

SNAKE-19



과 목 명	게임수학 [화 7-9]
지도교수	최영미 교수님
팀 명	코로나싫조
학번 & 성명	20191024 조민지 (조장)
	20181006 윤홍비
	20190968 김은빈
	20191006 이재이
제 출 일	2020. 11. 16 (월)

목 차

1.	프로젝트 소개	3
	(가) 프로젝트 요약	
	(나) 프로젝트 팀 구성원	
	(다) 프로젝트 개요	
2.	연구 목적 및 필요성	5
3.	연구배경	6
	(가) 벤치마킹	
	(나) 기존게임 소스코드 분석	
	(다) 기존게임 함수계층도 분석	
4.	연구 범위 및 방법	13
	(가) 개작 게임 설명	
	(나) 실행 화면	
5.	개작 게임 분석 및 함수 계층도	15
6.	개작 전과 후의 비교	26
7.	연구 결과에 대한 기대효과	27
8.	연구 추진일정	28
9.	참고문헌	29

1. 프로젝트 소개

(가) 프로젝트 요약

1) 프로젝트명	스네이크 게임
2) 팀 명	코로나싫조
3) 결과물 유형	Win32 콘솔 응용 프로그램 실행파일(.exe) 콘솔 화면을 이용한 게임
4) 제작기간	2020.09. 07 ~ 2020. 11 . 10 (약 10주)
5) 프로젝트 주요내용	<p>저희 코로나싫조는 이번 게임수학 과목을 통해 기존의 C언어 기반 콘솔 창에 구현된 스네이크 게임을 분석하고 부족한 점을 채워 넣어 좀 더 완성도 높은 게임을 제작하고자 합니다. 스토리를 넣어 재미를 더하고 인터페이스를 구상해 저희 조만의 업그레이드 된 스네이크 게임을 제작하는 것이 목표입니다.</p> <p>스토리 구상은 저희 코로나싫조에 걸맞게 먼 훗날 코로나19가 심각해져 밖으로 쉽게 나가지 못하는 상황에 집에 식량이 다 떨어져 바이러스를 피해 식량을 구하러 가는 것이 목표입니다.</p> <p>게임방식은 스네이크 게임방식을 따라서 방향키를 통해 맵 곳곳에 있는 식량을 구해 게임을 진행하는 방식입니다.</p> <p>인터페이스 구상은 게임 시작 시 스토리라인을 설명할 것이며 인게임에서는 맵 안에서의 식량을 랜덤으로 배치하고, 식량을 먹을 시 꼬리 부분이 하나씩 늘어나는 모습을 구상했습니다. 벽에 닿아 실패 시 게임오버화면을 띄우고 다음화면으로 코로나 예방 수칙을 띄워 이번 코로나 수칙을 상기시키도록 구상했습니다.</p>

(나) 프로젝트 팀 구성원

사진	학번	성명	연락처	이메일	주요업무
	20191024	조민지	010-9184-2877	minjicho2008@naver.com	전체적인 총괄 제안서 작성 아이디어 제안 발표 코딩
	20181006	윤홍비	010-8506-7190	dbsghdql555@naver.com	코딩 아이디어 제안 ppt 제작 제안서 작성
	20190968	김은빈	010-5913-0619	eunbinyi@naver.com	자료수집 코딩 제안서 작성 아이디어 제안 ppt 제작
	20191006	이재이	010-3023-5138	wodl0416@gmail.com	자료수집 코딩 제안서 작성 아이디어 제안 발표

(다) 프로젝트 개요

저희 조 이름에 맞게 짠 스토리에 따라서 게임이 진행됩니다. 방향 키로 이동 방향을 조정하며 식량을 구할수록 꼬리가 늘어납니다. 움직이다가 벽에 닿게 되면 실패합니다. 실패하면 코로나 예방 수칙과 함께 창이 떠서 교육적인 부분도 있습니다.

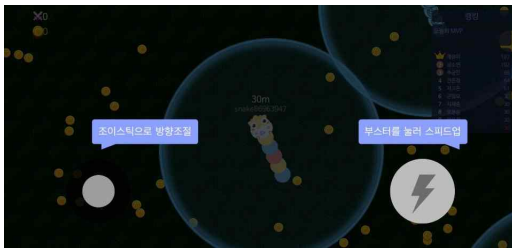
2. 연구 목적 및 필요성

- 게임 제작에 필요한 수학과 물리 응용 능력을 키우기 위함.
- 협업 능력을 키우기 위함.
- 해당 게임 연구를 통해 게임의 필요한 기초적인 게임 소스 코드를 읽고 이해할 수 있는 능력 향상을 위함.
- 언어를 기초로 하는 프로젝트이므로 C언어 코딩 능력의 향상을 기대할 수 있다.

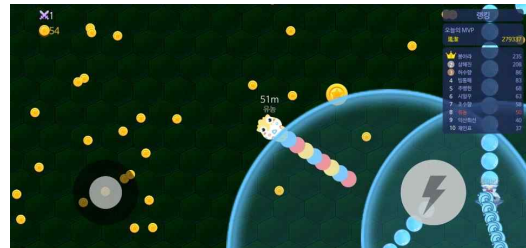
3. 연구 배경

(가) 벤치마킹

사례 1 : 머지 스네이크



< 게임 설명 화면 >



< 게임 진행 화면 >

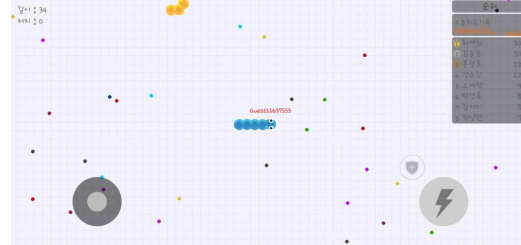
머지 스네이크는 조이스틱과 부스터를 이용해서 코인을 먹으면서 상대방인 적을 피해 다니는 식으로 진행되는 게임이다. 부스터를 사용해서 빠른 속도로 이동이 가능하고 제한 시간이 없기 때문에 코인을 더 많이 먹고 적이나 벽과 부딪히지 않으면 게임을 계속 진행 할 수 있습니다. 그리고 일반적인 뱀 모양이 아니라 치즈를 달고 다니는 햄스터, 닌자 등 다양한 캐릭터가 있습니다.

장점	일반적인 스네이크 모양이 아닌 다양한 캐릭터가 있어서 시각적 즐거움이 있음.
단점	광고가 너무 자주 뜨며, 게임을 진행할 때 나오는 목소리가 거슬림.
공통점	저희 게임과의 공통점은 캐릭터가 무언가를 먹으면서 게임이 진행된다는 점과 제한 시간이 업삼는 점이 있습니다.
차이점	머지 스네이크는 부스터라는 아이템을 활용할 수 있지만 저희는 그런 아이템은 만들지 않았습니다.
개선사항 및 적용할 것	머지 스네이크에 단순히 코인 먹기, 상대 피하기만 있는 것이 아닌 다양한 미션이 있는 스테이지가 있으면 더 좋을 것 같습니다. 저희가 머지 스네이크에서 가져와 저희 게임에 적용한 것은 바로 게임이 시작하고 게임에 대한 설명 방법이 나오는 것입니다.

사례 2 : 스네이크 오프



< 게임 시작 화면 >



< 게임 진행 화면 >

스네이크 오프는 터치한 곳으로 뱀이 이동해서 조작법이 편한 게임이다. 다른 게임과는 다르게 다양한 모드가 없고 단 2가지 모드만 있기 때문에 간단하고 누구나 할 수 있는 게임이다.

