

C++프로그래밍 프로젝트


프로젝트 명	snake game
팀 명	1분반 3팀
문서 제목	결과보고서

Version	2.1
Date	2020-JUN-20

팀원	김 신건 (팀장)
	김 상홍
	김 유진
	이 하영
	최 영락

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 “1분반 3팀”의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 “1분반 3팀”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

문서 정보 / 수정 내역


Filename	1분반_3팀_최종보고서_snake_game.doc
원안작성자	김신건, 김상홍, 김유진, 이하영, 최영락
수정작업자	김신건, 김상홍, 김유진, 이하영, 최영락

수정날짜	대표수정자	Revision	추가/수정항목	내 용
2020-06-15	김신건	1.0	최초 작성	
2020-06-17	김신건	1.1	개발 내용	
2020-06-17	김상홍	1.2	개발 내용	
2020-06-19	전원	2.0	개발 내용	
2020-06-20	전원	2.1	개발 내용	


 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

목 차


1. 개요	6
1.1. 구조 및 개발 내용	6
1.1.1. 전체 구조	6
1.1.2. 개발 내용	6
1.2. 사용한 외부 라이브러리	7
1.2.1. ncurses	7
1.2.2. ncurses 설치 방법	7
1.2.2.1. CentOS7 설치 방법	7
1.2.2.2. Ubuntu	7
1.2.3. make	7
1.2.4. vector	7
2. 개발 내용 및 결과물	8
2.1. 목표	8
2.1.1. 설계	9
2.1.1.1. 게임 루프 설계	9
2.1.1.2. 클래스 참조 관계 및 데이터 처리 설계	9
2.1.1.3. 클래스 명세	9
2.1.2. 구현	9
2.1.2.1. 틱(프레임) 구현	9
2.1.2.2. 인게임 데이터 처리 기능 구현	9
2.1.2.3. Map 구현	9
2.1.2.4. Snake 기능 구현 및 사용자 조작	9
2.1.2.5. Item 요소 구현	9
2.1.2.6. Gate 요소 구현	9
2.1.2.7. 플레이 점수 및 미션 구현 / 렌더링	9
2.1.2.8. 스테이지 확장	9
2.1.3. 확장	10
2.1.3.1. Makefile 작성	10
2.1.3.2. 멀티 플레이 구현 (2인용)	10
2.1.3.3. 인공지능 및 데이터베이스 구현	10
2.1.3.4. 인게임 UI 디자인	10
2.1.3.5. DeltaTime 적용	10
2.1.3.6. 레벨 디자인 (스테이지/미션 밸런스)	10
2.1.3.7. 게임플레이 설명	10
2.1.3.8. 엔딩 크레딧	10
2.2. 개발 내용 및 결과물	10
2.2.1. 개발 내용	10
2.2.1.1. 설계 수행 내용	11

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

2.2.1.1.1. 게임 루프 설계	11
2.2.1.1.2. 클래스 참조 관계 및 데이터 처리 설계	11
2.2.1.1.3. 클래스 명세	11
2.2.1.2. 구현 수행 내용	11
2.2.1.2.1. 틱(프레임) 구현	11
2.2.1.2.2. 인게임 데이터 처리 기능 구현	11
2.2.1.2.3. Map 구현	11
2.2.1.2.4. Snake 기능 구현 및 사용자 조작	12
2.2.1.2.5. Item 요소 구현	12
2.2.1.2.6. Gate 요소 구현	12
2.2.1.2.7. 플레이 점수 및 미션 구현 / 렌더링	12
2.2.1.2.8. 스테이지 확장	12
2.2.1.3. 확장	12
2.2.1.3.1. Makefile 작성	13
2.2.1.3.2. 멀티 플레이 구현 (2인용)	13
2.2.1.3.3. 인공지능 및 데이터베이스 구현	13
2.2.1.3.4. 인게임 UI 디자인	13
2.2.1.3.5. DeltaTime 적용	13
2.2.1.3.6. 레벨 디자인 (스테이지/미션 밸런스)	13
2.2.1.3.7. 게임플레이 설명	13
2.2.1.3.8. 엔딩 크레딧	13
2.2.2. 시스템 구조 및 설계도	14
2.2.2.1. Renderable 클래스	14
2.2.2.2. Kbhut 소스 코드	14
2.2.2.3. NCursesSetting 소스 코드	15
2.2.2.4. SnakeGame 소스 코드	15
2.2.2.5. GameFlow 클래스	15
2.2.2.6. Game 클래스	16
2.2.2.7. Point 클래스	17
2.2.2.8. GameData 클래스	18
2.2.2.9. GameManager 클래스	20
2.2.2.10. SnakeMap 클래스	20
2.2.2.11. Snake 클래스	21
2.2.2.12. Mission 클래스	22
2.2.2.13. Item 클래스	22
2.2.2.14. ItemManager 클래스	23
2.2.2.15. GateManager 클래스	24
2.2.2.16. UserData 클래스	25
2.2.3. 활용/개발된 기술	26
2.2.3.1. ncurses	26
2.2.3.2. vector	26
2.2.3.3. sqrt decompositon	27

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

2.2.3.4. DFS & Grouping	29
2.2.4. 현실적 제한 요소 및 그 해결 방안	30
2.2.4.1. unicode 구현	30
2.2.4.2. tick 구현	30
2.2.4.3. GUI요소 개선	30
2.2.5. 결과물 목록	30
3. 자기평가	32
4. 참고 문헌	35
5. 부록	36
5.1. 사용자 매뉴얼	36
5.1.1. 게임 규칙	36
5.1.1.1. 기본적인 게임 규칙	36
5.1.1.2. 뱀의 이동	36
5.1.1.3. 아이템	36
5.1.1.4. 게이트	36
5.1.1.5. 미션	36
5.2. 설치 방법	37
6. 부록	37
6.1. Github	37
6.2 Youtube	37

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

1. 개요

평가기준 (10점)

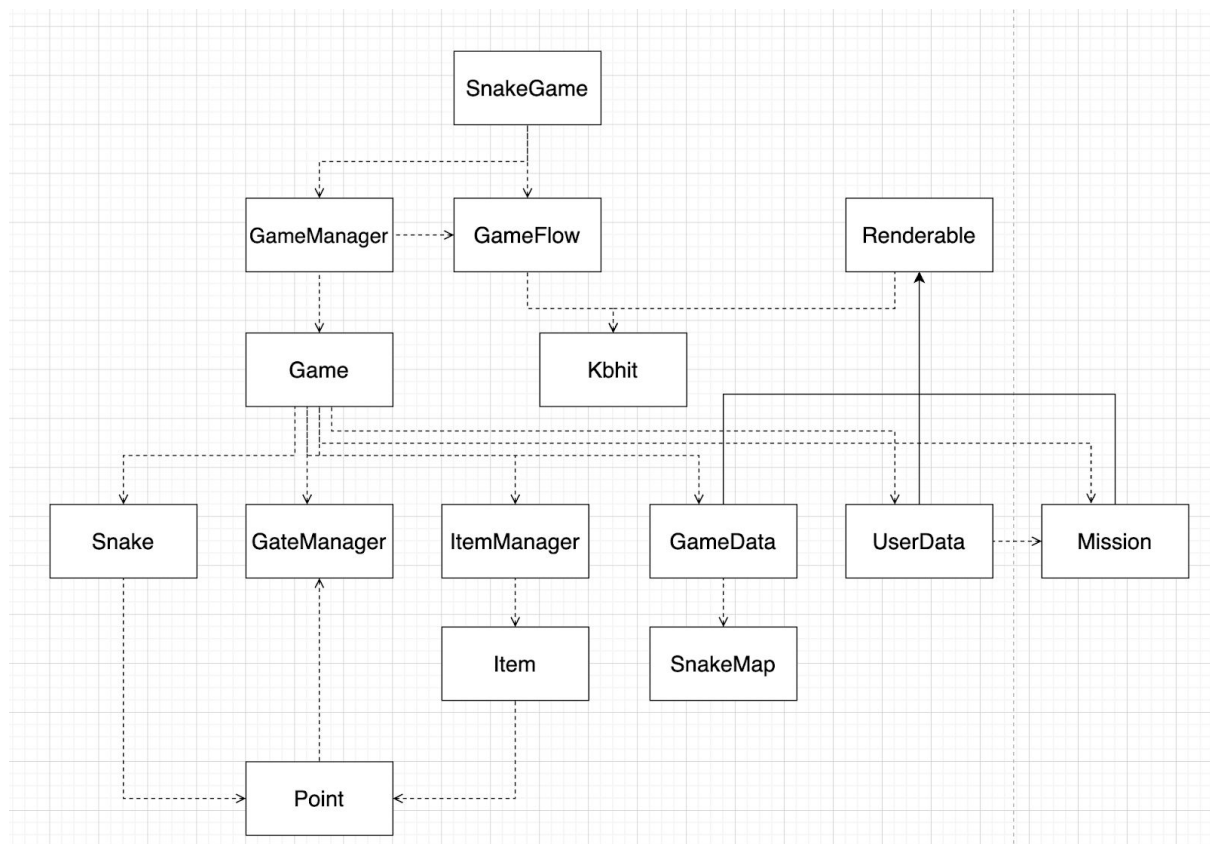
프로젝트를 완성하기 위해 사용한 개발 방법을 기술하세요.

