는 그 외에도 게임 시작 시 필요한 각종 변수들을 초기화하는 기능을 담고 있다.

Stage 1 을 21*21 크기의 2 차원 배열을 사용하여 표현하였다. Wall 의 값은 1, Immune Wall 의 값은 2 이고 Snake 가 이동할 수 있는 공간은 0 값으로 할당하였다.

Map 의 내용을 게임 화면에 출력하여주는 Show 함수를 만들었다. Map 이 2 차원 배열의 형태로 표현되므로 이중 for 문을 이용하여 각 좌표의 값을 불러와 그 값에 따라 표현하는 방식을 사용하였다. 기존의 C++에서 cout 과 같은 역할을 하는 NCurses Library 의 함수인 printw 를 이용하였고, Immue wall 은 +, wall 은 -로 뱀이 움직일 수 있는 공간을 공백으로 표현하였다.

(2 단계) 1 단계 맵 위에 Snake 를 표시하고, 화살표를 입력받아 Snake 가 움직이도록 프로그램을 완성한다.

각 Stage 시작 시 필요한 변수와 함수를 초기화, 호출하는 함수인 setstage 함수를 만들어 Snake 의 최초 위치 좌표를 vector 에 넣고 dir 변수에 방향 값을 설정한다. vector 를 사용한 이유는 Item 획득을 통해 Body 의 길이가 길어지거나 짧아지는 경우 push 나 pop 을 통해 값을

추가하거나 지우는 것이 용이하기 때문이다. 그리고 사용자로부터 키 입력을 받는 input 함수를 만들어 Ncurses Library 의 getch 함수를 이용하여 방향키 입력을 감지하고 키 입력에 따라 방향을 바꾼다. 방항에 따라 Snake 를 이동시키는 move 함수를 만들어 input 함수에서 설정된 방향에 따라 vector 값의 좌표를 변경하고 map 배열에 Snake Head 를 3 으로, Snake Body 를 4 로 표시한다. 이 때 Snake 의 이동은 Snake 의 꼬리를 지우고 Snake 좌표를 앞으로 한 칸씩 당긴 다음 이동 방향에 Head 를 추가하는 방식으로 구현하였다. 0.5 초마다 move 함수를 호출하는 timer 함수를 선언하고 시간을 측정하기 위해 movetimer 변수를 선언하였다. main 의 반복문이 한번의 반복 후 0.1 초동안 sleep 하기 때문에 0.1 초마다 movetimer 에 1 씩 더해주어 movetimer 가 5 가 되면 0.5 초가 지났다고 판단하여 move 함수를 호출하도록 함수를 정의했다. (실제로는 반복문의 실행에 걸린 시간 이후에 0.1 초를 sleep 하기 때문에 0.1 초보다 약간 더 걸린다.) sleep 은 unistd.h 헤더의 usleep 함수를 이용하여 구현하였다. usleep(n)은 n 마이크로 초만큼 sleep 하는 함수이다.

1 단계에서 만들었던 show 함수에 뱀의 몸과 머리를 표현하는 부분을 추가하고 Snake 의 Head 는 0, Body 는 O 로 표현하였다.

(3 단계) 2 단계 프로그램에서, Map 위에 Growth item 과 Poison item 을 출현하도록 수정한다.

각 Item 들의 좌표를 저장하는 변수 g_itemx, g_itemy, p_itemx, p_itemy 를 선언하고, Item 을 생성해 좌표를 할당하는 함수인 growth_item 과 poison_item 을 만든다. 아이템이 랜덤으로 생성되어야 하므로 난수를 생성하기 위해 <time.h>를 include 한다. Item 들은 map 배열에서 벽이 아닌 0 에 해당하는 부분에 생성되어야 하므로 map 에서 Wall 과 Immue Wall 인 0 과 21 을 제외한 1~19 에 해당하는 값이 나오도록 rand 함수를 사용하여 난수를 발생시켜 앞서 선언한 변수에 저장한다. 0 이나 21 의 벽을 제외한 map 내부에서도 다른 아이템이나 Snake, 또는 벽이 존재할 수 있기 때문에 배열 값이 0 이 아닌 경우를 체크하기 위해 crush_on 이라는 변수를 선언하여 0 이 아닌 부분과 만나면 1 이 되도록 한다. 만약 crush_on = 1 이면 다시 while 문을 돌게 하도록 설계하고, 만약 정상적으로 0 의 값에 해당하는 배열에 난수가 생성되면 map 배열 값을 Item 에 해당하는 값으로 바꾸어 표시한다.