장점	게임방식이 간단하고 모드가 2가지 밖에 없어서 게임을 처음 하는 사람도 쉽게 할 수 있다.
단점	게임 방식이 너무 간단해서 지루하고 오래 게임을 할 수는 없을 것 같다.
공통점	공통점은 모드의 종류가 별로 없어 간단한 게임이라는 점입니다.
차이점	터치를 해서 뱀을 이동시킬 수 있는 것이 차이점입니다.
개선사항 및 적용할 것	모드가 다양하지 않기 때문에 지루할 수 있어 유저들끼리 경쟁하는 그런 다른 방식의 새로운 모드를 만들 필요가 있다. 스네이크 오프에서는 모드가 적은 부분을 가져와서 저희 게임을 만들 때도 유저들이 쉽게 접할 수 있도록 모드의 종류를 최소화 해서 적용하였습니다.

(나). 기존게임 소스코드 분석

(1) gotoxy() 함수

- gotoxy() 함수는 커서의 위치 제어 함수이다.

라이브러리 함수가 아니므로 Windows.h 헤더 파일을 입력해야 사용가능 하다.

gotoxy	함수 원형	void gotoxy (int x, int y)	
	함수 인자	int x	화면에서의 가로 위치를 지정 (1~80)
		int y	화면에서의 세로 위치를 지정 (1~24)

- 기본적인 gotoxy() 함수의 형태

```
void gotoxy(int x, int y)
{
    COORD Pos;
    Pos.X = x;
    Pos.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}
```

- SetConsoleCursorPosition() 함수는 지정된 콘솔 화면 버퍼에서 커서 위치를 설정하며, 이 함수의 호출은 필수사항이다.
- GetStdHandle() 함수는 실제 핸들을 반환하는 함수이며 그 안에 파라미터로 종류를 정해주면 핸들 값을 반환한다. 인자로 STD_OUTPUT_HANDLE를 주면 표준 콘솔 출력의 핸들을 반환해준다.

(2) 키보드 플레이어 구문

- 방향키 입력제어 kbhit()과 getch() 함수

- 보통 키보드로 입력을 받을 때는 scanf 함수를 사용하지만, 게임의 조작키를 입력 받을 경우에는 kbhit()과 getch()를 사용한다.
- kbhit() 함수는 현재 키보드가 입력된 상태인지를 조사한다. 즉, 눌렀느냐 안 눌렀느냐를 알아볼 수 있는 함수이다. 또한 kbhit() 함수는 키의 입력이 없으면 생략한다.
- kbhit() 함수의 장점은 waiting 함수가 아니라는 점이다. kbhit() 함수는 입력이 있든 없든 바로 반환하기 때문에 반환 값에 따라 작업을 진행할 수 있다.

함수 원형	int kbhit (void)
변환값(return)	키 입력 여부, 입력이 있으면 1, 없으면 0

- getch() 함수는 키보드의 입력을 받는 함수로, scanf()와 다른 점은 입력한 키보드의 내용이 화면에 보이지 않는다. 그리고 Enter key를 통해 입력을 확인하지 않고 키를 누른 순간 입력 버퍼에 값이 들어가게 된다.

함수 원형	int getch (void)
-------	------------------

변환값(return)

입력된 key의 아스키코드 값

- 기존게임 코드를 보면, if문과 kbit()을 이용해 키보드 입력을 감지하고, getch()로 아스키코드 값을 가져온 후, switch ~ case를 이용해 동작을 정의했다.
- kbhit(), getch()은 <conio.h>에 정의되어 있으므로 <conio.h>를 include 한다.

(3) move_snake() 함수

뱀 캐릭터의 움직임과 위치를 설정하는 함수 move_snake()는 UP, LEFT, RIGHT, DOWN 네 개의 방향키에 따라 뱀의 x, y 좌표를 어떻게 변화시킬 것이며, 장애물과 충돌했을 때의 위치는 어떻게 설정할 것인지에 대한 내용을 다루고 있다.

- 사용된 함수들

- strike_check() : 상황별 반응 설정 함수
- gotoxy() : 커서 위치 제어 함수
- fflush() : 파일 스트림 버퍼를 지우는 함수
- SetColor() : 콘솔 내 색깔 설정
- free() : 동적으로 할당한 메모리를 해제하는 함수

- switch(direction)-case 구문

switch(direction)	
case 1: tmp_y--;	y 좌표 -1
case 2: tmp_x--;	x 좌표 -1
case 3: tmp_x++;	x 좌표 +1
case 4: tmp_y++;	y 좌표 +1

-> 어떤 방향키를 누르는지에 따라 뱀 머리의 위치를 나타내는 x, y 좌표가 바뀌게 되는데, 이를 swith-case 구문으로 표현했다. case 별로 각각 case 1은 위쪽 방향키, case 2는 왼쪽 방향키, case 3은 오른쪽 방향키, case 4는 아래쪽 방향키를 눌렀을 때의 변화를 나타낸 것이다.