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

프로젝트의 전체적인 구조 및 개발 내용을 명확하게 기술한다.

1.1. 구조 및 개발 내용

1.1.1. 전체 구조




Class 설계를 하면서 member variable과 member function의 개수가 많아질 것을 예상하고, Class 명만 명시한 UML을 작성하였다.

1.1.2. 개발 내용

C++ 언어와 ncurses 라이브러리를 활용하여, snake 게임을 제작한다.

- 사용자에게 보이는 화면은 ncurses 윈도우를 이용해 구현한다.
- C++ 버전은 11을 사용하여 개발하였다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

1.2. 사용한 외부 라이브러리

1.2.1. ncurses

ncurses는 GNU에서 개발한 new curses 라이브러리이며, 1990년대 중반에 curses 라이브러리의 개발이 중단된 후 개발된 라이브러리이다. 이때, curses는 Cursor Optimization에서 유래했으며, 이를 재미있게 발음한 것을 말한다. curses 라이브러리는 유닉스 계열 운영체제를 위한 제어 라이브러리 중 하나이고 매우 유연하고 효율적인 Application Programing Interface를 제공한다. 또한, 커서를 움직이거나, 윈도우를 생성하고, 색깔을 표시하거나, 마우스 관련 함수를 제공하는 등의 TUI 응용 프로그램들의 구성을 가능하게 하는 라이브러리이다.

1.2.2. ncurses 설치 방법

1.2.2.1. CentOS7 설치 방법

```
sudo yum install ncurses-devel
```

1.2.2.2. Ubuntu

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


1.2.3. make

make는 소프트웨어 개발을 위해 유닉스 계열 운영체제에서 주로 사용되는 프로그램 빌드 도구이다. 또한, 여러 파일들끼리의 의존성과 각 파일에 필요한 명령을 정의함으로써 프로그램을 컴파일할 수 있으며 최종 여러 프로그램을 만들 수 있는 과정을 서술할 수 있는 표준 문법을 가지고 있다.

make는 파일 관련 유틸리티로써, 각 파일 간의 종속관계를 파악하여 기술파일에 기술된 대로 컴파일 명령 혹은 셸 명령을 순차적으로 실행한다. 즉, make를 활용하면 각 파일에 대한 반복적인 명령을 자동화시켜서 개발자의 수고를 덜고 시간을 절약할 수 있다.

1.2.4. vector

vector는 C++ STL 라이브러리에 포함되어있는 자료구조 중 하나이며 동적 배열처럼 작동하는 순차 컨테이너이다. STL은 표준 템플릿 라이브러리로 많은 프로그래머들이 공통적으로 사용하는 자료구조와 알고리즘들을 template으로 구현한 클래스이며 std 이름 공간에 포함되어 있다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20


2. 개발 내용 및 결과물

2.1. 목표

작성요령 (10점)

프로젝트의 목표를 기술하세요. 각 단계별 목표를 구체적으로 쓰세요.

적용단계	세부 단계	내용	적용 여부
1.설계	1	게임 루프 설계	적용
	2	클래스 참조 관계 및 데이터 처리 설계	적용
	3	클래스 명세	적용
2.구현	1	틱(프레임) 구현	적용
	2	인게임 데이터 처리 기능 구현	적용
	3	Map 구현	적용
	4	Snake 기능 구현 및 사용자 조작	적용
	5	Item 요소 구현	적용
	6	Gate 요소 구현	적용
	7	플레이 점수 및 미션 구현 / 렌더링	적용
	8	스테이지 확장	적용
3.확장	1	Makefile 작성	적용
	2	멀티 플레이 구현 (2인용)	미적용
	3	인공지능 및 데이터베이스 구현	미적용
	4	인게임 UI 디자인	적용
	5	DeltaTime 적용	미적용
	6	레벨 디자인 (스테이지/미션 밸런스)	적용
	7	게임플레이 설명	적용
	8	엔딩 크레딧	적용

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

프로젝트의 각 적용 단계별 구현 목표를 명확하게 제시한다. 제시한 권고안의 내용을 포함하여, 변경된 부분 등을 구체적으로 단계별 구현 목표를 작성해야 한다.

2.1.1. 설계

2.1.1.1. 게임 루프 설계

: 게임이 실행되며 호출되는 흐름 설계

2.1.1.2. 클래스 참조 관계 및 데이터 처리 설계

: 게임오브젝트들이 서로 참조하는 관계 설계

2.1.1.3. 클래스 명세

: Github 에 클래스 명세를 md 파일로 작성

2.1.2. 구현

2.1.2.1. 틱(프레임) 구현

: 메인에서 일정 시간마다 게임을 업데이트하게끔 틱 단위로 게임 구성

2.1.2.2. 인게임 데이터 처리 기능 구현

: 클래스 별 참조 관계를 줄이기 위한 데이터 클래스 구성

2.1.2.3. Map 구현

: 맵 정보 저장
화면에 렌더링 구성

2.1.2.4. Snake 기능 구현 및 사용자 조작

: 사용자 조작에 따른 움직임
게이트 통과 및 아이템에 대한 처리

2.1.2.5. Item 요소 구현

: 아이템 생성 및 제거

2.1.2.6. Gate 요소 구현


: 제시된 조건에 맞는 게이트를 생성하고 관리해주는 요소들 구성

2.1.2.7. 플레이 점수 및 미션 구현 / 렌더링

: 점수 계산 알고리즘 및 스테이지별 달성 목표 구성
우측에 윈도우 2개 구현하여 각각 정보 출력

2.1.2.8. 스테이지 확장

: n 개의 스테이지로 확장 가능하도록 유연하게 구성

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

2.1.3. 확장

2.1.3.1. Makefile 작성

: 컴파일 편의를 위한 Makefile 작성

2.1.3.2. 멀티 플레이 구현 (2인용)

: 2인 플레이로 서로 견제하면서 목표를 달성하는 기능 구현

2.1.3.3. 인공지능 및 데이터베이스 구현

: 멀티 플레이시 인공지능 플레이어 구현하여 경쟁
플레이 데이터를 저장해놓고 추후 로드할 수 있게끔 DB 에 저장

2.1.3.4. 인게임 UI 디자인

: 스테이지 진입/종료 시 화면 구성
점수 및 미션 정보 구성

2.1.3.5. DeltaTime 적용

: 메인에서 틱 계산시 게임 업데이트에서 처리한 시간을 기준으로 연산
사용자 컴퓨팅 성능에 따른 차이를 줄이기 위한 작업

2.1.3.6. 레벨 디자인 (스테이지/미션 밸런스)


: 각 스테이지별 플레이 밸런스 및 난이도 조정

2.1.3.7. 게임플레이 설명

: 초기 화면에서 게임 플레이 방법에 대해 설명해주는 화면 구성

2.1.3.8. 엔딩 크레딧

: 게임 종료 시 제작자들 렌더링해주는 화면 구성

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

2.2. 개발 내용 및 결과물

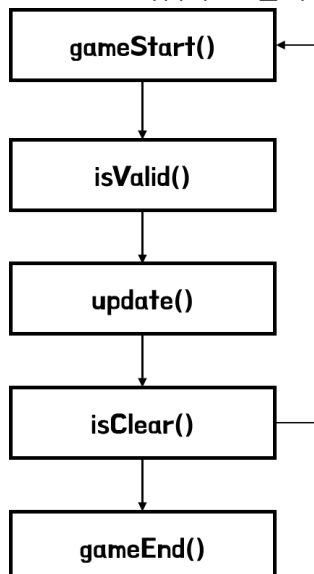
2.2.1. 개발 내용

작성요령 (10점)

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

2.2.1.1. 설계 수행 내용

2.2.1.1.1. 게임 루프 설계



- 유니티의 게임 루프 패턴을 참고하여 스네이크 게임루프를 설계하였다.

