

C++프로그래밍 프로젝트


프로젝트 명	snake game
팀 명	A++ 프로그래밍
문서 제목	최종보고서_SNAKEGAME

Version	1.0
Date	2020-JUN-26

팀원	김서경(20191560)
	이소영(20191641)
	이아연(20191643)

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 “A++ 프로그래밍”의 팀원들의 자산입니다. 국민대학교 소프트웨어학부 및 팀 “A++프로그래밍”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

문서 정보 / 수정 내역


Filename	최종보고서_SnakeGame.doc
원안작성자	김서경, 이소영, 이아연
수정작업자	김서경, 이소영, 이아연

수정날짜	대표수정자	Revision	추가/수정항목	내 용
2020-06-26	이아연	1.0	내용 작성	전체 내용 작성

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26


목 차

개요	4
개발 내용 및 결과물	5
목표	5
1단계 목표: 화면 구성	5
2단계 목표: Snake 출력 및 이동	5
3단계 목표: Item 출현 및 효과	5
4단계 목표: Gate 출현 및 효과	5
5단계 목표: 게임 점수 표시 화면 구성	6
개발 내용 및 결과물	6
개발 내용	6
시스템 구조 및 설계도	7
활용/개발된 기술	15
현실적 제한 요소 및 그 해결 방안	16
결과물 목록	16
자기평가	18
참고 문헌	19
부록	19
사용자 매뉴얼	19
설치 방법	19

 국민대학교 컴퓨터공학부 C++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

1 개요

- 프로젝트 개요:
 - 2020년도 국민대학교 “C++ 프로그래밍” 교과목에서 배운 내용을 활용 및 심화학습하기 위해 해당 프로젝트를 진행하였다.
 - 프로젝트는 C++ 언어를 이용한 스네이크 게임의 구현을 목표로 하였다. GUI의 구현엔 ncurses 라이브러리를 이용하였다.
- 개발방법:
 - 단계별 구현을 기본으로 하였다. 단계별로 요구사항에 따라 파트를 나누어 각자 구현 및 단위테스트를 진행 후 합쳐서 통합 테스트를 진행 후 다음 단계로 넘어갔다. 코드는 깃허브를 이용해 관리하였다.
- 사용한 라이브러리:
 - random: 난수 발생을 위해 이용
 - unistd.h: 난수 발생을 위해 이용
 - thread: 게임과 아이템 발생을 각각 독립적으로 동시에 진행하기 위해 이용
 - locale.h: 한글 사용을 위해 이용
 - chrono: 시간 측정을 위해 이용
- 외부 라이브러리 참조 : <ncurses>
 - ncurses 라이브러리 설치 방법: 교과목 강의에서 안내한 내용대로 진행하였다.
- 게임 규칙
 - 기본: 방향키로 스네이크를 조작해 아이템과 게이트를 획득하여 단계별 미션을 클리어하는 게임
 - 세부사항:
 - 단계별 미션 조건을 통과하면 다음 스테이지로 넘어간다.
: 몸 길이, grow Item 획득 횟수, poison Item 획득 횟수, gate 통과 횟수, 생존 시간
 - 스네이크는 0.5초마다 이동한다.
 - grow Item 획득 시 스네이크의 몸 길이가 1 증가하고, poison Item 획득 시 몸 길이가 1 감소한다.
 - gate 통과 시 짝이 되는 다른 gate로 진출한다.
 - 스네이크 헤드의 진행 방향과 같은 방향의 키 입력은 무시한다.
 - 게임 오버:
 - 스네이크 헤드와 바디 또는 벽 충돌
 - 스네이크 몸 길이가 3 미만

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

2 개발 내용 및 결과물

2.1 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: 화면 구성

기본 규칙

- main window 를 생성하고, 게임 실행 윈도우인 sub window를 생성한다.
- sub window에서 snake game의 맵을 구현한다.
- 맵의 크기는 21x40로 한다.

추가 사항

- 총 8개의 맵이 존재하며, 스테이지 시작 시 임의의 맵이 랜덤으로 선택되어 나타난다.

2단계 목표: Snake 출력 및 이동

기본 규칙

- map위에 snake를 출력한다.
- 사용자로부터 키입력을 받아 0.5초마다 snake가 움직이도록 기능을 구현한다.
- snake의 head가 body 또는 wall에 접촉했을 때 게임이 실패한다.
- snake의 길이가 3보다 작아지면 게임이 실패한다.

3단계 목표: Item 출현 및 효과


기본 규칙

- growth item과 poison item을 랜덤으로 출현시킨다. items는 출현 후 일정 시간이 지나면 사라지고, 다른 위치에 다시 출현한다.
- 동시에 출현하는 item의 최대 갯수는 3개로 제한한다.
- growth item : 획득시 body의 길이가 1 증가한다. (body는 꼬리 부분이 증가한다)
- poison item : 획득시 body의 길이가 1 감소한다 (꼬리부분이 감소한다.)