각 item 들은 일정 시간이 지나면 사라지고 다른 곳에 생성되도록 해야 하므로 2 단계에서 만든 timer 함수를 이용한다. 일정 시간에 도달하면 원래 Item 이 있던 배열의 값을 0 으로 바꾸고 Item 생성 함수를 다시 호출한다.

일정 시간에 도달하기 전에 Item 을 획득할 때의 효과를 구현하기 위해 move 함수에 이동 후 아이템을 획득하였는지 판단하는 기능을 추가한다. Growth Item 을 획득하는 경우 이동 이전의 꼬리 좌표를 vector 에 추가하여 준다. Poison Item 을 획득하는 경우 이동 후의 꼬리 좌표 vector 를 pop 하고 이후의 vector 값이 3 보다 작아지면 게임을 fail 시킨다. Item 과 Snake 가 겹치지 않도록 하기 위해 Snake 를 map 배열에 먼저 표시한 후 Item 생성 함수를 호출하고, 각 timer 값을 처음 값으로 할당한다.

Item 을 만드는 함수에서 Growth Item 은 배열에서 5 의 값으로, Poison Item 은 6 의 값으로 표시하였고 show 함수에도 이에 대한 사항을 추가하였다.

(4 단계) 3 단계 프로그램에서, Map 의 Wall 의 임의의 위치에 한 쌍의 Gate 가 출현할 수

있도록 변경하고, 각 Gate 에 Snake 가 통과할 수 있도록 수정한다

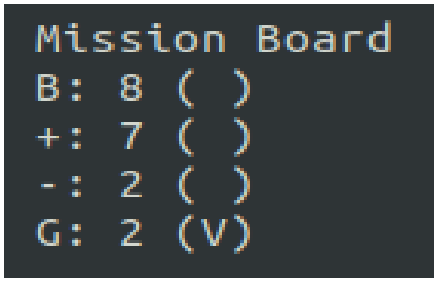
2 개의 gate 를 만들기 위해 각 gate 의 x,y 좌표를 저장하는 변수를 전역변수로 선언하고 Gate 를 만드는 함수 Gate 를 만든다. Gate 역시 임의의 위치에 랜덤으로 생성해야 하므로 Growth Item 과 Poison Item 의 방법과 같이 각 x, y 좌표를 난수로 설정한다. 그러나 Gate 는 Wall 에 생성되도록 해야한다. 즉 map 배열에서 1 에 해당하는 값에 나타나도록 해야하므로 난수로 설정된 x, y 좌표가 map 배열 값 1 에 해당할 때까지 실행하도록 하였다. 조건을 만족하면 Gate 가 생성되도록 하고, Gate 는 map 배열 값 7 로 할당하였다. 2 개의 Gate 를 생성해야 하므로 같은 코드를 한번 더 실행하도록 하였다. 그리고 show 함수에서 배열 7 에 해당하는 값을 게임 화면에 G 로 표현하였다. Snake 가 Gate 에 진입하였을 때의 기능을 구현하기 위해 move 함수에서 관련 기능을 추가하였다. Snake 를 이동한 후 Gate 의 진입 여부를 감지하고 진입한 경우 두 Gate 중 어느 Gate 로 진입하였는지 판단하여 진입한 Gate 이외의 다른 Gate 로 진출하도록 하였다. Game Rule #3 를 참고하여 진출 방향을 설정했다. Snake 가 Gate 를 얼마나 통과해야 하는지 저장하는 변수 gatecount 를 만들어 Snake 의 길이를 저장하고 길이 1 만큼 통과할 때마다 1 씩 값을 감소시킨다. 다 통과한 경우 Gate 를 다시 벽으로 만들고 새 Gate 를 생성하기 위해 gate 함수를 호출하였다.

(5 단계) 4 단계 프로그램에서, 우측에 게임 점수를 표시하는 화면을 구성한다. Mission 을 달성하면 다음 Map 으로 진행하도록 프로그램을 완성한다.