2.2.1.1.2. 클래스 참조 관계 및 데이터 처리 설계

- 클래스 간의 참조 관계를 설정하였다. 1.1.1. 전체 구조 참조

2.2.1.1.3. 클래스 명세


- Github 에 markdown 파일로 클래스를 명세하였다. Github 주소는 부록 참조

2.2.1.2. 구현 수행 내용

2.2.1.2.1. 틱(프레임) 구현

-메인에서 일정 시간마다 게임을 업데이트 하게끔 틱 단위로 게임을 구성하였다.

2.2.1.2.2. 인게임 데이터 처리 기능 구현

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

-gate, item, snake기능을 구현하는 클래스에서 같은 데이터를 참조하기 위해 GameData, UserData를 구현하였다. GameData에서는 game에 관한 데이터를 담고있고, 각 필드에 대한 getter, setter를 구현했다. UserData는 user에 대한 데이터를 담고있고, 각 필드에 대한 getter, setter를 구현했다.

2.2.1.2.3. Map 구현

-SnakeMap 클래스를 구현하고, 내부 속성으로 map을 저장하는 vector 동적 배열을 활용하여 게임 Map을 구현하였다. 또한, 이 Map 정보를 다른 Class의 객체와 공유하기 위해 GameData 클래스의 객체가 SnakeMap의 레퍼런스를 저장하고 있는 방식으로 게임오브젝트를 취합하여 맵의 최종적인 맵 정보 저장을 하여 화면에 그려지게 하였다.

-인게임 렌더링에 관련한 처리를 맡아주는 Renderable 클래스를 구현하였다. 해당 클래스를 상속받아 자신에게 맞게끔 윈도우를 구성해주는 가상 함수 render()를 구현하였고, 화면에 Map을 띄우는 세부 목표를 달성하였다.

2.2.1.2.4. Snake 기능 구현 및 사용자 조작

-Snake 클래스를 통해 뱀의 기능을 구현하였다. 뱀의 좌표는 vector<Point> 동적 배열을 통해 관리하였다. 뱀의 움직임은 큐의 성질을 활용하여 틱마다 머리에 넣어주고 꼬리를 빼줌으로써 과도한 데이터 처리 없이 동작하게 하였다. 게이트 통과 및 아이템에 관한 처리는 현재 뱀의 방향에 따른 다음 뱀의 머리의 위치를 저장한 뒤 update() 메소드 내부에서 다음 머리의 위치와 맵을 비교해 처리하며 세부 목표를 달성하였다.

-사용자 조작은 Kbhhit 클래스를 통해 구현하였다. 터미널을 제어하는 라이브러리인 <termios.h>를 사용하여 현재 키보드의 입력 여부 상태를 보고 bool 값을 반환하여, 참일 경우 입력된 키값을 설정해주는 방식으로 뱀의 방향을 결정하였다.

2.2.1.2.5. Item 요소 구현

-Item의 생성과 삭제를 다루는 ItemManager클래스를 구현하였다. Item정보를 업데이트하기위해 update()를 이용하였고, Item의 생성은 makeItem(), 삭제는 deleteItem()을 이용해 구현하였으며, 뱀의 길이는 뱀이 먹은 Item의 종류에 따라 달라진다. 뱀의 다음 머리위치가 아이템이라면 그 아이템 정보를 리턴하고 해당 아이템을 삭제하기위해 eatItem()을 이용하였다.


-Item을 생성하는 makeItem()을 초기에 제작할 때, 무작정 난수를 생성하면서 시간 복잡도가 균일하지 못한 이슈가 있었다. 이에 평방 분할 알고리즘을 활용하여 아이템 생성 알고리즘을 최적화하는 방식을 고안해냈고 이는 2.2.2.3. sqrt decomposition 의 내용에 설명되어 있다.

2.2.1.2.6. Gate 요소 구현

-Gate의 생성과 관리를 담당하는 GateManager클래스를 구현하였다. Gate를 생성할 때, 같은 벽에 게이트 두 개가 생길 수 없으므로 같은 벽을 구분하기 위해 DFS 알고리즘을 이용했다. 게이트의 생성은 srand()를 이용해 구현하였다. 게이트의 나가는 방향은 사용자에게 입력에 따른 우선순위에 대한 방향값들을 선계산하여 비용 소모가 크지 않게 구현하였다.

2.2.1.2.7. 플레이 점수 및 미션 구현 / 렌더링

-플레이 점수를 다루기 위해 UserData 클래스를 구현하였다. 내부 속성으로는 현재 뱀에 관한 정보와 현재 틱을 가지고 있게 하였고, 뱀에 관한 정보는 Snake, ItemManager, GateManager의 update() 메소드에서 UserData 객체를 통해 정보를 수정하게 하였다. 또한 뱀의 정보와 현재 틱을 이용해 점수를 표시하는 세부 목표를 getScore() 메소드를 통해 달성하였다. Map 구현과 같이 Renderable 클래스를 상속받아 render()메소드를 이용해 화면에 표시하는 세부 목표를 달성하였다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트명	snake game	
	팀명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

-미션을 다루기 위해서 Mission 클래스를 구현하였다. Mission 클래스는 각 스테이지마다 미션 목록을 가지고 있고, 매 프레임마다 UserData에서 받은 뱀의 현재 정보와 현 스테이지의 미션을 비교하여 미션이 달성되었는지 확인하도록 isComplete() 메소드를 통해 세부 목표를 달성하였다. 마찬가지로 Renderable 클래스를 상속받아 render()메소드를 이용해 화면에 표시하는 세부 목표를 달성하였다.

2.2.1.2.8. 스테이지 확장

-기존에 스테이지를 확장할 것을 염두해 두고 코드를 구현하였기 때문에, 이에 따라 n 개의 스테이지로 확장 가능하도록 유연하게 구성하는 세부 목표를 달성하기 위해서, map의 개수를 늘리고, 스테이지가 끝나고 시작할 때의 기능들을 개선하였다.

2.2.1.3. 확장

2.2.1.3.1. Makefile 작성

-게임을 실행할 때 필요한 모든 소스 코드를 한 번에 실행하게 하기 위해 Makefile을 작성하였다. 컴파일러로 g++을 사용하였고, 코드 안에 소스 코드들의 파일명을 넣어 세부 목표를 구현하였다. 컴파일 뿐만 아니라 컴파일 후 실행 파일의 실행, 오브젝트파일을 지워주는 기능을 넣어 개발환경을 세팅하였다.

2.2.1.3.2. 멀티 플레이 구현 (2인용)

-2인 플레이로 서로 견제하면서 목표를 달성하는 기능을 구현하려 했으나, 일정 상의 이유로 구현이 힘들었으며, 키보드 입력의 종류를 두 부류로 나누어 구현해야 했기 때문에 해당 세부 목표를 달성하지 못하였다.

2.2.1.3.3. 인공지능 및 데이터베이스 구현

-사용자와 인공지능이 동시에 게임을 플레이하면서, 사용자가 인공지능보다 먼저 게임을 클리어해야 하는 경쟁 게임을 만들려 하였으나, 제대로 된 신경망 알고리즘을 구현하지 못하여 해당 목표를 달성하지 못하였다.

-사용자 이름을 입력받고, 사용자마다 최고 점수를 기록하는 기능을 추가하려 했으나, 일정상의 이유로 구현을 못하였고 해당 세부 목표를 달성하지 못하였다.

2.2.1.3.4. 인게임 UI 디자인


-스테이지에 진입했을 경우와 종료했을 경우에 대한 화면 렌더링을 구현하였고, 점수 및 미션을 구성하는 윈도우를 좀 더 깔끔하게 구성하였다.

2.2.1.3.5. DeltaTime 적용

-게임 화면을 렌더링하는 틱(프레임)을 구현함에 있어서 성능을 개선하기위해 프레임을 고정해주는 deltatime을 구현하려 했으나, 직접적으로 구현하지는 못하였다. 하지만 item과 gate를 생성해주는 알고리즘을 개선하여 프레임 저하를 방지하는 방식으로 세부 목표를 바꾸어 달성하였다.

2.2.1.3.6. 레벨 디자인 (스테이지/미션 밸런스)

-임의로 맵과 미션을 만든 후에 지인을 통해 난이도에 대한 피드백을 받아 게임의 틱 주기와 맵, 미션, 아이템 생성 등을 수정하여 밸런스를 조절하였다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

2.2.1.3.7. 게임플레이 설명

-GameFlow에서 아이템, 게이트, 미션설명 및 조작법을 출력하였다. 시작화면에서 about 을 선택할 경우 화면에 ncurses 라이브러리를 사용하여 해당 설명에 대한 윈도우를 출력해준다.