4단계 목표: Gate 출현 및 효과

기본 규칙

- 일정 시간이 지나면 gate를 랜덤으로 출현시킨다.
- gate는 출현 후 일정시간이 지나면 삭제된다.
- gate는 한번에 한쌍 출현한다.
- 하나의 gate에 진입하면 다른 나머지 gate로 진출한다.
- 진출방향 : gate가 가장자리 wall에 위치할 땐 항상 Map의 안쪽방향으로 진출

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

gate가 map의 가운데에 위치할 땐 다음의 우선 순위에 따른 방향으로 진출
-> 1. 진입방향과 일치하는 방향 2.진입방향의 오른쪽
3.진입방향의 왼쪽 4.진입방향의 반대방향

5단계 목표: 게임 점수 표시 화면 구성

기본 규칙

- 게임점수를 표시하는 윈도우 score board와 미션을 표시하는 윈도우 mission board를 sub window로 추가한다.
- score board : 현재 snake size, 획득한 growth item, 획득한 poison item, gate사용횟수, 스테이지 진행 시간을 나타낸다.
- mission board : 스테이지마다 지정된 미션들과, 달성현황을 나타낸다.
- 미션을 모두 달성했을 때 다음 스테이지로 이동한다.
- 4스테이지를 클리어했을 때 게임이 완전히 종료된다.

2.2 개발 내용 및 결과물

2.2.1 개발 내용

1. Snake의 기능 구현


- head와 body 구현 : single linked list 사용
- 화면에 snake 출력 : map배열의 head 부분을 3, body 부분을 4로 지정
- move 구현 : snake body의 맨 끝 요소 삭제 후 head 변경
- body 증가 : snake body의 맨 끝 요소 추가
- body 감소 : snake body의 맨 끝 요소 삭제
- 벽, 아이템, 게이트, 자신의 body와의 충돌 감지 : map에서 head가 위치한 부분이 해당 배열값(벽 = 1, 아이템 = 5 or 6, 게이트 = 7, body = 4)이라면 기능 수행
- 게이트 통과 시 방향과 이동 결정
- (전역함수) 가장자리 벽 구분(Upper Edge, Right Edge, Left Edge, Lower Edge)

2. Item 출현 및 삭제 구현

- growItem 출현 및 삭제
- posionItem 출현 및 삭제
- gate 출현 및 삭제

3. 다양한 map 배열 생성

- 8가지의 맵 배열 생성 : wall = 1, immune wall = 2

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

4. map의 배열값에 따른 출력 결정

- map의 배열값에 따라 색을 다르게 해서 출력(wall, immune wall = BLACK, head = YELLOW, body = CYAN, growlItem = GREEN, poisonItem = RED, gate = MAGENTA)

5. 스테이지 레벨별 미션 설정 및 미션 진행상황 표시

- 각 스테이지 레벨마다 새로운 미션을 설정
- 해당 스테이지의 미션이 진행여부 설정 - 각 미션이 완료되면 1을 출력한다.

6. 스코어 보드, 미션 보드 및 화면전환 설정

- 게임 시작 화면
- 게임 실패 화면
- 게임 올클리어 화면
- 각 스테이지 시작 전 화면
- 스코어 보드, 미션 보드

7. 전체 게임 실행(main)

- 게임 진행 시간 기록
- snake 동작 함수 : snake class의 각 함수 사용
- 미션 및 맵 선택
- 게임 진행, 스테이지 전환, 종료 판단 : 게임 실행 변수(bool 타입)과 stageLevel(5 이상이면 clear)등을 이용해서 게임의 스테이지와 진행여부를 판단, 각 아이템과 게이트의 생성은 게임과 독립적으로 동시에 실행하기 위해서 thread를 사용함.

8. Position 구조체

- row, column값 저장

2.2.2 시스템 구조 및 설계도

1. Snake.h - Snake .cpp


Element class: Snake 헤드, 바디 요소로 이용할 클래스

- variables
 - int r, c: 맵에서의 좌표
 - Element* next: 다음 바디 포인터

Element(int a, int b): 클래스 멤버 변수를 초기화하는 생성자

개발자: 이소영(100%)

매개변수로 받은 a, b를 각각 r, c 값으로 할당해준다. next는 0으로 초기화해준다.