- 캐릭터 모양 출력 및 설정

```
SetColor(GRAY);
gotoxy((snake_head->x * 2) + abs_x, snake_head->y + abs_y);
printf("○");
gotoxy(tmp_x * 2 + abs_x, tmp_y + abs_y);
```

```
printf("●");
gotoxy(77, 23);
SetColor(BLACK);

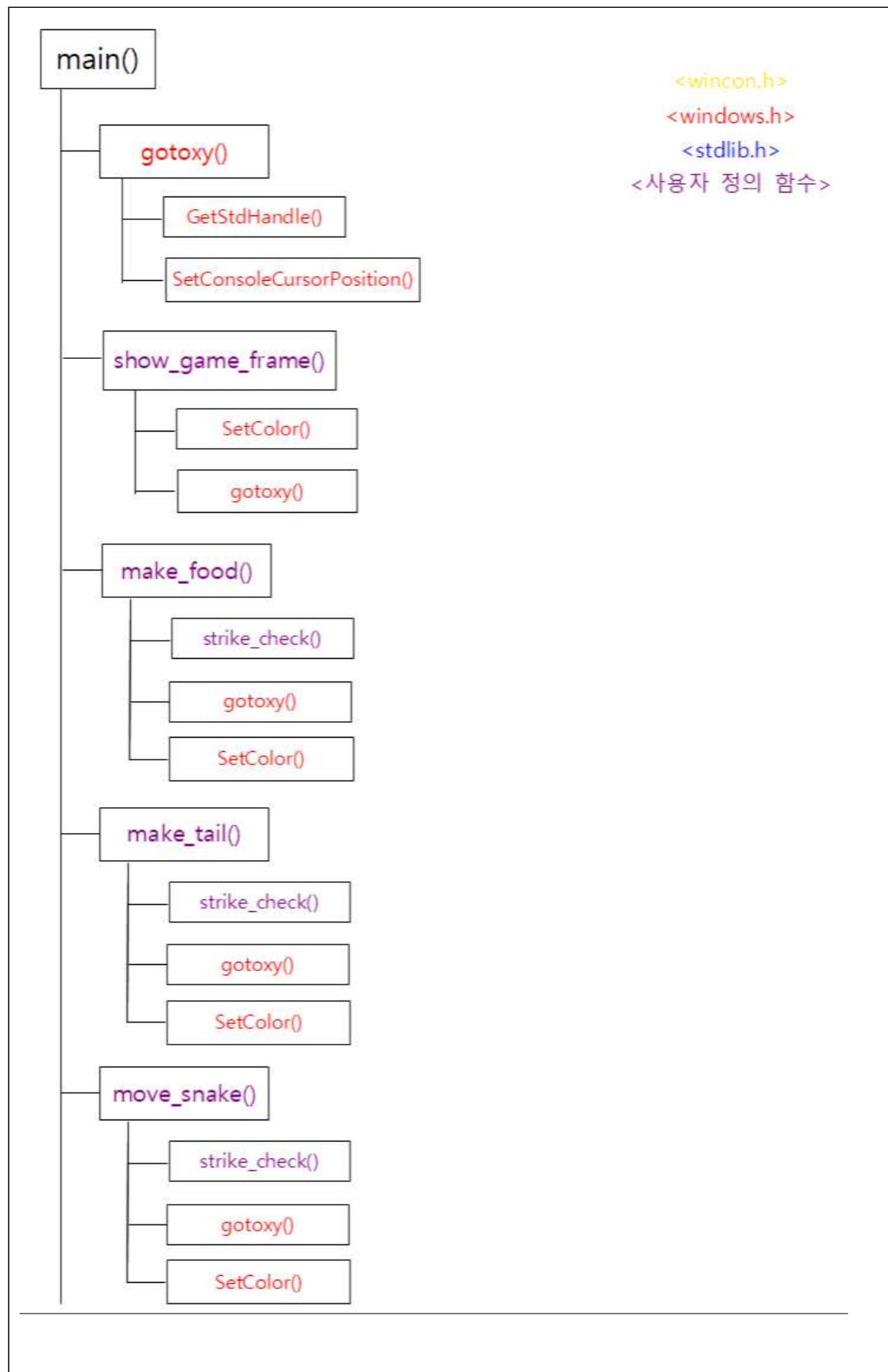
tmp_body = (struct BODY*)malloc(sizeof(struct BODY));
tmp_body->x = tmp_x;
tmp_body->y = tmp_y;
tmp_body->next = snake_head;
snake_head = tmp_body;

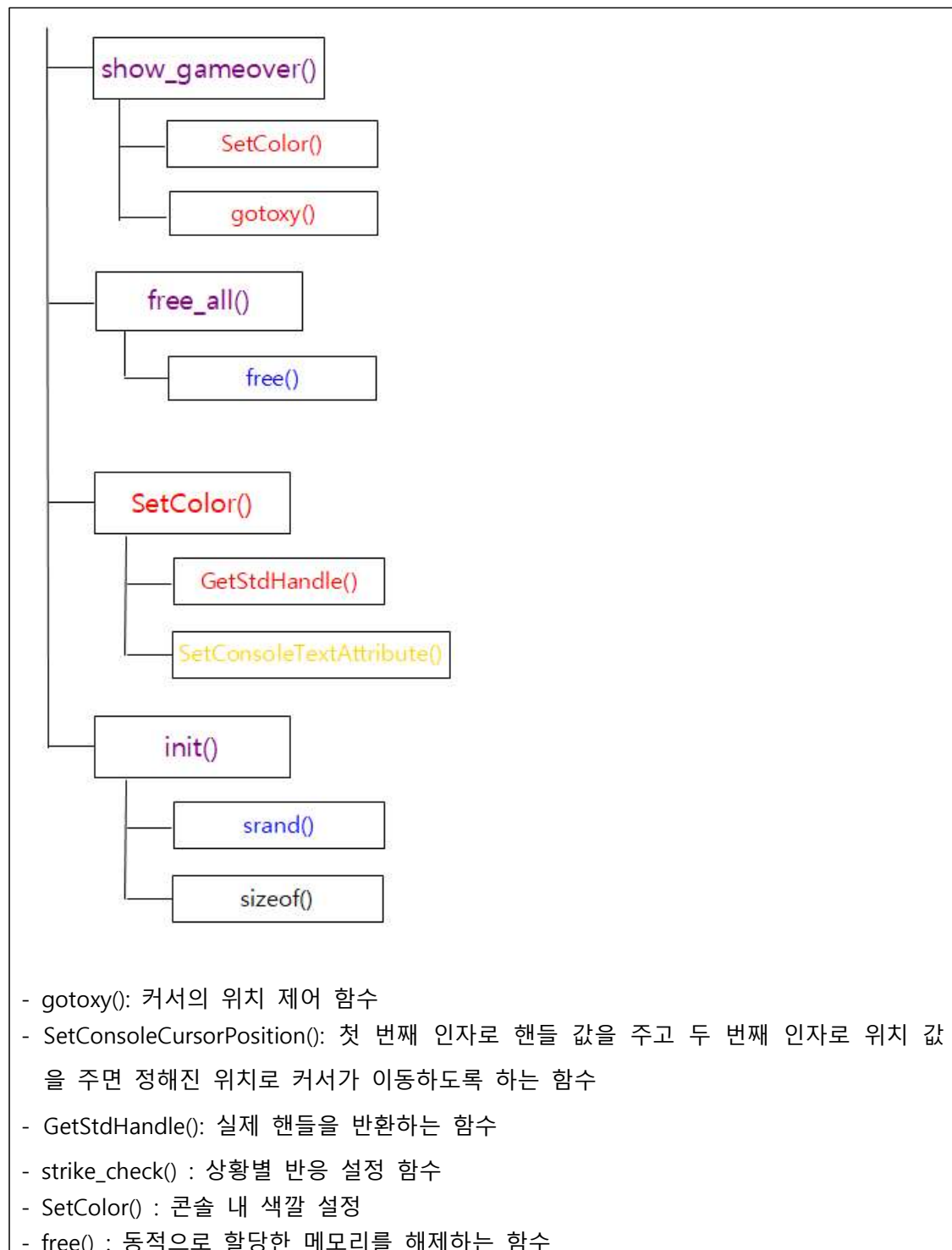
while (tmp_body->next != snake_tail) //꼬리노드 지우기
    tmp_body = tmp_body->next;

tmp_body->next = NULL;
free(snake_tail);
snake_tail = tmp_body;
return i;
```

-> 출력할 캐릭터(뱀)의 머리와 몸체 부분을 표현한 부분이다. 이동하거나 방향키를 눌러 방향을 바꿨을 때, 캐릭터의 모양을 어떻게 설정할 것인지에 대해 나타내고 있다.

(다). 기존게임 함수계층도 분석





4. 연구 범위 및 방법

(가) 개작 게임 설명

- 스토리

코로나19 바이러스가 창궐하는 시대! 제한 시간 안에 바이러스를 피해 식량을 확보하라.

- 게임 방식

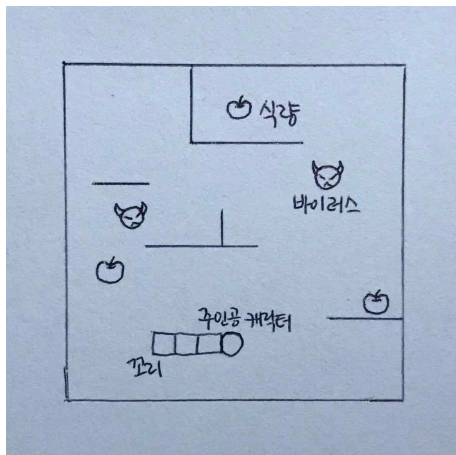
주인공 뱀은 다른 방향키를 누르기까지 현재 설정된 방향으로 계속 이동한다.

주인공 뱀은 먹이를 먹으면 꼬리가 하나씩 늘어나지만 벽에 닿게 되면 게임오버가 된다.

- 제한 사항

게임 시작과 동시에 먹이를 먹으면서 꼬리를 늘려나갈 수 있다. 하지만 벽(바이러스)에 닿을 시 게임은 종료된다.