2.2.1.3.8. 엔딩 크레딧

-GameFlow에서 팀원들의 이름과 학번을 ncurses로 출력하였다. 게임이 종료될 시 출력된다.

2.2.2. 시스템 구조 및 설계도

작성요령 (30점)

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.

2.2.2.1. Renderable 클래스

- 개발자: 김상홍


- 소스 파일: Renderable.h, Renderable.cpp

- attribute

access modifier	type	identifier	description
protected	Window*	window	해당 클래스를 상속받는 자식 클래스 객체들이 각자 제어할 윈도우

- method

access modifier	return	identifier	description
public	void	render()	각자 렌더링 처리를 해 줄 가상함수
private	void	eraseWindow(Window * window)	윈도우를 지우고 해제하는 함수
public		~Renderable()	소멸자

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

2.2.2.2. Kbhit 소스 코드

- 개발자: 김상홍

- 소스 파일: Kbhit.h, Kbhit.cpp

- function

access modifier	return	identifier	description
---	int	kbhit()	사용자가 키보드를 제어중인지 확인

2.2.2.3. NCursesSetting 소스 코드

- 개발자: 김유진

- 소스 파일: NCursesSetting.h

- function

access modifier	return	identifier	description
---	void	NcursesSetting()	ncurses 관련 설정 정보

2.2.2.4. SnakeGame 소스 코드

- 개발자: 이하영, 최영락

- 소스 파일: SnakeGame.cpp

- function

access modifier	return	identifier	description
---	int	main()	게임 실행


2.2.2.5. GameFlow 클래스

- 개발자: 김상홍

- 소스 파일: GameFlow.h, GameFlow.cpp

- attribute

access modifier	type	identifier	description
private	char *	text[400]	엔딩 크레딧 텍스트정보
private	vector<int>	scores	최종 점수 정보

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

private	string	explain[17]	게임 설명에 대한 텍스트정보
---------	--------	-------------	-----------------

- method

access modifier	return	identifier	description
public		GameFlow()	생성자
public	void	renderStartMenu()	게임 시작, 게임 정보, 게임 종료 메뉴를 선택하는 게임 메뉴 창 렌더링
public	void	renderMakers()	엔딩 크레딧 렌더링
public	void	renderGameEnd()	게임 종료시의 화면 렌더링
public	int	renderStageEnter(const int stage)	스테이지 진입시 화면 렌더링
public	int	renderStageClear(const int stage, const int score)	스테이지 클리어시 화면 렌더링
public	void	setStageScore(int stage, int score)	각 스테이지의 점수를 세팅
private	void	eraseWindow(WINDOW * window)	윈도우를 지우고 해제


2.2.2.6. Game 클래스

- 개발자: 김상홍

- 소스 파일: Game.h, Game.cpp

- attribute

access modifier	type	identifier	description
public	bool	is_clear	mission 클리어 판단
public	bool	is_valid	게임 유효성 검사
public	int	key	사용자 입력값
private	int	my_stage	해당 스테이지 번호
private	Snake*	player	해당 스테이지의 플레이어(뱀)
private	GameData*	game_data	해당 스테이지의 게임데이터
private	userData*	user_data	해당 스테이지의 플레이 정보
private	ItemManager*	item_manager	해당 스테이지의 아이템 관리자
private	Mission*	mission	해당 스테이지의 미션 정보

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

private	GateManager*	gate_manager	해당 스테이지의 게이트 관리자
private	int	my_start_tick	스테이지 시작 틱
private	Renderable*	panels[3]	렌더링 객체들

- method

access modifier	return	identifier	description
public		Game()	생성자
public	void	init(const int stage)	게임별 맵 저장, 게임매니저 생성자에서 각각 호출
public	void	setInput()	사용자로부터 입력받은 키보드 값 설정
public	void	gameStart(const int tick)	스테이지 시작 될 때 한번 호출
public	int	isValid()	매프레임 게임 유효성 검사
public	void	update(const int tick)	매 프레임 게임 업데이트
public	bool	isClear()	update 마친 후 호출 (클리어 판단)
public	void	gameEnd()	스테이지 종료 시 한번 호출
public	int	getScore()	UserData에서 점수를 받아옴

2.2.2.7. Point 클래스

- 개발자: 김유진, 최영락


- 소스 파일: Point.h, Point.cpp

- attribute

access modifier	type	identifier	description
public	int	x	x 좌표
public	int	y	y좌표

- method

access modifier	return	identifier	description
public		Point ()	생성자
public		Point(int x, int y)	생성자
public	bool	isValid()	유효한 좌표인지 여부
public	Point	moveTo(const int curDir)	해당 방향으로 이동했을 때 좌표

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

public	Point&	operator=(const Point& a)	대입 연산자 오버로딩
friend	bool	operator==(const Point& x, const Point& y)	operator== 오버로딩
friend	ostream&	operator<<(ostream& outStream, const Point& point)	operator<< 오버로딩 함수

2.2.2.8. GameData 클래스

- 개발자: 김신건


- 소스 파일: GameData.h, GameData.cpp

- attribute


access modifier	type	identifier	description
public	vector<vector<int>>>	mo_count	평방 분할 알고리즘을 활용하면서 item을 만들수 있는 칸의 개수를 저장하는 벡터
public	vector<vector<Point>>>	mo_points	평방 분할 알고리즘으로 나눈 구간마다 담긴 점들의 모음을 나열한 벡터
public	int	sq	평방 분할 알고리즘에 사용되는 구간의 길이 및 구간의 개수
private	Point	next_point	뱀의 머리의 가상 다음 위치
private	Point	next_head_point	뱀의 머리의 실제 다음 위치
private	SnakeMap*	snake_map	맵
private	int	current_tick	현재 틱
private	int	current_direction	뱀의 현재 방향
private	vector<Point>	gates	gate의 좌표를 담아두는 벡터
private	vector<vector<int>>>	gate_directions	gate에 진입/진출 방향에 대해 담아두는 벡터
private	int	key	현재 저장된 키보드 입력 방향

- method

access modifier	return	identifier	description
public		GameData(const int stage)	생성자

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

public	void	update(const int current_tick)	현재 틱을 업데이트한다.
public	int	checkItem(const Point& head)	들어오는 위치가 growth_item, poison_item, 빈칸인지에 따라 1, -1, 0을 return 한다.
public	vector<vector<int>>	getMap()	snakeMap의 현재 스테이지의 맵을 return한다.
public	void	setNextPoint(const Point& next_point)	next_point를 들어오는 점으로 변경한다.
public	void	setNextHeadPoint(const Point& next_head_point)	next_head_point를 들어오는 점으로 변경한다.
public	Point	getNextPoint()	next_point를 return한다.
public	Point	getNextHeadPoint()	next_head_point를 return한다.
public	void	setCurrentTick(const int current_tick)	현재 틱을 업데이트한다.
public	int	getCurrentTick()	현재 틱을 반환한다.
public	void	updateDirection()	키보드 입력 방향을 반영한다.
public	void	updateSnakePosition(const vector<Point>& snake_body)	현재 뱀의 위치를 맵에 새로 반영한다.
public	void	updateItemPosition(const vector<Point>& item_positions, const vector<int>& item_infos)	현재 아이템의 위치와 정보를 맵에 새로 반영한다.
public	void	updateGateDirection(bool isExist, const vector<vector<int>>& gate_directions)	현재 게이트의 위치를 맵에 새로 반영한다.
public	vector<Point>	getGatePositions()	gate의 위치 벡터를 반환한다.
public	int	getCurrentDirection()	현재 뱀의 방향을 반환한다.
public	void	setCurrentDirection(const int current_direction)	현재 뱀의 방향을 반영한다.
public	int	mapReset()	맵을 초기화한다.
public	int	getPositionInfo(const int x, const int y)	맵에서 들어오는 위치에 해당하는 셀의 정보를 return한다.
public	void	setPositionInfo(const int x, const int y, const int info)	맵에서 들어오는 위치에 해당하는 셀의 정보를 수정한다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

public	int	getKey()	key를 반환한다.
public	void	render()	화면에 render한다.
private	wchar_t	changeMap(int i)	셀의 정보를 랜더링 문자로 변경하여 반환한다.