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

Snake class: 스네이크 헤드와 바디를 저장 및 스네이크 관련 함수를 구현한 클래스

• variables

- Element* head: 스네이크 헤드를 저장
- std::chrono::system_clock::time_point mvSpan: 이동 시간 제어에 이용
- int dir: 스네이크 헤드의 진행 방향
- int size: 스네이크 사이즈
- position offset[4]: 이동에 이용할 이동좌표

Snake(): 멤버 변수 초기화 및 초기 스네이크를 생성하는 생성자

개발자: 이소영(100%)

초기 스네이크 생성: 초기 스네이크의 몸길이가 3이 되도록 스네이크 헤드와 바디를 생성한다.
클래스 멤버 변수 초기화: 초기 진행 방향은 위쪽으로, 스네이크 사이즈는 3으로 초기화한다. mvSpan에 스네이크를 생성한 시간을 기록한다.
offset 초기화: 각각의 이동 방향에 따른 좌표의 변화값을 이용해 스네이크에 이동에 이용할 이동 좌표를 설정해준다.

~Snake(): 동적 할당으로 생성한 변수를 정리하는 소멸자

개발자: 이소영(100%)

스네이크 헤드 및 바디는 동적으로 할당하였으므로 링크드리스트를 돌며 메모리를 해제해준다.

void printSnake() : 화면에 스네이크를 출력하는 함수

개발자 : 김서경(100%)


스네이크 리스트의 모든 노드들을 while문을 사용하여 맵 배열에 업데이트한다. 이때 삼항 연산자를 사용하여 head와 body의 값을 다르게한다.

void addbody() : 스네이크의 바디를 1만큼 길어지게 하는 함수

개발자 : 김서경(바디 생성구현 60%) , 이아연(충돌처리 구현)

바디생성 : 현재 Snake의 맨뒤에서 첫번째 노드, 두번째 노드의 좌표를 활용하여 새로운 바디가 생성될 좌표를 알아낸다. ((first좌표)+(first좌표 - second좌표)) 해당 좌표를 새로운 바디의 좌표로 설정하고 맨 뒤 노드에 링크한다.

void removebody() : 스네이크의 바디를 1만큼 줄어들게 하는 함수

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

개발자 : 김서경(100%)

현재 Snake의 맨뒤 노드를 delete한다. delete한 노드의 배열좌표를 0으로 바꿔준다.

bool isBody() : 헤드가 바디에 닿았는지 확인하는 함수

개발자 : 김서경(100%)

현재 맵배열에서 다음이동할 위치가 body이면 true리턴한다.

bool isWall() : 헤드가 벽에 닿았는지 확인하는 함수

개발자 : 김서경(100%)

현재 맵배열에서 다음이동할 위치가 wall이면 true리턴한다.

void isGrowItem() :헤드가 grow아이템에 닿았는지 확인해 처리하는 함수

개발자 : 김서경(100%)

현재 맵배열에서 다음이동할 위치가 growthitem이면 addbody함수를 호출하고 growthItem++한다.

void isPoisonItem() :헤드가 poison아이템에 닿았는지 확인해 처리하는 함수

개발자 : 김서경(100%)

현재 맵배열에서 다음이동할 위치가 poisonitem이면 removebody함수를 호출하고poisonItem++한다.

void isGate() :헤드가 gate에 닿았는지 확인해 처리하는 함수

개발자 : 김서경(100%)


현재 맵배열에서 다음이동할 위치가 gate이면 맵의 가장자리인지 확인하고 확인결과를 인수로 전달하여 passgate함수를 호출한다.

void move(int d): 스네이크 이동을 구현한 함수

개발자 : 이소영(100%)

스네이크의 이동은 헤드와 매개변수로 받은 이동방향을 합한 좌표로의 헤드 변화와 스네이크의 마지막 바디의 삭제로 구현하였다.

새 Element 생성: 헤드의 좌표와 이동방향을 합한 좌표를 이용해 생성한다.

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

마지막 바디 삭제: while loop을 이용해 마지막의 이전 요소로 이동 후 마지막 요소와의 연결을 제거하고 마지막 요소를 메모리에서 해제한다.
head 변경: 새로 생성한 Element로 헤드를 변경하고 매개변수로 받은 이동 방향을 head의 이동 방향으로 설정한다.

void passGate(position to): 게이트 통과 기능을 구현한 함수

개발자: 이소영(100%)

게이트 통과 후 방향 설정: 기존 방향과 방향 설정 우선순위에 따른 새로운 방향 사이의 관계식을 구해 우선 순위에 맞춰 통과 후 이동 방향을 담은 이동 방향 배열(next_dir)을 만든다.
가장자리 벽에 위치한 게이트인 경우: isEdge() 함수를 이용해 이동할 게이트가 맵의 가장자리에 있는 벽인지 검사한다. 맵의 가장자리인 경우 isEdge() 함수의 리턴값을 이용해 이동 방향을 설정하고 스네이크의 헤드 좌표를 이동한다.
맵 중간에 위치한 게이트의 경우: while loop과 next_dir배열을 이용해 이동방향 우선순위에 따라 다음 좌표를 계산하고 다음 좌표로 이동이 가능한 경우 스네이크의 헤드 좌표를 이동한다.