Score Board 에 Snake 의 길이, Item 획득 개수와, Gate 통과 횟수를 저장해야 표시가 가능하므로 각 횟수를 저장하는 전역 변수를 생성한다. 그리고 미션표에 보여질 목표 점수에 해당하는 변수도 같이 생성한다. Item 을 획득했을 때와, Gate 를 통과했을 때 이를 감지하는 기능이 move 함수 내에 구현되어 있으므로 각 조건을 만족하는 경우 move 함수에서 각 변수의 값을 1 씩 증가시키도록 한다. Snake 의 최대 길이는 Growth Item 을 획득할 때마다 현재 길이와 최대 길이를 비교하여 저장한다. Snake 의 현재 길이는 Snake 좌표 vector 의 size 를 통해 알 수 있다.

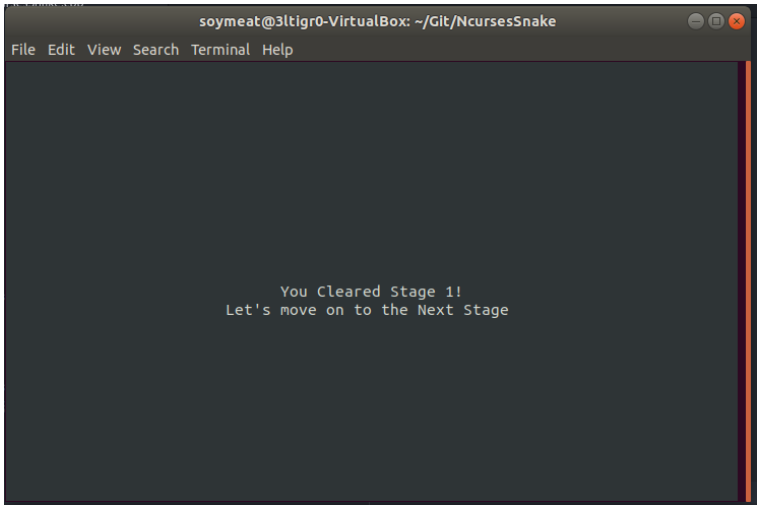
각 미션에 해당하는 목표는 랜덤으로 생성하는 방식을 선택하였다. Body Length 는 5-8, Growth Item 은 3-7 개, Poison Item 은 1-3, Gate 통과 횟수는 1-5 로 각 미션에 해당하는 최소값과 최대값을 두어 이 범위 내에서 난수가 생성되도록 하였다.

Score Board 와 Mission Board 를 게임 화면에 표시하기 위해 원하는 위치에 문자열을 출력해주는 Ncurses Library 함수인 mvprintw 를 사용하였다. mvprintw 의 사용 방식은 mvprint(y 값, x 값, 문자열)이다. 그러나 3 번째 인자에 정수를 단독으로 사용할 수 없다. 그래서 %d 를 하고 해당하는 변수를 4 번째 인자로 넣어주어 값이 출력될 수 있게 하여 show 함수에 작성하였다.



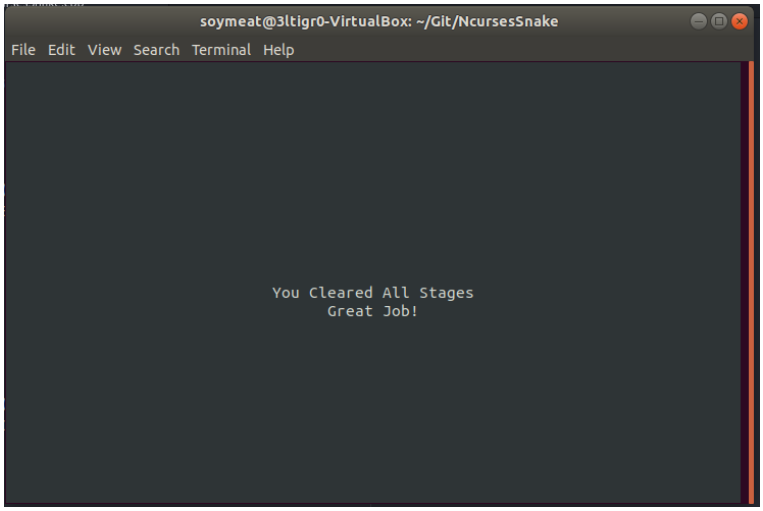
<사진 7. 목표 달성 표시>

각 목표를 달성하였을 때 Mission Board 에서 (V) 표시가 나타나도록 값을 비교하고 표현하는 기능을 작성하였다.