2.2.2.9. GameManager 클래스

- 개발자: 김상홍

- 소스 파일: GameManager.h, GameManager.cpp

- attribute

access modifier	type	identifier	description
private	int	tick	게임의 전체 틱
private	int	curStage	현재 플레이중인 스테이지
private	Game*	games	전체 게임(스테이지)들
private	Game*	curGame	현재 게임(스테이지)
private	Game*	game_flow	게임플로우 렌더링 객체


- method

access modifier	return	identifier	description
public		GameManager(GameFlow& gameflow)	생성자
public	void	start()	첫 스테이지 시작 전 호출
public	void	end()	마지막 스테이지 종료 후 호출
public	void	setInput()	입력받은 키 값 세팅
public	int	isValid()	유효성 검사
public	void	update(const int tick)	게임 업데이트
private	void	stageSetting(const int score)	스테이지 클리어시 관련 정보들 처리

2.2.2.10. SnakeMap 클래스

- 개발자: 김신건

- 소스 파일: SnakeMap.h, SnakeMap.cpp

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

- attribute

access modifier	type	identifier	description
private	int	total_stage_count	전체 게임 스테이지의 갯수
private	vector<vector<vector<int>>>	total_map	Snake game 전체 맵의 초기상태를 담고 있는 3차원 벡터
private	int	current_stage	현재 stage 번호, 0부터 시작
private	vector<vector<int>>	current_map	현재 stage의 맵을 담고 있는 2차원 벡터

- method

access modifier	return	identifier	description
public		SnakeMap()	생성자
public	int	getTotalMapCount()	전체 게임 stage의 개수를 반환한다.
public	vector<vector<int>>	getCurrentMap()	current_map을 반환한다.
public	int	getPositionInfo(int x, int y)	current_map[x][y]를 반환한다.
public	void	setCurrentMap(int next_stage)	current_stage를 next_stage로 set하고, current_map을 totalmap[next_stage]로 변경한다.
public	void	setPositionInfo(int x, int y, int Info)	current_map[x][y]를 info로 변경한다.
public	void	update(vector<Point> snake, vector<Point> gates, vector<Point> items)	뱀, 게이트, 아이템 정보를 맵에 반영한다.


2.2.2.11. Snake 클래스

- 개발자: 김상홍

- 소스 파일: Snake.h, Snake.cpp

- attribute

access modifier	type	identifier	description
private	Point	next_pos	머리가 향할 다음 위치
private	Point	head_pos	머리의 현재 위치

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

private	vector<Point>	bodies	스네이크 몸체들의 위치 (머리 포함)
---------	---------------	--------	----------------------

- method

access modifier	return	identifier	description
public		Snake()	생성자
public	void	update(GameData& game_data, UserData& user_data)	뱀의 움직임 업데이트
public	Point	getNextPoint(const int curDir)	유효성 검사할 때 뱀의 다음 머리 위치 반환
public	int	getSnakeLength()	뱀의 길이 반환

2.2.2.12. Mission 클래스

- 개발자: 최영락, 이하영

- 소스 파일: Mission.h, Mission.cpp

- attribute

access modifier	type	identifier	description
private	vector<bool>	current_mission_state	현재 미션 하나하나의 클리어 상태
private	vector<int>	current_mission_list	현재 라운드의 미션 목록
private	vector<int>	total_mission_list	모든 라운드의 미션 목록
private	vector<int>	current_state	현재 게임의 상태


- method

access modifier	return	identifier	description
public		Mission(const int stage)	생성자
public	int	render()	미션을 화면에 표시한다.
public	void	isComplete()	미션이 클리어되었는지 확인한다.

2.2.2.13. Item 클래스

- 개발자: 최영락

- 소스 파일: Item.h, Item.cpp

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

- attribute

access modifier	type	identifier	description
private	Point	pos	아이템의 위치(positon)
private	int	kinds	아이템의 종류 1: growth item/2: posion item
private	int	created_tick	아이템 생성 시기

- method

access modifier	return	identifier	description
public		Item()	생성자
public	int	getKinds()	아이템의 종류반환
public	int	getCreatedTic()	아이템 생성 시기 반환
public	void	setPos(const Point pos)	아이템의 위치 설정
public	void	setKinds(const int kinds)	아이템의 종류 설정
public	void	setCreatedTic(const int tick)	아이템 생성 시기 설정
public	Point	getPos()	아이템의 위치 반환

2.2.2.14. ItemManager 클래스


- 개발자: 최영락, 김신건

- 소스 파일: ItemManager.h, ItemManager.cpp

- attribute

access modifier	type	identifier	description
private	int	last_made_tick	가장 최근 아이템을 만든 틱
private	int	delay	아이템을 만든 후 다음 아이템을 만들 때까지의 틱
private	int	growth_odd	아이템을 만들 때 성장 아이템이 나올 확률
private	int	disappear_tick	아이템이 사라지는 틱
public	vector<Item>	items	현재 존재하는 아이템 목록

- method

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

access modifier	return	identifier	description
public		ItemManager()	생성자
public	void	makeItem(int current_tick, const vector<vector<int>>& map, GameData &game_data)	아이템을 만들 조건이 되면 아이템을 만든다.
public	int	eatItem(const Point& next_head_point, GameData &game_data)	Snake의 다음 머리 위치가 아이템이라면 아이템의 정보를 리턴하고 그 아이템을 삭제한다.
public	void	deleteItem(const int current_tick, GameData game_data)	어떠한 아이템이 너무 오래 생성되어있으면 그 아이템을 삭제한다.
public	void	update(GameData &game_data, Userdata &user_data)	아이템의 정보를 업데이트한다.


2.2.2.15. GateManager 클래스

- 개발자: 김유진

- 소스 파일: GateManager.h, GateManager.cpp

- attribute

access modifier	type	identifier	description
private	const int	GATE_MANAGER_IMMUNE_WALL	immune wall을 구분하기 위한 상수
private	pair<Point, Point>	gates	생성된 gates를 가지고 있음
private	pair<vector<int>, vector<int>>	gate_directions	gate로 나갈 방향
private	bool	is_passing	뱀이 통과중이면 true, 아니면 false
private	int	live_time	게이트 생존시간
private	int	snake_entered_tick	뱀이 게이트에 처음 들어간 틱
private	int	last_gate_deleted_tick	게이트가 삭제된 틱
private	int[][]	wall_map[MAP_X][MAP_Y]	벽이 있는 곳 표시
private	vector<Point>	wall_list	벽의 좌표
private	int	wall_count	벽 뭉텅이의 개수

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

private	bool	isExist	게이트가 존재 여부
private	const int[][]	DIR_PRIORITY_TABLE[5][5]	방향 우선순위 테이블

- method

access modifier	return	identifier	description
public		GateManager(vector<vector<int>> snake_map)	생성자
private	void	makeWallMap(const int num, const int x, const int y)	맵으로부터 벽을 체크하는 메서드
private	vector<int>	makeGateDirection(Point gate)	게이트 나가는 방향을 구하는 메서드
public	void	makeNewGate()	새로운 게이트 생성
public	void	update(GameData &game_data, UserData &user_data)	매 틱마다 업데이트
public	pair<Point, Point>	getGates()	게이트 반환

2.2.2.16. UserData 클래스

- 개발자: 이하영


- 소스 파일: UserData.h, UserData.cpp

- attribute

access modifier	type	identifier	description
private	int	current_length	현재 뱀의 길이
private	int	max_length	현재 라운드의 뱀의 최대 길이
private	int	growth_item_count	성장 아이템을 먹은 횟수
private	int	posion_item_count	독 아이템을 먹은 횟수
private	int	used_gate_count	게이트를 사용한 횟수

- method

access modifier	return	identifier	description
public		UserData()	생성자

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

public		UserData(int current_length, int max_length, int growth_item_count, int posion_item_count, int used_gate_count)	생성자
public	int	getCurrentLength()	현재 길이 반환
public	void	setCurrentLength(int current_length)	현재 길이 설정
public	int	getMaxLength()	최대 길이 반환
public	void	setMaxLength(int max_length)	최대 길이 설정
public	int	getGrowthItemCount()	성장 아이템 먹은 횟수 반환
public	void	setGrowthItemCount(int growth_item_count)	성장 아이템 먹은 횟수 설정
public	int	getPosionItemCount()	감소 아이템 먹은 횟수 반환
public	void	setPosionCount(int posion_item_count)	감소 아이템 먹은 횟수 설정
public	int	getUsedGateCount()	게이트 사용 횟수 반환
public	void	setUsedGateCount(int used_gate_count)	게이트 사용 횟수 설정
public	void	growthItemIncrease()	성장아이템 먹은 횟수+1
public	void	poisonItemIncrease()	감소아이템 먹은 횟수+1
public	void	usedGateCountIncrease()	게이트를 사용한 횟수+1
public	void	render()	Userdata를 window에 렌더링

2.2.3. 활용/개발된 기술

작성요령 (10점)


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.