int isEdge(position to): 매개변수로 받은 좌표가 가장자리 벽에 위치했는지 검사하는 함수

개발자: 이소영(100%)

매개변수로 받은 좌표와 맵의 마지막 행과 열의 값을 비교해 가장자리 벽인지 검사한다.
가장자리 벽인 경우: 해당 벽의 게이트를 통과할 때의 진행 방향을 리턴한다.
가장자리 벽이 아닌 경우: 4를 리턴한다.


2. Items.h - Items.cpp

• variables

- **extern bool go** : 게임이 실행중인지 판단
- **int itemCnt** : 아이템 개수. 3개 이하일때 아이템 생성
- **extern position g1** : gate1의 위치(row, column)값을 저장하는 position 구조체
- **extern position g2** : gate2의 위치(row, column)값을 저장하는 position 구조체

void makeGrowltem() : Growltem의 출현 및 삭제 함수

개발자: 이아연(100%)

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

- 1) 첫 아이템 출현 시간은 게임 시작 후 3초에서 7초 후 사이에서 랜덤하게 정한다.
- 2) `go == true`(게임 실행 중)일 경우 while loop를 실행한다.
- 3) `itemCnt < 3`일 경우 아이템을 생성한다.
- 4) 아이템을 생성할 위치를 난수로 받는다.
- 5) 해당 위치의 배열값이 0이라면, `itemCnt`를 증가하고, 해당 위치의 배열값을 5로 바꾼다.
- 6) 아이템이 존재하는 시간을 11초에서 15초 사이에서 랜덤하게 정한다.(그 시간 동안 sleep)
- 7) 해당 시간이 지난 후, 아이템 위치의 배열값을 0으로 바꾸고, `itemCnt`를 감소한다.
- 8) 3초에서 7초간 동작을 멈춘다.(다음 아이템이 바로 등장하지 않도록)
- 9) 3단계 ~ 8단계를 while loop 내부에서 실행

void makePosionItem() : PoisonItem의 출현 및 삭제 함수


개발자: 이아연(100%)

- 1) 첫 아이템 출현 시간은 게임 시작 후 3초에서 7초 후 사이에서 랜덤하게 정한다.
- 2) `go == true`(게임 실행 중)일 경우 while loop를 실행한다.
- 3) `itemCnt < 3`일 경우 아이템을 생성한다.
- 4) 아이템을 생성할 위치를 난수로 받는다.
- 5) 해당 위치의 배열값이 0이라면, `itemCnt`를 증가하고, 해당 위치의 배열값을 6으로 바꾼다.
- 6) 아이템이 존재하는 시간을 11초에서 15초 사이에서 랜덤하게 정한다.(그 시간 동안 sleep)
- 7) 해당 시간이 지난 후, 아이템 위치의 배열값을 0으로 바꾸고, `itemCnt`를 감소한다.
- 8) 3초에서 7초간 동작을 멈춘다. (다음 아이템이 바로 등장하지 않도록)
- 9) 3단계 ~ 8단계를 while loop 내부에서 실행

void makeGate() : Gate의 출현 및 삭제 함수

개발자 : 이아연(100%)

- 1) 첫 게이트 출현 시간은 게임 시작 후 3초에서 7초 사이에서 랜덤하게 정한다.
- 2) `go == true`(게임 실행 중)일 경우 while loop를 실행한다.
- 3) 게이트1의 위치(row, column), 게이트 2의 위치를 난수로 받는다.
- 4) 게이트1과 게이트2의 위치가 같을 경우 다시 난수를 받도록 while loop를 돌려준다.
- 5) 이전 해당 위치의 배열값을 저장하는 `prev_g1`, `prev_g2`를 선언한다.
- 6) 게이트1과 게이트2 위치의 배열값이 0 또는 1(벽)일 경우, 해당 위치의

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

배열값을 7로 바꾼다.