- 기획 화면

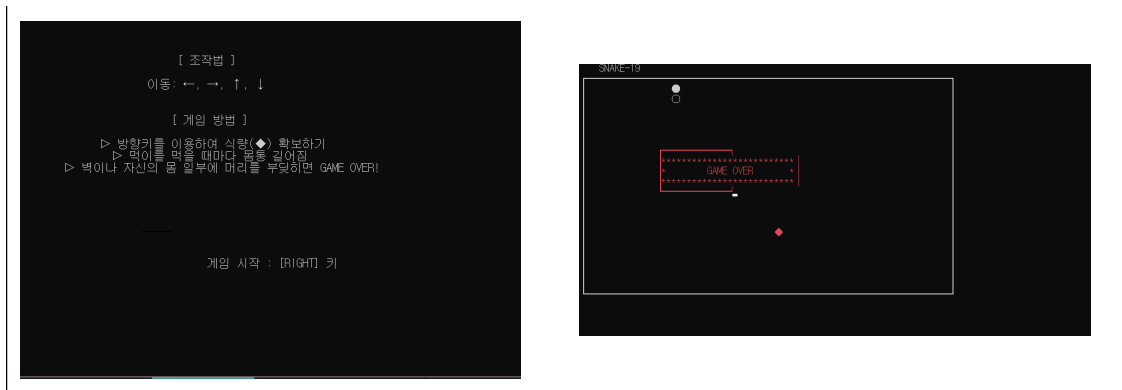


- 팀에서 진행한 개작 내용을 기술하시오. (벤치마킹에서 개선하려고 기술한 기능을 어떻게 구현하였는지 등)

원래 교수님이 주신 코드에서 저희가 처음에 기획했던 각종 게임 화면들을 만들어내기 위해서 코드를 추가했습니다. void game_explain()이라는 함수를 만들어서 여기에서 게임 설명하는 화면이 나오도록 구현했습니다. 그리고 int MainMenu라는 함수를 만들어서 게임을 시작하려고 클릭하면 나오는 제일 첫 번째 화면에서 저희 게임 이름이 나오고 게임을 그냥 시작할 것인지 아니면 게임 방법에 대한 설명을 들을 것인지 선택할 수 있습니다. 게임 방법을 선택하면 먼저 게임을 설명하는 짧은 문장들이 나오고 주의사항도 말해줍니다. 그 다음에는 조작법이 나오고 어떻게 게임을 플레이 할 수 있는지 알려줍니다.

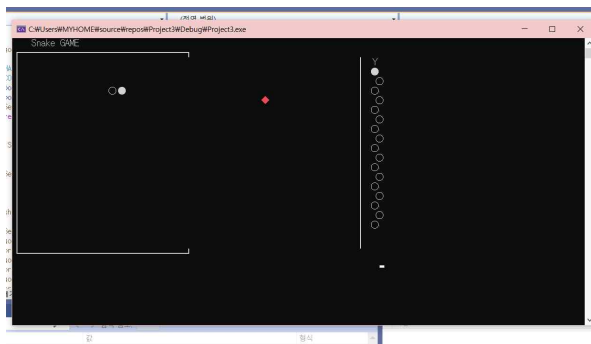
(나) 실행 화면

게임 첫 실행 화면	준비 화면
 <p>1. 게임 시작 2. 게임 방법</p> <p>> 메뉴를 선택하세요 [_]</p>	 <p>게임 설명 최대한 많은 이빨 방향을 조정하고 식량을 확보하면 성공!</p> <p>- 주의 - 자기 몸에 닿거나 바이러스 안전 구역 밖으로 나갈 경우 게임은 종료됩니다.</p> <p>다음 : [RIGHT] 키</p>
게임 실행 화면 (게임판)	스토리 화면
	 <p>게임 스토리 면역 증강 코로나 바이러스가 심각해져 결박에 나가지 못하는 상황 하지만 면역은 식량이 없어져 면역 증강을 구해오자! 코로나 바이러스를 피해 마트에서 식량을 구해오자!</p> <p>다음 : [RIGHT] 키</p>
결과창	
 <p>코로나 예방 수칙 1. 사교와 거리 두기 지키기 2. 마스크 착용하기 3. 손 자주 씻기</p> <p>다시 시작 : [RIGHT] 키</p>	
도움말 화면	종료 화면



5. 개작게임 분석 및 함수 계층도

SetColor함수에 static HANDLE std_output_handle=GetStdHandle(STD_OUTPUT_HANDLE); 이 줄이 예러가 납니다. visual studio 2019에서 오류가 떠서 창을 보면서 몇 번째 줄에 어떤 게 오류가 나는지 알 수 있었다. 오류는 상수 식에 함수 호출을 사용할 수 없다고 나옵니다. 이것은 공부해본 결과 static이라는 정적 변수 자료형에 GetStdHandle이라는 함수를 사용해서 상수의 값을 변경할 수 없는데 함수를 통해서 바꾸려고 하니 오류가 나는 것 같았습니다. HANDLE이라는 것은 찾아보니 콘솔 프로세스는 핸들을 사용하여 콘솔의 입력 및 화면 버퍼에 액세스한다고 합니다. 그래서 콘솔 창에서 사용하는 색상 변경 코드를 찾아보았더니 스네이크 게임에 들어있는 색상 코드 그대로 SetConsoleTextAttribute라는 함수가 있는데 거기에서는 괄호 안에 두 가지 매개변수가 있었는데 첫 번째 매개변수가 앞에서 정의한 GetStdHandle과 관련된 함수의 이름을 그대로 써서 호출하는 것을 보았습니다. 그래서 정적 변수에다가 함수를 정의하는 것보다는 SetConsoleTextAttribute라는 함수의 첫 번째 매개변수로 GetStdHandle(STD_OUTPUT_HANDLE)을 넣어서 함수 자체를 불러오는 것을 어떻게 생각을 해보고 수정한 후 실행해 보았더니 결과는 성공적이었고 스네이크 게임은 실행되었습니다.



▲ 스네이크 게임 실행 화면

그래서

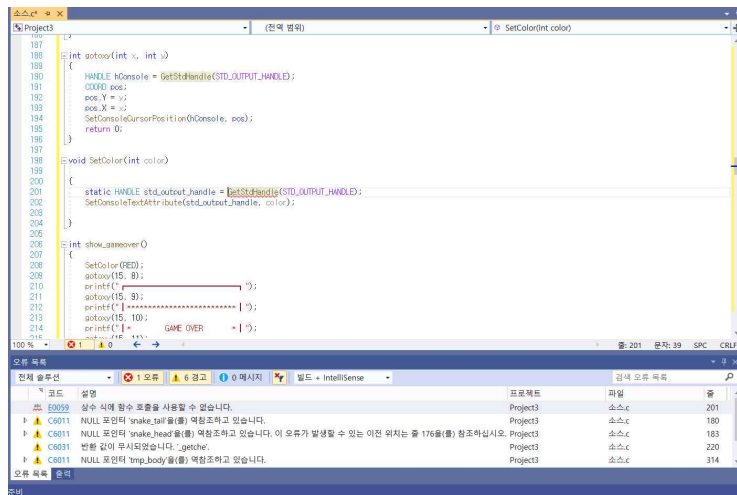
SetColor

함수의

내용이

전에는

[2020년도 게임수학 Team Project 최종보고서]