2.2.3.1. ncurses

화면에 맵, 스코어보드, 미션 정보 등을 표시하기 위해 ncurses를 사용하게 되었다. ncurses의 기능 중, 윈도우 기능을 사용하여, 맵 윈도우, 스코어보드 윈도우, 미션 윈도우로 분리하여 터미널에 윈도우들을 각각 렌더링하였다. 또한, mvwprintw 등의 출력문을 활용하여 ♥, ♡, ■ 등의 유니코드를 화면에 렌더링하였다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

2.2.3.2. vector

초기에 게임을 설계하는 과정에서 소스코드에서 사용되는 정보들을 저장하는 배열의 크기 등은 동적으로 바뀌는 경우가 많을 것으로 예상하였다. 이에 따라 데이터들을 저장하는 배열로 C++ STL의 vector를 주로 사용하기로 하였다.

vector의 동적 배열 기능을 활용하였고 push_back(), pop_back(), at(), begin(), end() 등의 메서드를 활용하여, 데이터의 저장, 삭제, 정렬등을 할 수 있었다.

2.2.3.3. sqrt decompositon

초기에 아이템을 무작위로 만드는 알고리즘은 다음과 같았다. 먼저 맵 안의 좌표에서 하나를 무작위로 뽑아낸다. 해당 좌표에 벽, 스네이크, 이전에 생성된 아이템이 배치되어있다면 다시 무작위로 뽑고 아니라면 그 좌표를 사용한다.

이 방식은 걸리는 시간 복잡도가 균일하지 않고, 운이 안 좋다면 한 프레임 사이에 굉장한 딜레이가 걸릴 수도 있었다. 따라서, 이를 개선하기 위한 알고리즘을 구상하였다.

sqrt decomposition은 우리말로 평방 분할 알고리즘이라 한다. 이는 어떤 배열에 대해서 특정 데이터를 찾는 쿼리는 $O(\sqrt{n})$ 으로 실행하고, 데이터를 갱신하는 쿼리는 $O(1)$ 로 수행하는 알고리즘 혹은 자료구조이다.


sqrt decomposition의 핵심 아이디어는 배열을 \sqrt{n} 만큼의 길이를 가진 \sqrt{n} 개의 구간으로 나눈 뒤, 구간마다의 정보를 저장하고 조회하자는 아이디어이다.

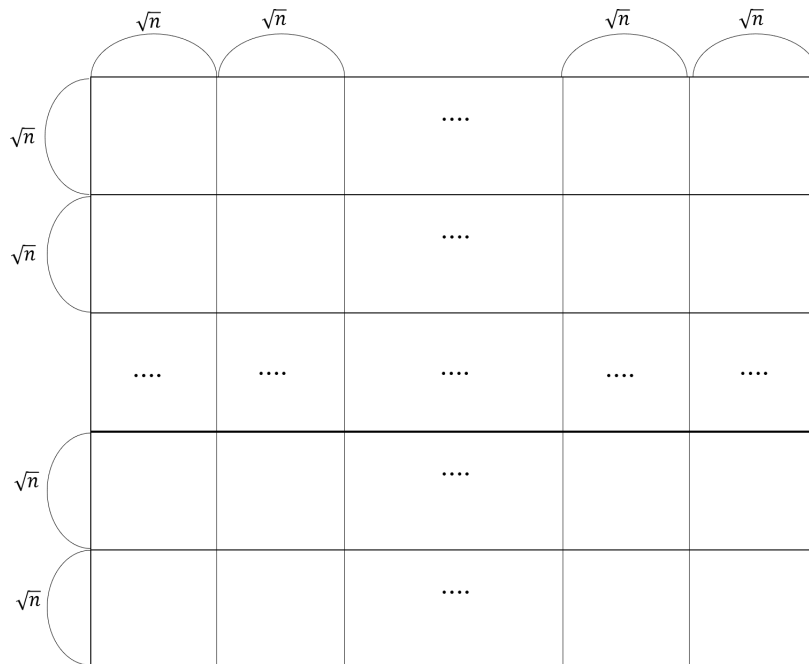
만약 일차원 배열에 sqrt decomposition을 적용한다면, 아래 그림과 같이 구간이 나누어질 것이다.



총 구간의 개수 : \sqrt{n}
 각 구간의 길이: \sqrt{n}

하지만, snake game의 맵은 2차원이고, 만약 한 변의 길이를 n 이라 한다면(맵은 정사각형이라 가정한다.) 전체 $n * n$ 칸이 있다. 이를 sqrt decomposition으로 나누어보면 가로 세로가 각각 \sqrt{n} 인 정사각형이 $\sqrt{n} * \sqrt{n}$ 개 있는 것이므로, n^2 칸의 맵을 아래 그림과 같이 n 개의 구간으로 나타낸 것이 된다.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20



총 구간의 개수 : $\sqrt{n} * \sqrt{n}$
 각 구간의 길이: \sqrt{n}


따라서, 이 나누어진 n 개의 구간마다, 아이템을 만들 수 있는 칸의 개수를 저장/갱신하면서, 특정 셀에 대해서 일부 가중치를 주고 아이템을 가장 만들기 좋은 구간을 고른 뒤, 그 구간 내에서 랜덤으로 위치를 고르면 된다.

이전의 알고리즘은 시간복잡도를 특정할 수 없었지만, 이 알고리즘은 다음과 같이 시간복잡도가 정해진다.

- sqrt decomposition 배열 갱신 : $O(1)$
- 아이템 생성: $\Omega(1)$, $\Theta(n)$

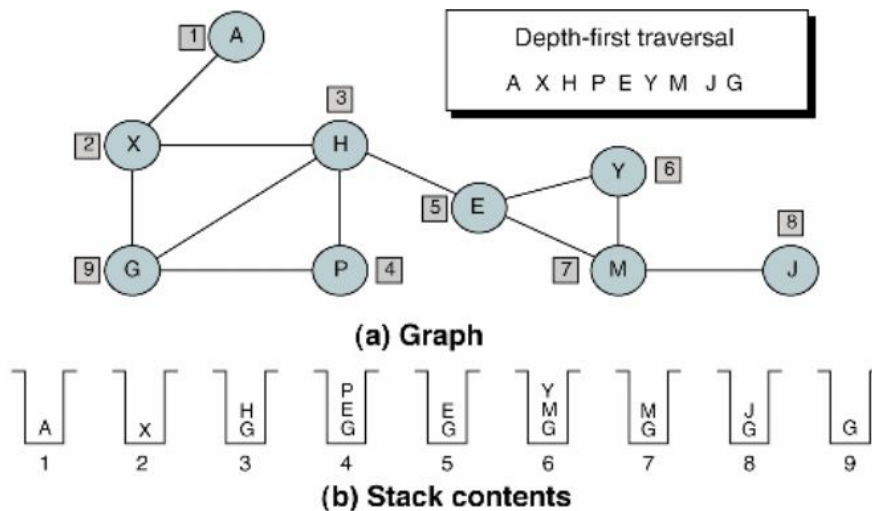
2.2.3.4. DFS & Grouping

Gate를 만들 때 같은 벽에 여러 게이트가 생기면 안되기 때문에, 같은 벽인지 판별해야한다. 이를 위해 map 데이터를 받아서 DFS를 이용해 다음과 같은 형태로 저장한다.

 <div> 국민대학교 소프트웨어학부 C++ 프로그래밍 </div>	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
2																				3
2										4										3
2										4										3
2										4										3
2										4										3
2										4										3
2										4										3
2				4	4	4	4	4	4	4	4									3
2																				3
2																				3
2				5																3
2				5																3
2				5																3
	6	6	6		7	7	7			8	8	8	8	8						3
9				10										8						3
9				10										8						3
9				10										8						3
9														8						3
9														8						3
9																				3
	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	

DFS(Depth First Search)는 갈 수 있는 노드를 스택에 쌓는다. 스택에서 꺼내면서 노드들이 갈 수 있는 노드들을 또 스택에 쌓으면서 스택이 빌 때까지 반복하면 연결되어있는 모든 노드를 방문할 수 있다.




GateManager에서는 재귀를 이용해 구현하였다.

2.2.4. 현실적 제한 요소 및 그 해결 방안

작성요령 (5점)

제안된 프로젝트의 단계 별 수행에 있어, 제한 요소를 찾아 작성한다. 해당 제한 요소를

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

해결하기 위해서 어떤 방법으로 해결하였는지 작성한다.