- 7) g1의 r과 c에 게이트1의 row, column값을 저장한다.
- 8) g2의 r과 c에 게이트2의 row, column값을 저장한다.
- 9) 게이트가 존재하는 시간을 11초에서 15초 사이에서 랜덤하게 정한다. (그 시간동안 sleep)
- 10) 해당 시간이 지난 후, 게이트1과 게이트2의 배열값을 prev_g1, prev_g2로 바꾸어준다.
- 11) 3초에서 7초간 동작을 멈춘다. (다음 게이트 쌍이 바로 나오지 않도록)
- 12) 3단계 ~ 11단계를 while loop 내부에서 실행

3. map_array.cpp : 맵 데이터 저장 (개발자 : 김서경)

4. map.h - map.cpp

• variables

- **int map_array[21][40]** : 21x40 크기로 된 맵배열
- **Window *gamewin** : game 화면을 보여주는 window

void map(): 게임 윈도우에 맵, 아이템, 게이트, 스네이크를 출력하는 함수

개발자: 이소영(50%), 이아연(50%)

각 요소들에 대한 COLOR_PAIR와 맵 배열에 저장되는 값 사이의 관계(맵 배열 값 + 1)를 이용해 색을 지정하고 게임 윈도우에 출력한다.

5. display.h - display.cpp : 서브 윈도우 출력 및 화면 효과 제어

##void fancy_lighting(int) : 게임오버, 클리어, 시작화면을 출력하는 함수

개발자: 김서경(100%)


3차원 배열으로 화면에 출력할 텍스트를 정수형으로 만든다.
반복문을 사용해서 각 정수 값마다 다른 유니코드를 출력한다.(이 때 usleep 함수를 사용하여 왼쪽 부터 출력되는 애니메이션을 연출한다.)

##void nextStageEffect(int) : 다음 스테이지 넘어갈 때 대기화면 출력함수

개발자: 김서경(100%)

화면을 완전히 클리어한 후, 5초동안 초단위로 남은 전환시간을 출력하여 대기시간을 알린다.

##void State_board() : stateboard를 출력하는 함수

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

개발자: 김서경(90%), 이소영(10%)

stateboard를 sub window로 생성하고
현재 스네이크의 정보(길이,아이템획득현황,게이트지나횟수,시간)을 state board에
출력한다.

##void Mission_board(mission,mision_result):미션 보드를 출력하는 함수

개발자: 김서경(90%), 이소영(10%)

missionboard를 sub window로 생성하고
미션 달성 현황을 mission board에 출력한다.

6. mission.h - mission.cpp (김서경 :100%)

\$struct mission_result : 미션 수행결과를 저장한 구조체

미션 수행 결과를 리턴하는 함수에서 여러개의 변수를 반환받기위해 만들었다.

#class Mission : 미션에서 요구하는 조건들을 저장할 클래스

variables:

- int level : 스테이지 레벨(레벨별 미션설정 위험)
- int leng : 스네이크 길이
- int gitem : growth item개수
- int pitem : poison item개수
- int gate : gate 지나간 횟수
- int runtime : 시간
- bool lock : poison 아이템 조건을 이하/이상으로 설정할 것 인지


##get[멤버변수]함수 : Mission클래스의 private변수들을 리턴하는 함수

##void set_mission() : class Mission에 저장된 level값에따라 미션조건 설정

switch-case 문을 사용하여 level값에 따라 mission class의 멤버변수를 변경한다.

##void isMissioncomplete(mission_result&,int,int,int,int,int)

if문을 사용해서 현재 미션의 조건을 달성했는지 각각 멤버변수 마다 확인하고,
참조한 mission_result 객체의 변수값을 변경하여준다.
(달성 시 1, 미달성 시 0)

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

7. main.cpp

• variables

- int growlItems : 현재 스테이지에서 획득한 growlItem의 수
- int posionItems : 현재 스테이지에서 획득한 poisonItem의 수
- int gates : 현재 스테이지에서 통과한 gate의 수
- int runtime : 현재 스테이지에서의 스네이크 생존 시간
- int snSize : 현재 스테이지에서의 스네이크 사이즈
- bool go : 스테이지별 흐름 제어를 위한 변수
- bool ramerun : 전체 게임 제어를 위한 변수
- int stageLevel : 스테이지 레벨 변수
- int nowMap : mapList에서의 현재 맵의 인덱스
- int maxR : 맵의 행 개수
- int maxC : 맵의 열 개수
- position g1 : gate1의 위치(row, column)값을 저장하는 position 구조체
- position g2 : gate2의 위치(row, column)값을 저장하는 position 구조체
- extern int mpList[8][21][40] :
- Window *state_board : 현재 stage의 snake 상태를 보여주는 윈도우
- Window *mission_board : 현재 스테이지의 미션 수행 상태를 보여주는 윈도우

void checkTime(): 현재 스테이지에서의 스네이크 생존 시간을 측정하는 함수


개발자: 이소영(100%)

go 변수를 이용해 스테이지가 진행되는 동안 스네이크의 생존 시간을 측정한다.