◀이렇게 되어 있었지만

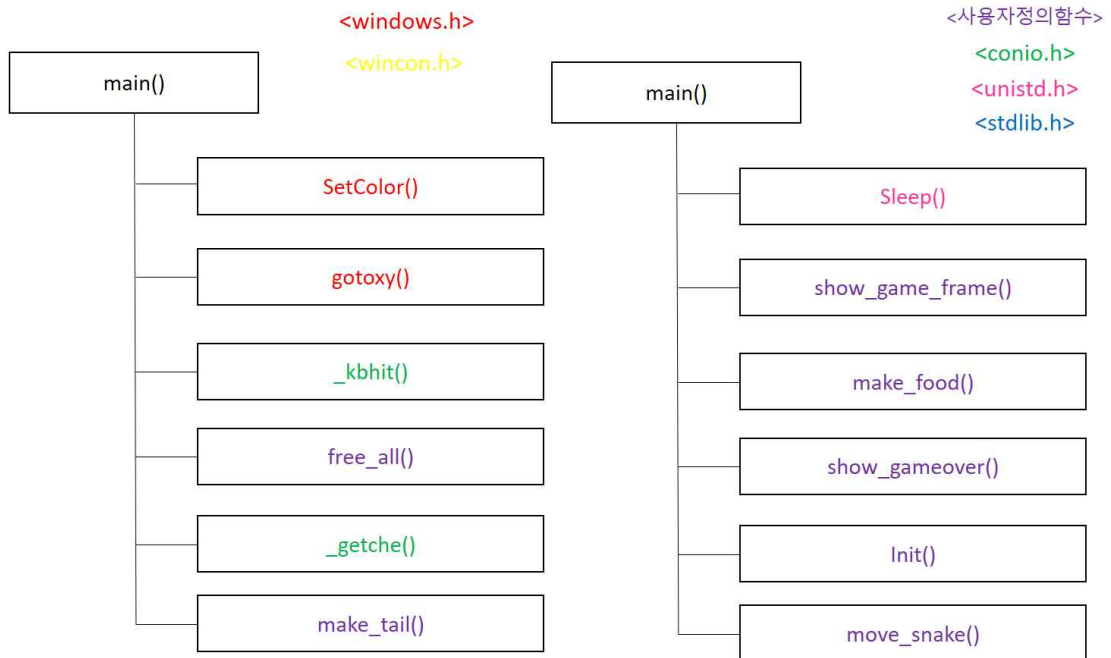
나중에는 이렇게 ▶

```
197
198 void SetColor(int color)
199 {
200     SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), color);
201 }
202
203
204
```

바꾸었습니다.

소스코드를 분석하면서 함수들의 호출 구도를 알 수 있었고, 이를 통해 그림으로 도식화 할 수 있었습니다. 우선 **검정색 상자**는 C 기본 함수를 의미하고, **초록색 상자**는 <conio.h> 헤더를 사용한 함수이고, **파란색 상자**는 <stdlib.h> 헤더를 사용한 함수이고, **핑크색 상자**는 <unistd.h>헤더를 사용한 함수이며, **보라색 함수**는 사용자 정의 함수이다. **빨간색 상자**는 <windows.h>헤더를 사용한 함수이고, **노란색 상자**는 <wincon.h> 헤더를 사용한 함수이다. 호출 계층도는 main() 함수로부터 시작하여 게임 기본 구조인 11가지 함수들을 중심으로 작성하였습니다.

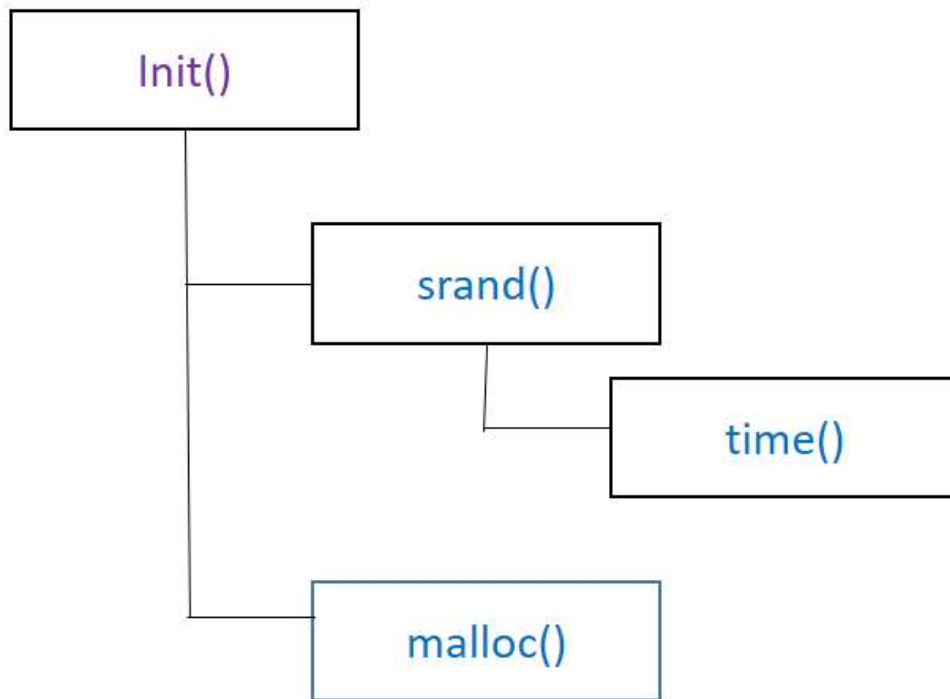
(1) main() 함수



[그림 1] main() 함수의 호출 계층도

main() 함수는 게임의 주 루틴으로서 게임의 기본 구조인 초기화(Init())등 을 기본으로 기타 여러 부가적인 함수를 포함하고 있습니다. 특히 중간에 있는 while 반복문은 게임이 종료될 때까지 반복하는데, 이는 게임을 시작하여 종료할 때까지 gotoxy(), show_game_frame(), make_food(), make_tail(), move_snake(), fflush(), show_gameover(), SetColor() 함수를 반복하여 호출하도록 하고 있고, 동시에 사용자로부터의 입력을 계속 받고 있음을 확인할 수 있습니다.

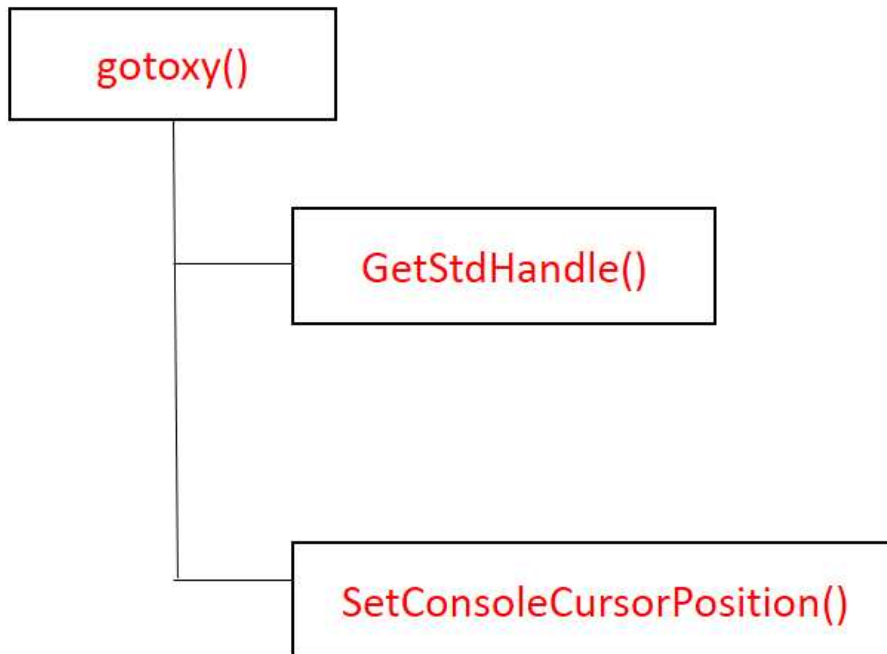
(2) Init() 함수



[그림 2] Init() 함수의 호출 계층도

`Init()` 함수는 게임의 초기화를 담당하는 함수입니다. 처음에는 `srand` 함수로 시간의 난수를 생성하고 전역 변수를 초기화한다. 그 다음에 뱀의 초기값(뱀의 꼬리, 머리 길이)을 입력한다.