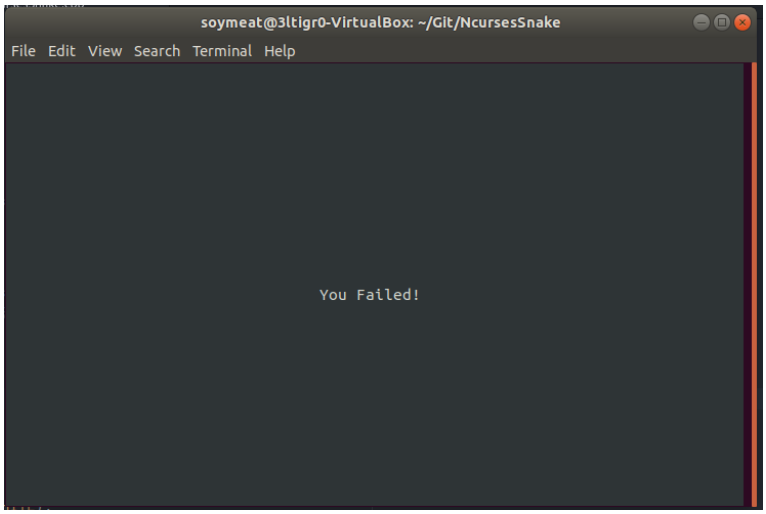
<사진 8. Stage Clear>

Stage 에서 주어진 모든 목표를 달성하는 경우를 판단하기 위해 chkgoal 함수를 만들었다. chkogoal 함수에서는 모든 목표를 클리어하였을는지 판단하여 True 인 경우 Stage 를 Clear 했다는 메시지를 보여주고 setstage 함수를 호출하여 다음 stage 로 이동하게 한다.



<사진 9. All Stage Clear>

Stage 4 를 Clear 하면 main 의 반복문을 빠져나와 모든 스테이지를 Clear 했다는 메시지를 출력하고 게임이 종료되도록 해야하므로 fail 과 clearall 에 true 를 할당하여 실패하였다는 메시지가 아닌 All Stage Clear 메시지가 뜨도록 하였다. 그 후 프로그램이 종료된다.



<사진 10. Failed>

게임 Clear에 실패한 경우 fail 만 true로 바뀌어 main의 반복문을 빠져나온 후 실패 메시지를 출력하고 프로그램이 종료된다.

3 부록

3.1 포함 파일

- C++2020-기말프로젝트-박정재, 최유찬팀-보고서.pdf – 프로젝트 보고서
- NcursesSnake.cpp – 소스코드
- Makefile – Make 파일
- C++2020-기말프로젝트-박정재, 최유찬팀-발표자료.pptx – 프로젝트 요약 ppt

3.2 사용자 매뉴얼

게임 실행 시 Stage 1부터 시작한다. 입력은 방향키(↑ ↓ ←→)를 이용하여 조작할 수 있다. Snake의 이동은 0.5초마다 이루어진다. 흰색은 Wall, 파란색은 Immune Wall, 빨간색은 Poison Item, 초록색은 Growth Item, 보라색은 Gate, 노란색과 청록색은 각각 Snake Head와 Snake Body이다. Growth Item을 획득하면 Body의 길이가 1 증가한다. Poison Item을 획득하면 Body의 길이가 1 감소한다. Gate에 진입하면 다른 Gate로 진출한다. Mission Board에 제시된 목표를 달성하면 다음 Stage로 넘어간다. Mission 목표는 매 게임마다 새롭게 바뀐다. 현재 Score 는 Score Board에서 확인할 수 있다. Snake가 Wall과 충돌하거나 진행 방향과 반대 방향의 키를 입력하거나 Snake의 길이가 3보다 작아지면 실패한다. 총 4개의 Stage로 이루어져있어 모든 Stage를 Clear하거나 실패 시 게임이 종료된다.

3.3 설치 방법

1. 소스코드와 Make 파일을 같은 경로에 둔다.

2. 터미널에서 make all 을 입력한다.
3. 컴파일한 결과로 SnakeGame 파일이 생성된다.