void run(Snake &s) : 스테이지별 게임 진행에 필요한 모든 기능을 수행하는 함수

개발자: 이소영(40%), 이아연(30%), 김서경(30%)

- 1) 초기화: 미션을 초기화하고 스네이크 이동에 이용할 변수를 초기화한다.
- 2) 스네이크 이동 제어: 0.5초마다 키 입력을 받아 스네이크가 이동하게 한다.
- 3) 스네이크 이동에 따른 효과 발생: 스네이크 클래스 멤버 함수를 호출해 이동한 좌표에 따라 효과가 발생하도록 한다.
- 4) 게임 오버 조건 확인: 이동에 따라 게임 오버 조건에 해당하는 지 확인하고 해당한다면 gamerun, go 변수 값을 false로 바꿔준다. 윈도우의 오른쪽아래에 로딩메세지를 출력한다.
- 5) 미션 성공 조건 확인: if문으로 모든 미션조건을 달성했는지 확인하고 해당 반복문을 빠져나가도록 설정한다. stageLevel++한다. 윈도우의 오른쪽아래에 로딩메세지를 출력한다.
- 6) 화면 출력: 각각의 서브 윈도우에 해당 내용을 반영하여 출력한다.

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26


int main() : 전체 게임의 실행과 종료, 스테이지 전환을 담당하는 함수

개발자 : 이아연(50%), 김서경(40%), 이소영(10%)

- 1) **ncurses 셋업**: UTF-8을 사용할 수 있는 환경 설정 및 메인 윈도우 실행, COLOR_PAIR 정해주기
- 2) **서브윈도우 설정**: 게임 윈도우, 스테이트 윈도우, 미션 윈도우 위치 설정
- 3) **프로그램 시작**: 시작화면 출력 및 stageLevel = 1로 초기화
- 4) **게임 실행**:
 - i. gamerun == true && stageLevel <= 4일동안 while loop을 돌려준다.
 - ii. go == true로 설정
 - iii. 8가지의 맵 중 하나를 랜덤으로 결정하고 맵 배열을 초기화한다.
 - iv. 현재 상태를 초기화한다.
 - v. 아이템 함수와 게이트 함수를 thread를 사용하여 독립적, 동시 실행시키고 run함수를 실행시킨다.
 - vi. 스테이지가 종료되면(go == false) 모든 thread를 종료한다.
 - vii. 게임 오버 조건(gamerun == false, stageLevel <= 4)과 게임 성공 조건(stageLevel >= 5)이 달성될 경우, 해당 화면을 출력한다.

2.2.3 활용/개발된 기술

- **사용한 라이브러리**:
 - random, unistd.h: 아이템 생성시간, 삭제시간, 생성위치, 스테이지별 맵 랜덤 선택을 난수를 사용하여 제어하기 위해 사용하였다.
 - thread: 스네이크 이동과 아이템 발생을 각각 독립적으로 동시에 진행하기 위해 사용하였다.
 - locale.h: 유니코드와 한글 사용을 위해 사용하였다.
- **외부 라이브러리 : <ncurses>**
 ncurses는 curses라이브러리의 새로운 버전이다.(new curses) curses라이브러리는 텍스트 모드에서 윈도우, 메뉴, 마우스, 색상등을 쉽게 사용할 수있도록 도와주는 라이브러리이다. ncurses라이브러리도 curses와 거의 같은 기능을 수행한다.
- **사용한 자료구조**:
single linked list : 스네이크의 몸통 구현을 SLL(single linked list)를 사용하여 구현 하였다.
 - SLL의 head노드는 스네이크의 헤드로 설정하여 바디들을 링크하였다.
 - 스네이크의 이동은 스네이크의 맨앞에 이동할 좌표의 노드를 생성한뒤 head노드 앞에 삽입하고, 맨뒤 노드를 delete하는 방식으로 구현하였다.
 - addbody(스네이크 바디추가)기능은 SLL의 맨뒤 노드뒤에 새로운 노드를 추가하는 방식으로 구현하였다.
 - removebody(스네이크 바디삭제)기능은 SLL의 맨뒤 노드를 하나 삭제하는 방식으로 구현 하였다.

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

2.2.4 현실적 제한 요소 및 그 해결 방안

제한요소


- 1) 아이템 생성과 게이트 생성, snake이동 등의 기능에서 시간제어 기능을 구현하기 위해, 시간 지연 함수를 사용했을 때 모든 프로그램이 중지되어 프로그램이 원활히 실행되지 않았다.
- 2) 방향키 입력에 conio.h 라이브러리를 이용하려 했으나 C++ 표준 라이브러리가 아니어서 사용할 수 없었다. getch() 함수의 경우 키가 입력될 때까지 프로그램이 대기하여 게임 진행과 키입력을 병렬적으로 진행할 수 없었다.
- 3) 스네이크 게임 진행 시간 측정 및 스네이크 이동 tick 측정을 위해 time.h 라이브러리의 time(), clock() 함수를 이용했을 때 게임 진행 시 스네이크의 이동 속도가 일정하게 유지되지 않았다.