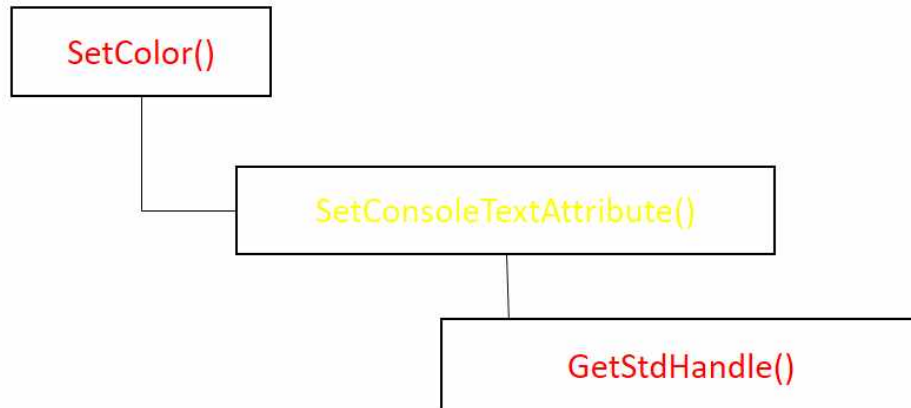
(3) gotoxy() 함수



[그림 3] gotoxy() 함수의 호출 계층도

`gotoxy()` 함수에서는 `GetStdHandle()` 함수로 반환되는 핸들은 콘솔에서 읽거나 쓰는 데 필요한 응용 프로그램에서 사용할 수 있도록 합니다. `SetConsoleCursorPosition()` 함수로 커서 위치를 제어할 때 사용합니다.

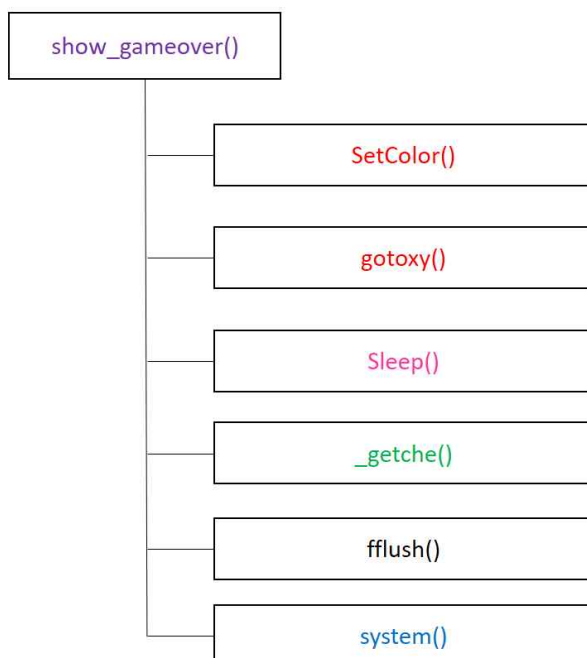
(4) SetColor() 함수



[그림 4] SetColor() 함수의 호출 계층도

선언한 정적 변수인 `HANDLE`에다가 `GetStdHandle()` 함수를 선언해서 표준 핸들값을 반환한 다음에 `SetConsoleTextAttribute()` 함수의 두 번째 전달 인자에다가 값을 넣어주어서 배경 색상과 글자 색상을 바꿀 수 있다.

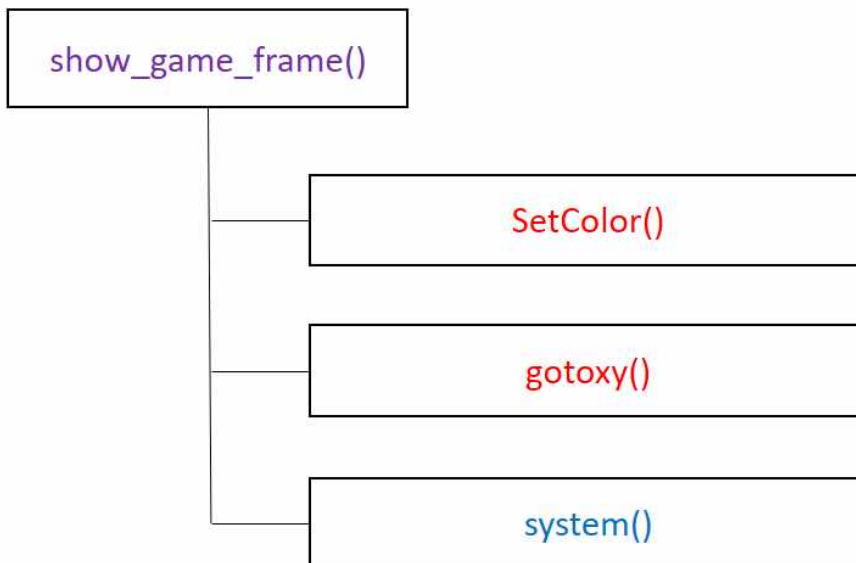
(5) show_gameover() 함수



[그림 5] show_gameover() 함수의 호출 계층도

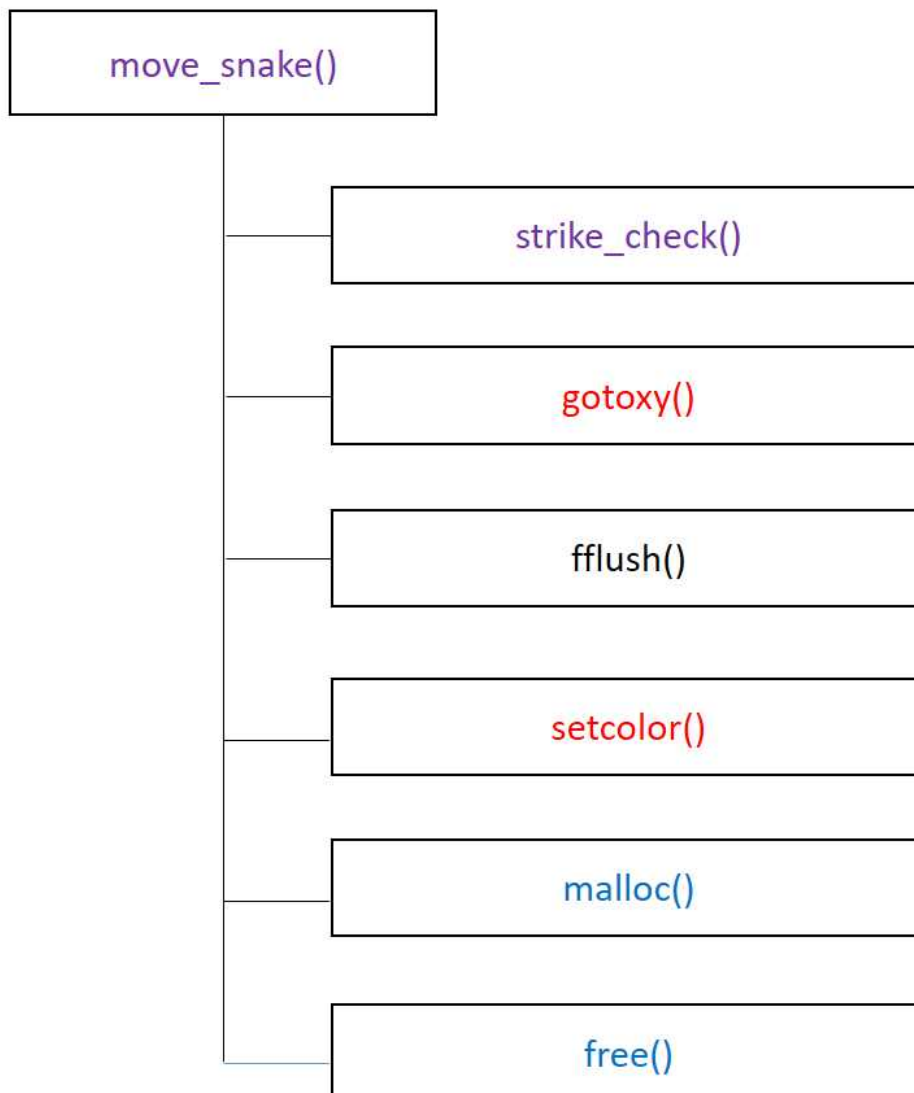
색상을 설정해준 다음에 gotoxy() 함수로 커서 위치를 지정해준 다음에 그 위치에서 구조물을 출력해준 다음 이 방법을 반복하면서 게임이 종료되었을 때 출력하는 화면을 완성한다. 완성된 화면을 1초 동안 보여주고 getch() 함수로 값을 입력받아서 fflush(stdin)으로 입력 버퍼를 지워주고, fflush(stdout)로 출력 버퍼를 지워주어서 제대로 입력 받을 수 있도록 한다. system("cls")로 화면을 클리어해준다.