2.2.4.1. unicode 구현

unicode를 사용하게 되면서 우분투환경에서 특수문자의 width가 절반으로 줄어드는 이슈를 개선하기 위해 unicode출력시, 한칸을 띄워 출력하도록 렌더링 자체를 수정하였다.

2.2.4.2. tick 구현

틱(프레임) 구현에 있어서의 성능을 개선하기 위해 연산 처리량에 따라 프레임을 고정해주는 deltatime을 구현하려 했으나, item과 gate를 생성해주는 알고리즘을 개선하였다.

2.2.4.3. GUI요소 개선


사용자면에서의 구현요소들을 개선하기 위해 각 팀원의 지인들에게 피드백을 요청해 GUI적인 요소, 게임기획, 구성, 레벨디자인등을 수정하였다.

2.2.5. 결과물 목록


작성요령 (5점)

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

파일명	역할
Game.cpp	Game.h에서 정의한 메소드를 구현하는 소스 코드
Game.h	하나의 스테이지의 시작부터 종료까지의 내용들을 다루기 위해 필요한 것을 정의한 소스 코드
GameData.cpp	GameFlow.h에서 정의한 메소드를 구현하는 소스 코드
GameData.h	게임이 실행되는 맵의 정보를 담고 있기 위해 필요한 것을 정의한 소스 코드
GameFlow.cpp	GameFlow.h에서 정의한 메소드를 구현하는 소스 코드
GameFlow.h	게임흐름상 나오는 화면들을 관리하기 위해 필요한 것을 정의한 소스 코드
GameManager.cpp	GameManager.h에서 정의한 메소드를 구현하는 소스 코드
GameManager.h	전반적인 게임을 관리하기 위해 필요한 것을 정의한 소스 코드
GateManager.cpp	GateManager.h에서 정의한 메소드를 구현하는 소스 코드
GateManager.h	게이트의 생성, 삭제를 관리하기 위해 필요한 것을 정의한 소스 코드
Item.cpp	Item.h에서 정의한 메소드를 구현하는 소스 코드
Item.h	아이템의 위치, 타입, 생성 시기를 저장하기 위해 필요한 것을 정의한 소스 코드
ItemManager.cpp	ItemManager.h에서 정의한 메소드를 구현하는 소스 코드
ItemManager.h	아이템의 생성과 삭제를 다루기 위해 필요한 것을 정의한 소스 코드

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

Mission.cpp	Mission.h에서 정의한 메소드를 구현하는 소스 코드
Mission.h	게임의 목표를 정하고, 그 목표를 달성했는지 확인하기 위해 필요한 것들을 정의한 소스 코드
NcursesSetting.h	게임의 정보를 화면에 표시할 때 기본 틀을 정의해놓은 소스 코드
Point.cpp	Point.h에서 정의한 메소드를 구현하는 소스 코드
Point.h	게임의 맵에서의 좌표를 표시하기 위해 필요한 것을 정의해놓은 소스 코드
Renderable.cpp	Renderable.h에서 정의한 메소드를 구현하는 소스 코드
Renderable.h	게임의 정보를 화면에 표시하기 위해 필요한 것을 정의해놓은 소스 코드
Snake.cpp	Snake.h에서 정의한 메소드를 구현하는 소스 코드
Snake.h	뱀의 움직임, 뱀과 아이템, 게이트와의 상호 작용을 구현하기 위해 필요한 것을 정의해놓은 소스 코드
SnakeGame.cpp	전체 게임을 실행시키는 소스 코드
SnakeMap.cpp	SnakeMap.h에서 정의한 메소드를 구현하는 소스 코드
SnakeMap.h	게임이 진행되는 맵의 정보를 가지고 있고, 그 정보를 리턴하기 위해 필요한 것을 정의해놓은 소스 코드
UserData.cpp	UserData.h에서 정의한 메소드를 구현하는 소스 코드
UserData.h	현재 플레이어의 정보를 저장하기 위해 필요한 것을 정의해놓은 소스 코드
kbhit.cpp	kbhit.h에서 정의된 메소드를 구현하는 소스 코드
kbhit.h	키보드의 입력을 받아 리턴하는 행위를 위해 필요한 것을 정의해놓은 소스 코드
MakeFile	프로그램을 실행시키기 위해 필요한 모든 코드들을 한 번에 실행시키는 소스 코드


 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

3. 자기평가


작성요령 (5점)

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.

김신건	<p>본 프로젝트에서 조장의 역할과 UserData 클래스, SnakeMap 클래스, Item 생성 개선 알고리즘 등을 제작하는 역할을 수행하였고 전반적인 코드의 전체 추합을 맡았다.</p> <p>프로젝트를 진행하기 전에 클래스를 설계하는 과정과 프로그램 소스 코드를 구현하는 과정에서 팀원 사이에 의사소통이 잘 안되었다. 이에 따라 구현 사항이 서로 달라졌던 일이 있었는데, 조장으로서 소통을 원활하게 유지하지 못한 잘못을 하였던 것 같다. 이에 대해 반성하고 이후 개발을 하면서 팀원들과의 소통을 중요시 여기고 대화하게 되었다.</p> <p>프로젝트 초기에 Github을 잘 활용하지 못하고 test1, test2 등의 폴더를 만들게 되면서 버전관리가 제대로 되지 않았다. 평소에 깃헙을 사용하던 방식인 git flow를 따르지 못한 팀프로젝트가 된 것 같아 아쉬움을 느낀다.</p> <p>개인적으로, 이번 프로젝트를 통해 nCurses를 다루는 방법과 게임 프로그래밍에 대해서 알아보고 공부할 수 있는 계기가 되었으며, C++을 이용한 게임 프로그래밍 프로젝트를 진행하고 개발하는 방법에 대해 배울 수 있었다.</p>
김상홍	<p>전반적으로 게임이 돌아가는 흐름과 플레이어를 제어하는 역할을 맡았다. 항상 잘 짜여진 게임 엔진을 활용하여 게임을 개발하다가 C++로 하나부터 열까지 만들어볼 수 있다는 것에 설렘, 그래서 스테이지 진입 및 종료 시에 호출되거나 매 프레임마다 게임오브젝트들이 행동하게 하는 등의 각종 게임 루프를 만들어보았다.</p> <p>프레임워크를 만드는 개발자가 되고 싶었는데, 실제로 조원들이 내가 만든 게임 루프 위에서 코드를 작성하는 걸 보니 기분이 좋았다.</p> <p>본격적인 코딩을 시작하기 전에 팀원들과 함께 클래스를 설계하는 과정을 먼저 진행하였다. 이 과정에서 각자 프로그램을 설계하는 개념이 달라서 맞추는데 어려움이 있었지만, 각자의 구조가 가지는 단점을 스스로 말해보며 서로를 이해해나가는 과정 속에서 합의점을 도출해나갔다.</p> <p>직접 가상 함수를 만들어보고, 상속 구조를 짜보는 등 수업 시간에 배웠던 내용을 실제로 적용해서 프로그램을 완성시킨다는 것이 가장 재미있고 뿌듯했다. 수업을 들으면서 이걸 프로젝트에 이렇게 적용해보면 어떨까? 하는 생각이 드는 스스로의 모습이 대견(?)했다.</p> <p>또한 다른 사람들과 함수명, 변수명 등의 작성규칙까지 맞춰보는 경험도 좋았다.</p> <p>물론 평소와는 다른 스타일로 작성한다는것이 다소 어색하고 불편하였지만, 남들과의 협업에 있어서는 이런 사소한 부분들을 맞춰나가며 프로젝트를 완성해나가는것도 중요하다는 걸 느꼈다.</p>