해결방법


- 1) c++ 표준 라이브러리 < thread >를 include하여, 여러개의 작업을 병렬로 처리하도록 설계하였다. 따라서 스레드 함수내부의 sleep함수가 프로그램 전체에 영향을 끼치지 않게되어 해당 제한요소를 해결할 수 있었다.
- 2) ncurses의 nodelay() 함수를 이용하여 getch() 함수 이용 시 프로그램이 대기하지 않도록 설계하였다. 따라서 게임의 진행과 키 입력이 병렬적으로 이루어져 해당 제한요소를 해결할 수 있었다.
- 3) OS 독립적으로 시간을 측정할 수 있는 chrono 라이브러리를 이용하여 tick의 측정에 절대적인 값을 이용하였다. 따라서 게임 진행 시 스네이크의 이동 속도가 일정하게 유지되도록 하여 해당 제한 요소를 해결할 수 있었다.

2.2.5 결과물 목록

소스파일	기능
main.cpp	게임 실행 소스파일
Snake.cpp, Snake.h	snake 관련 기능(이동,길이조정 등)을 구현한 소스파일
Items.cpp, Items.h	아이템 및 게이트 생성, 소멸 기능을 구현한 소스파일

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

display.cpp, display.h	게임시작, 실패, 성공 화면 출력함수 mission board, score board 출력 함수
mission.cpp, mission.h	스테이지별 미션 정보 저장 및 미션 성공 조건 확인
map.cpp, map.h	게임 윈도우(sub window)를 업데이트 하는 소스파일
map_array.cpp	맵데이터를 저장하고 있는 소스파일
position.h	position 구조체를 구현한 헤더파일

 국민대학교 컴퓨터공학부 c++ 프로그래밍		결과보고서	
		프로젝트 명	Snake Game
		팀 명	A++ 프로그래밍
		Confidential Restricted	Version 1.0 2020-JUN-26

3 자기평가

김서경: 스네이크 클래스의 일부 함수 구현 및 서버윈도우 구현, 전체 미션기능 구현, 게임의 그래픽 업그레이드 등에 참여하였다. 생소한 주제가 아닌, 게임이라는 주제가 주어졌기 때문에 더욱 흥미를 가지고 개발했다. 또한 어떻게 하면 더 재밌게, 예쁘게 만들 수 있을지 계속 생각하게되어 게임의 전체적인 퀄리티를 높이는 것에 도움을 준것 같다. 전체적으로 이번 프로젝트에 열심히 참여했다. 프로젝트를 처음 수행할때에는, ncurses 라이브러리에 대해 아예 무지하여 처음부터 공부해야하는 과정이 어려웠다. 그러나 팀원들과 만나서 서로 공부한 내용을 공유하며 점점 ncurses 라이브러리를 사용한 프로그램 구현에 적응 할 수 있었다. 또한 본격적으로 github를 사용하여 협업프로젝트를 하게된 것은 처음이었다. 기존에는 혼자서 작업하는 것을 선호했었는데, 이번 기회를 통해서 협업은 어떻게 하는 것인지 알아가는데 도움이 되었다. 앞으로도 실무에서, 학교에서 협업해야할 상황들이 많이 생길 것이 분명한데, 본 프로젝트 경험을 통해 잘 해나갈 수 있을 것이다.

이소영: 이번 프로젝트에서 스네이크와 게이트 기능을 위주로 담당했다. 메이크 파일 작성 및 전체 코드 분리, 전역 변수 관리를 담당했다. 전체적으로는 코드의 불필요한 중복이나 조건문을 간소화할 수 있도록 자잘하게 수정했다.


이번 프로젝트 진행 시 기능별로 파일을 분리할 때 전역 변수 관리가 어려웠다. extern에 대해서 수업시간에 배울 땐 이해했다고 생각했는데 실제 사용을 위해 찾아보며 내가 생각했던 개념과는 잘못 이해했음을 깨달았다. 더불어 전역 변수 관리를 위해 정보를 찾아보며 C++의 컴파일 및 링킹과정에 대해 더 이해하게 되었다.