(6) show_game_frame() 함수



SetColor() 함수로 색을 지정해주고 system("cls")로 화면을 클리어 해준 다음에 gotoxy() 함수로 커서 위치를 지정해주고 구조물을 출력해준다. for 반복문을 돌려서 커서 위치를 제어하면서 구조물을 출력해낸다. 그리고 몇 개의 for 반복문을 통해서 몸의 부분이 어느 특정한 위치에서 생기도록 하고 머리도 특정 위치에서 생성해서 시작하도록 한다.

(7) move_snake() 함수

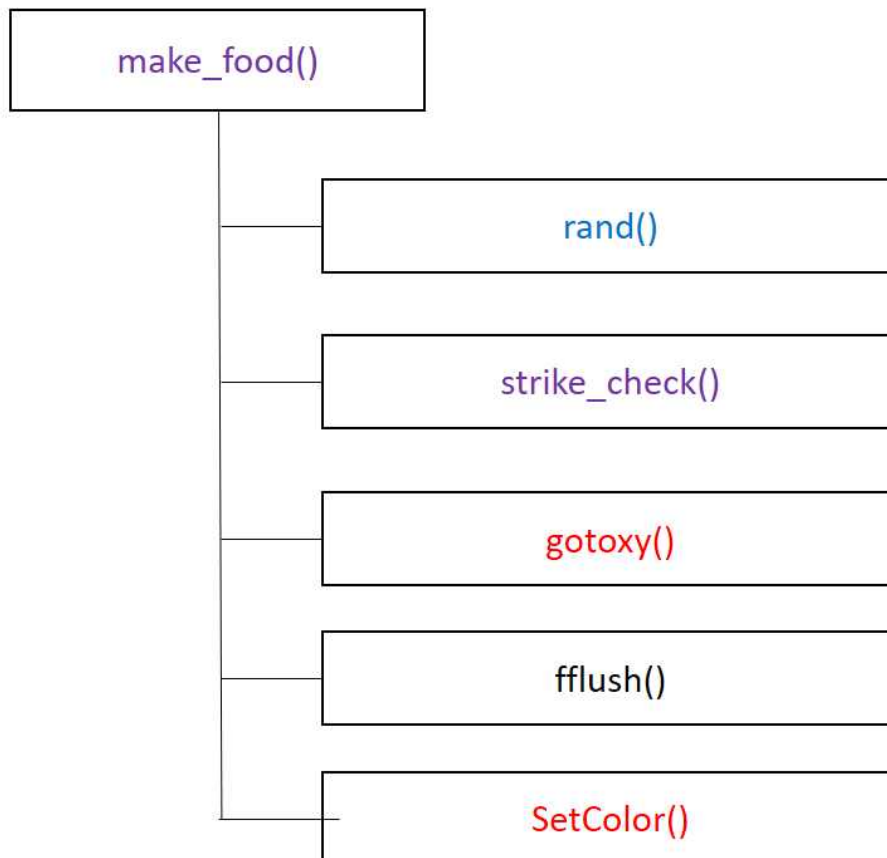


구조체로 몸에 해당하는 변수를 만들고 switch 조건문으로 뱀이 움직이는 방향을 설정한다. 충돌할 때는 1로 값을 반환한다. 그리고 gotoxy() 함수를 통해서 뱀이 이동한 후의 잔상을 지운다. gotoxy() 함수로 화면 깨지는 것을 막고 출력 버퍼 값을 지워준다. 색을 지정해준 다음에 다시 gotoxy() 함수로 이동 후의 머리 모양 잔상을 지워준다. 머리 모양은 gotoxy() 함수로 프린트를 해준다. malloc함수로 머리 부분의 새로운 노드를 만든다. 그리고 while 반복문으로 꼬리 노드를 지워준다. free() 함수로 뱀의 꼬리를 malloc 함수로 만든 값을 시스템에 반환해준다.

(8) strike_check() 함수

if 조건문으로 머리가 벽에 부딪히는지 검사한다. 그리고 while 반복문으로 몸이 null값이면 안에 조건문에 해당하면 1을 반환하고 아니면 다음으로 넘어간다. 만약에 먹이를 획득했을 때 2를 반환한다.

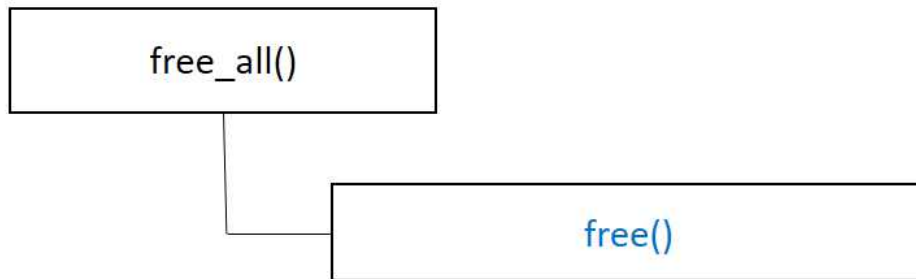
(9) make_food() 함수



`rand()` 함수로 음식을 랜덤하게 만든다. while문으로 `strike_check` 함수에서 값이 1이 반환되면 난수를 또다시 함수 초기에 선언했던 것처럼 다시 만든다.

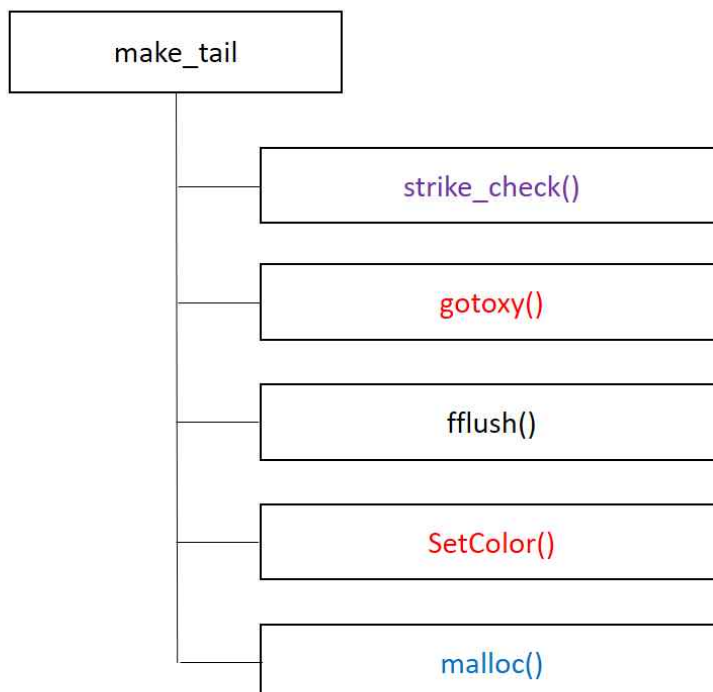
여기에서도 화면이 깨지는 것을 막기 위해서 `gotoxy()` 함수를 쓰고, 출력 버퍼 값을 지워 준다. 그리고 먹이의 색을 지정해주고 `gotoxy()` 함수로 위치를 지정해준 다음에 모양을 프린트해 준다.