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

	<p>그 외에도, 혼자서는 단순히 개인 저장소처럼 사용하던 것을, 협업을 위해 사용하면서 다른 관점에서 사용해 보는 것도 도움이 많이 되었다.</p> <p>프로젝트 운영에 있어서는 시간을 조금 더 넉넉하게 받아서 학기초부터 진행할 수 있으면 좋을 것 같다. 기존에 목표로 했던 확장 기능들을 구현하지 못한 점도 있지만, 팀원들과도 친해져서 그런지 벌써 끝나나?라는 느낌이 든다.</p> <p>개인적으로는 const 와 레퍼런스에 대해서 능숙하게 사용하지 못한 점이 아쉽다.</p>
김유진	<p>일단 코로나때문에 팀풀을 할 때 오프라인으로 하는게 맞는가? 라는 의문이 들었는데 결과적으로 오프라인으로 했기 때문에 완성도가 높게 나온 것 같다.</p> <p>처음에 주제를 들었을 때 워낙 유명한 게임이기도 하고 주변에서 구현한 사람을 본 것 같아서 그리 어렵지않을 것이라고 예상했다. 하지만 실제로 함께 구체화하다보니 신경써줘야 할 것이 많았다. 게이트, 아이템, 뱀의 업데이트가 조화롭게 잘 이루어져야 원하는대로 동작한다.</p> <p>나는 GateManager클래스를 맡았다. 게이트를 생성하고 관리하는 클래스이다. 금방 할 것 같았는데 생성하는 것부터 만만치 않았다. 알고리즘 문제에 나왔던 방의 크기구하기처럼 DFS알고리즘을 이용하여 해결했다.</p> <p>다수의 인원과 한 프로젝트를 개발하는 것은 처음이었다. 그래서 함께 클래스 명세를 먼저 했다. 확실히 각자 어떻게 구현할지 생각해 놓은게 달라서 커뮤니케이션을 통해 맞춰갔다. 이 작업이 제일 오래걸렸고 점점 산으로 가는 느낌이 중간에 들었지만 그래도 잘 마무리했다.</p>
이하영	<p>본 프로젝트에서 UserData클래스, Item클래스등을 구현하였다.</p> <p>프로젝트 초기에 nCurses라는 생소한 라이브러리에대한 기본사용법부터, 한글깨짐 현상등 어려운점이 많았지만 이번 프로젝트를 통해 nCurses를 다루는 방법과 GUI 응용 소프트웨어를 개발하는 방법에 대해 배울 수 있었다.</p> <p>또한, 프로젝트를 진행하면서 초기에는 github을 잘 활용하지 못하고 버전관리가 제대로 되지 않아서 아쉬웠지만, 후에 vs live share, git등을 잘 활용하여 비교적 원활하게 소통하며 협업하였다.</p> <p>개발 프로젝트가 처음이었기 때문에 설계하는 과정에서부터 구현과정, 알고리즘, 변수명이나 함수명설정같은 사소한 코딩 습관까지 부족한게 많았지만 팀원들에게 도움을 받아 무사히 프로젝트를 마칠 수 있었다.</p> <p>프로젝트를 마무리하면서 마우스동작, 배경음악, 2인용구현등 구현하고 싶은 내용이 많았지만 시간상 구현하지 못한 요소들이 많아 아쉬움이 남았다.</p>
최영락	<p>이번 프로젝트에서 ItemManager 클래스, Mission 클래스를 담당했다.</p> <p>ItemManager 클래스를 작성하기 전에는 머릿속으로 어떻게 코드가 동작할지 다 정한 다음 코드를 작성했지만 실제 코드를 작성해보니 생각과는 다른점이 매우 많았다. 코드의 사소한 부분에서 자꾸 오류가 나서 팀원들의 도움으로 수정하였다. 또한 코드를 완성해도 단순 랜덤의 반복을 통해 만들었기에 시간이 오래 걸리는 문제가 있었다. 이는 김신건 학생이 평방분할</p>

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트명	snake game	
	팀명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

	<p>알고리즘을 통해 해결하였다.</p> <p>이번 프로젝트에서 내가 다른 팀원들에 비해 능력이 부족하다고 느껴 중심이 되는 코드나 어려운 알고리즘을 사용하는 코드를 담당하지 못하였는데, 내가 조금 더 실력이 좋았다면 더 팀에 도움이 될 수 있었는데 그렇지 못해 아쉬웠다.</p> <p>처음으로 여러 사람들과 힘을 합쳐 프로젝트를 하였는데, 생각보다 코딩 습관이 여러 사람이 모이는 프로젝트에서 큰 영향을 끼친다는 것을 알게 되었다. 또한, 프로젝트를 진행할 때 사용하는 절차와 서로 마찰이 있을 때 이를 해결하는 방법에 대해 실제로 느껴볼 수 있는 좋은 경험이었다.</p>
--	---


4. 참고 문헌

번호	종류	제목	출처	발행년도	저자	기타
1	웹사이트	Capture characters from standard input without waiting for enter to be pressed(Stack Overflow)	https://stackoverflow.com/questions/421860/capture-characters-from-standard-input-without-waiting-for-enter-to-be-pressed	2009	Adam	
2	웹사이트	평방 분할(Square Root Decomposition), 모스 알고리즘(Mo's Algorithm)	http://blog.naver.com/PostView.nhn?blogId=kks227&logNo=221401154455&parentCategoryNo=311&categoryNo=&viewDate=&isShowPopularPosts=true&from=search	2018	라이	
3	웹사이트 이미지	그래프 탐색	https://www.zerocho.com/category/Algorithm/post/5870153c37e1c80018b64eb0	2017	ZeroCho	

5. 부록

작성요령 (15점)

프로젝트의 결과물을 사용하기 위한 방법에 대해서 작성하세요.

 국민대학교 소프트웨어학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

5.1. 사용자 매뉴얼

5.1.1. 게임 규칙

5.1.1.1. 기본적인 게임 규칙

- 뱀은 0.25초(1틱)에 한칸씩 움직인다.
- 사용자는 뱀을 상하좌우로 움직일 수 있으며, 벽에 닿으면 게임이 끝난다.
- 벽이 아닌 곳에 아이템이 생성되며, 먹으면 뱀의 길이가 변한다.
- 게이트로 들어가면 반대 게이트로 나올 수 있다.

5.1.1.2. 뱀의 이동

- 뱀은 상하좌우로 움직일 수 있으며, 아래와 같은 방향키로 움직일 수 있다.
 - w : 상
 - s : 하
 - a : 좌
 - d : 우
- 뱀의 머리가 자신의 몸이나 벽으로 이동할 수 없으며, 해당 위치로 이동하는 경우 Game over이다.

5.1.1.3. 아이템

- Item은 Growth Item과 Poison Item으로, 2 종류가 있다.
- Growth Item과 Poison Item의 생성 비율은 7 : 3 이다.
- ♥ : Growth Item
 - 이 아이템을 먹으면 뱀의 몸 길이가 1 길어진다.
 - 아이템 생성된 시점에서 40틱 이후 없어진다.
 - 아이템이 없어진 시점에서 8틱 이후 새로운 아이템이 생성된다.
- ♡ : Poison Item
 - 이 아이템을 먹으면 뱀의 몸 길이가 1 짧아진다.
 - 아이템 생성된 시점에서 40틱 이후 없어진다.
 - 아이템이 없어진 시점에서 8틱 이후 새로운 아이템이 생성된다.

5.1.1.4. 게이트

- ■ : gate
- 게이트 입구로 들어가면 다른 게이트로 나온다.
- 게이트를 타면 해당 게이트는 사라진다.
- 게이트는 5틱이후 다시 생성된다.
- 같은 벽에 2개의 게이트는 생성되지 않는다.


5.1.1.5. 미션

- 미션은 4가지 종류가 있다.
- 'B' : 현재 몸 길이와 지금까지 몸 길이중 가장 길었던 몸 길이가 출력된다. (현재 몸 길이) / (가장 긴 몸 길이)
- '+' : Growth Item을 먹은 횟수
- '-' : Poison Item을 먹은 횟수
- 'G' : Gate에 들어간 횟수

5.2. 설치 방법

1. g++ 컴파일러 설치/업그레이드(g++ 버전은 7.x 버전을 사용한다. ex. 7.5.0)

```
$ sudo apt-get install -y software-properties-common
```

 <div> 국민대학교 소프트웨어학부 C++ 프로그래밍 </div>	결과보고서		
	프로젝트 명	snake game	
	팀 명	3팀	
	1분반 3팀	Version 2.1	2020-06-20

```
$ sudo add-apt-repository ppa:ubuntu-toolchain-r/test
$ sudo apt update
$ sudo apt install g++-7 -y

$ sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-7 60 \
--slave /usr/bin/g++ g++ /usr/bin/g++-7
$ sudo update-alternatives --config gcc
```

2. 게임소스 코드 파일을 다운로드 한다.

- 방법1: git clone 사용

```
$ git clone https://github.com/shinkeonkim/snake-game.git
```

- 방법2: 문서와 함께 제출한 zip 파일 압축을 해제한다.

3. make 명령어로 컴파일 한다.

```
$ cd src
$ make
```

4. make run 명령어로 실행파일(게임)을 실행한다.

```
$ make run
```

6. 부록

6.1. Github

Github repository : <https://github.com/shinkeonkim/snake-game>

6.2 Youtube

플레이 영상 : <https://www.youtube.com/watch?v=U7oR8bssnOs>