프로젝트를 진행하며 스네이크 저장에 자료 구조 시간에 배운 single linked list를 이용함으로써 이동을 간단히 구현한 점과 이동 방향 좌표 계산을 위해 offset이란 배열을 만들고 이용했던 점을 잘 한 것 같다. 다른 교과목 시간에 배운 개념을 연계하여 활용해볼 수 있어서 도움이 되었다. 또 게이트를 통과 후 방향을 설정할 때 어떻게 해야하나 고민이 많았는데 이전 방향과 우선순위에 따른 통과 후 방향 사이의 관계를 수식으로 구현하여 간소화한 것 같아 뿌듯하다. 전체적으로 수업을 들을 때는 모호했던 개념들이 프로젝트를 진행하며 직접 활용해보고 찾아보며 더 잘 정립된 것 같다. 코드 관리에 있어서 깃허브를 이용했는데 처음으로 브랜치를 이용해봤다. 브랜치의 개념에 맞게 잘 활용한 것 같진 않지만 이번 경험을 토대로 앞으로는 더 잘 활용할 수 있을 것 같다.

이아연: 스네이크 아이템과 게이트의 출현 및 생성, thread이용을 주로 담당했다. 또한 코드의 통합을 도우며 바뀌는 함수와 변수의 사용법에도 지속적으로 관여하였다. 프로젝트 초반에는 굉장히 자신감이 없어서 팀의 다른 친구들에게 도움이 될 수 있을지 고민이 많았다. 특히, 만나지를 못해서 철저하게 분업화되어서 프로젝트를 만들어나갔는데 지금까지는 항상 모여서만 프로젝트를 진행했던지라 내가 맡은 부분을 제대로 수행할 수 있을지에 대한 걱정이 많았다. 하지만 프로젝트를 진행하면서, 스스로 맡은 부분에 대해서는 책임감을 가지고 모르면 알 수 있을 때까지 심혈을 기울이게 되었고, 실제로 해낼 수 있었다. 초반보다 자신도 하면 할 수 있다는 자신감을 갖게 되었다. 또한 분업화된 시스템이 각자의 시간의 절약될 뿐 아니라 보다 전문적인 수행이 가능하다고 느껴졌다.

코드를 작성할 때마다 새로운 방식을 시도하기를 꺼려왔다. 새로운 라이브러리, 새로운 코딩방식과 메서드의 사용등에도 새로운 것을 알아가는데 힘들어서 항상 부정적으로 바라보았는데, 이 프로젝트를 진행하면서 thread를 사용하고 ncurses를 사용하는 등, 스스로 다양한 시도를 해보게 되었고 심지어 thread의 이용을 담당하는 상황까지 오게 되었다. 해당 프로젝트를 통해서 새로운 기술의 사용에도 굉장히 호의적으로 변하고 다양한 기술의 활용이 반드시 필요한 일임을 느끼게 되었다.

프로젝트를 진행하면서 전체적으로 느낀 것은, 분업적으로 일을 진행하고 통합하는 과정과 새로운 기능의 사용에 굉장히 긍정적인 관점을 갖게 되었고 팀 협업의 중요성을 알게 되었다.

 국민대학교 컴퓨터공학부 c++ 프로그래밍	결과보고서		
	프로젝트 명	Snake Game	
	팀 명	A++ 프로그래밍	
	Confidential Restricted	Version 1.0	2020-JUN-26

또한 이런 과정을 통해서 스스로에게 자신감을 갖을 수 있던 시간이었다.

4 참고 문헌

번호	종류	제목	출처	발행년도	저자	기타
1	웹사이트	http://hughm.cs.ukzn.ac.za/~murrellh/os/notes/ncurses.html#input			X. Li	
	기사					

5 부록

5.1 사용자 매뉴얼

1. ./Snake 명령어로 프로그램을 실행하면 게임 타이틀이 나타난다, 이 때 사용자는 아무 키나 눌러 게임을 시작할 수 있다.
2. 키보드 방향키 ↑,←,→,↓ 을 눌러서 snake를 0.5초에 한번 씩 이동 시킬 수 있다.
3. 초록색 칸은 growth item이다. 획득할 시 snake의 길이가 1늘어난다.
4. 빨간색 칸은 poison item이다. 획득할 시 snake의 길이가 1줄어든다. 길이가 3 이하가 되면 게임오버이니 주의하자.
5. 보라색 칸은 gate이다. 보라색칸에 snake의 head가 접촉시 다른 gate로 이동된다.
6. 미션보드에 있는 미션들을 모두 완료하면 다음 스테이지로 넘어갈 수 있다.
7. 스테이지 클리어, 게임오버가 되면 로딩메세지가 나타난다. 사용자는 이때 키입력없이 화면전환을 기다려야한다.
8. 4스테이지를 클리어하면 게임클리어 화면을 볼 수있다.

5.2 설치 방법

소스파일이 있는 폴더위치에서 make Snake 명령어를 사용하여 컴파일 한다.

./Snake 로 실행한다.