(10) free_all() 함수



구조체인 `body`를 선언하고 변수를 생성해서 뱀의 머리의 값을 대입해주고 뱀의 머리가 다음으로 넘어가는 멤버에 접근하는 것을 두 번째 변수로 생성한다. `while`문에서 두 번째 변수가 `null`이 아니면 `free()` 함수로 시스템에 반환해주고 다시 처음부터 반복한다.

(11) make_tail() 함수

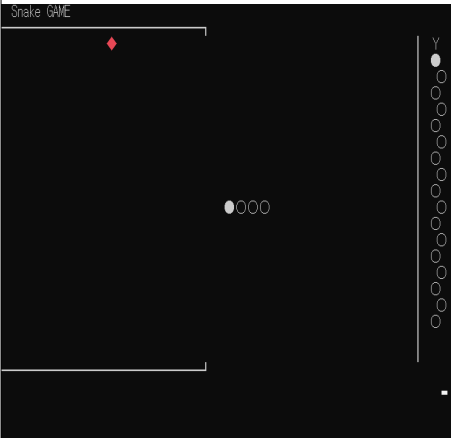



구조체인 `body`를 선언하고 뱀이 향하는 방향을 `switch`문으로 결정해서 `strike_check` 함수를 통해 반환 값이 1이 나올 때 충돌한다는 것을 알 수 있고 이때 1을 반환한다. `gotoxy()` 함수로 화면이 깨지는 것을 막고 출력 버퍼 값을 비워준다. 색깔을 지정해준 다음에 `gotoxy()` 함수로

[2020년도 게임수학 Team Project 최종보고서]

이동 후의 머리 모양 잔상을 지워주고 몸 모양을 출력해준다. gotoxy() 함수로 머리 모양을 프린트한다. malloc 함수로 머리 부분의 새로운 노드를 만든다.

6. 개작 전과 개작 후의 비교

	개작 전	개작 후
실행 화면		
비교	<p>프로그램 실행과 동시에 인게임 화면이 나오며 게임이 시작된다. 아이템을 먹을수록 꼬리가 늘어나고 벽에 부딪히면 게임 오버와 함께 아무 동작도 할 수가 없다.</p>	<p>프로그램을 실행하면 타이틀 화면이 나오게끔 int MainMenu()를 추가해 SNAKE-19로고와 게임시작화면과 게임방법을 고를 수 있게 한다. 게임 방법 메뉴를 선택하면 void game_explain()을 통해 맨 처음에 게임의 스토리를 알 수 있게 한다.</p> <p>keytemp변수를 사용해서 화살표 ->를 입력받으면 그 다음 화면으로 넘어가서 목표와 주의사항이 프린트된다. 그런 다음 똑같은 방법을 통해 화살표->를 입력받으면 게임 설명화면이 종료되고 게임 방법 화면이 나오게 된다. 게임 방법 화면은 게임을 플레이하는 조작키와 게임오버조건이 프린트 된다. 그 다음 ->키를 입력 받으면 게임 시작화면이 뜰 수 있게끔 만들었다.</p> <p>게임이 시작되면 머리와 꼬리로 구성된 캐릭터와 먹이 화면이 나온다. 먹이를 먹을수록 꼬리는 늘어나게 되고, 캐릭터가 벽에 부딪히면 gameover화면이 Sleep(1000);을 사용해 1초 동안 대기시키고, 다음 화면이 나오게 된다. 다음 화면은 게임의 주제에 맞게 코로나 예방수칙을 띄우고 keytemp를 사용해 ->키를 입력받으면 게임을 다시 시작할 수 있게끔 만들었다.</p>

7. 연구 결과에 대한 기대효과

○ 기술적 측면

- 각 물체들의 위치 및 이동을 이해하고 다룸으로서 수학적·물리적 사고력을 증진시킬 수 있다.
- 게임의 개작을 통해 소스코드 리뷰 능력과 이를 통한 응용력을 기를 수 있다.
- 현재 프로젝트와 유사한 다른 어플리케이션들을 조사하여 장단점을 찾아내고 이를 통해 프로젝트에의 적용점과 보완점을 마련한다.
- 레벨 에디터와 여러 함수들, 구조체를 학습하여 게임의 구성 요소와 작동 방식을 이해할 수 있다.

○ 경제.산업적 측면

- 프로젝트를 진행하며 기존의 어플리케이션보다 더욱 견고해진 결과물을 도출해 낼 수 있다.
- 게임 문화 사업이 더 발전할 수 있다.

○ 사회.문화적 측면

- 소통과 역할분담을 하는 팀 활동을 경험한다.
- 게임을 만드는 것과 동시에 수학적 공부가 가능하다

8. 연구 추진 일정

※ 추진일정은 가능한 한 구체적이고 상세하게 작성
 ※ 주요 결과물란은 예상되는 결과물(제안서, 프로그램 소스) 등의 주요사항 기재

주차 내용	2	3	4	5	6	7	8	9	10
1. 자료 수집									
2. 제안서 작성									
3. 게임 분석 및 학습									
4. 게임코드제작									
5. 게임 코드 수정 및 보완									
6. 최종점검									
7. ppt 작성 및 발표준비									
주요 결과물	기초 제안서	프로그램 소스 분석, 함수계층도	중간보고서, 프로그램 소스 개작	프로그램 소스, 알고리즘	최종 결과물, 최종 보고서	ppt 자료			

9. 참고 문헌

2020년도 게임수학 참고문헌 책3개, 논문3개, 인터넷3개 총 9개 이상

저자명. (발행년). 서명 (역할다른저자) (판차). 발행지: 발행사 [책 표기법]

저자명. (발행년). 논문명. 자료명, 권(호), 논문수록면수 [논문 표기법]

게임강의제목, 검색날짜, URL, 내용 [인터넷 표기법]

<https://m.blog.naver.com/PostView.nhn?blogId=goldmonji2&logNo=220726434215&proxyReferer=https:%2F%2Fwww.google.com%2F> 참고

게임프로그래밍

- [1] 로버트 나이스트롬(2018), 게임 프로그래밍 패턴: 더 빠르고 깔끔한 게임 코드를 구현하는 13가지 디자인 패턴, 박일 역, 서울: 한빛미디어
- [2] 제이슨 그레고리(2014), 게임 엔진 아키텍처: 게임 프로그래머가 꼭 알아야 할 게임 엔진 이론과 실무, 박상희 역, 의왕: 에이콘
- [3] 이태성(2010), C를 이용한 게임프로그래밍 = Game programming using C, 울산: 나우커 커뮤니케이션
- [4] 박현명(2018), 프로그래밍 언어에 대한 이해와 컴퓨팅 사고력의 향상을 위한 소프트웨어 교육용 게임에 대한 연구, 아주대학교 일반대학원, 794.8 판사항(22), p1-92
- [5] 김경식(2002), 게임프로그래밍의 교육 방향, 한국게임학회, 2권 2호, p.9-15(총 15페이지)
- [6] 김종훈, 신재훈(2001), 게임 프로그램 이해를 통한 체계적 "프로그래밍" 교수 자료 개발, 한국정보교육학회, 5권 1호, pp.133-142(총 142페이지)
- [7] (CGP) 11장 Snake 게임 툴 만들기, 2020.09.17, <https://nowcampus.tistory.com/entry/11%EC%9E%A5-Snake-%EA%B2%8C%EC%9E%84-%ED%88%B4-%EB%A7%8C%EB%93%A4%EA%B8%B0>, 스네이크 게임 툴 설명
- [8] (CGP) 12장 Snake 게임, 2020.09.17, <https://nowcampus.tistory.com/entry/12%EC%9E%A5-Snake-%EA%B2%8C%EC%9E%84>, 스네이크 게임 소스 코드와 상세 설명
- [9] [Snake Game] 0. 뱀 게임(Snake Game), 2020.09.17., <https://thisisvegetable.tistory.com/17?category=767984>, 스네이크 게임이란 무엇인가?