2020 Spring OOP Assignment Report

과제 번호 : ASSN3
학번 : 20190439
이름 : 오승훈
Povis ID : sho0927

명예서약 (Honor Code)

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.
I completed this programming task without the improper help of others.

프로그램을 하다 보면 결정해야 할 세부 사항이 많은데, 이러한 세부 사항을 처리한 방법과 이유를 보고서에 쓰십시오.
독창적인 아이디어와 추가 기능은 보너스 점수를 받을 수 있으므로, 보고서에 명확히 기재하십시오.
제출 기한이 지나면 LMS를 통하여 과제를 제출할 수 없습니다. 제출 기한이 지난 후에는 담당 조교에게 email로 보내야 합니다. 늦게 제출된 과제에 대한 감점은 담당 조교에게 email이 도착한 시간을 기준으로 합니다.
문제가 여러 개인 경우, 각 문제별로 정리해서 작성합니다.

각 문항별 설명은 편의를 위한 것으로, 삭제하고 제출한다.

1. 프로그램 개요

a. ASSN3.

A. Assn3는 우리가 흔히 아는 테트리스 구현을 목표로 한다.

B. 필요한 메소드들을 생각해보면, 오른쪽, 왼쪽, 아래, 반시계 회전, 시계 회전, 맨 밑으로 가기 등이 있다.

C. 또한 이번 어싸인의 경우 상속의 개념을 익히는 것을 목표로 하기 때문에 구조는 다음과 같다. Base class 는 Block class이다. 거기에 공통으로 들어갈 method들을 구현을 해두고, 그 후 Derived class에서 I,Z,O,S,L,J,T 각각의 블록 모양을 구성할 수 있도록 구현을 하면 된다.

D. 또한 이번 어싸인 클래스에서 Derived class, Base class는 X[4][4],. Y[4][4]의 배열을 가지고 있다. 스핀 값에 따라서 달라지는 좌표값들을 모두 가지고 있다고 생각을 하면 된다.

E. 또한 하나의 클래스가 더 있는데 그것은 Base class와 유사하지만 Field가 차있을 때 저장을 할 수 있는 Frame class가 된다. 이 클래스는 Field의 블록이 차있는 것을 display 할 때 또는 맨 아래가 어디인지 등을 판별할 때 사용이 된다.

F. 또한 추가구현을 한 것 중에 combo가 가능하며, T 스핀 역시 detect가 가능하도록 구현을 해뒀다. 또한 back-to-back을 할 경우 역시 추가 점수가 있는 것을 만들어 뒀다.

G. 또한 본인이 생각했을 때 테트리스의 묘미를 살려주는 것은 홀드 기능이라고 생각했기 때문에 홀드 기능 역시 추가 기능으로 구현해뒀다.

2. 프로그램의 구조 및 알고리즘

a. Block 클래스

Block private 변수	int x[4][4] : 0을 기점으로 시계 방향 회전시 1,2,3의 값을 가지고 그 뒤 크기가 4개짜리 배열은 그 회전에 맞는 x 값을 가지고 있다. int y[4][4] : y는 x와 동일하다. string msg : msg에는 블록의 특수문자 기호를 가지고 있다. Color color : color에서는 블록의 색깔을 저장하고 있다. int tempt_spin : 현재 스핀 값을 가지고 있다. bool non_empty : 이 블록이 비어있는지를 체크해준다. 이는 hold 기능을 구현하기 위해서 사용하였다.
Block public 함수	Block() Block(int, int, Color) Block(int, int, Color, string) : 3 개 모두 다양한 방법으로 사용되는 생성자 입니다. int getX(int i, int j) {return x[i][j];} spin 값과 원하는 index를 int getY(int i, int j) {return y[i][j];} 넣으면 x,y를 반환합니다. bool get_empty(){ return non_empty; } 이 block 값이 비어있는지를 반환해줍니다. 이는 후에 사용되는 hold 기능을 위해 필요합니다. int get_spin() { return tempt_spin; } spin 값을 반환해줍니다. virtual int spin_clock(Frame game_frame[][20]) 시계 방향 회전 virtual int spin_counter_clock(Frame game_frame[][20]) 반시계 회전 void move_left(Frame game_frame[][20]) 왼쪽으로 이동 가능하면 이동 void move_right(Frame game_frame[][20]) 오른쪽으로 이동 가능할 시 이동 void move_down() 아래로 내려갑니다. void display() 현재 블록을 보여줍니다. void bottom_display() 그림자를 보여줍니다. << 출력 값이

	다르므로 따로 구현됨 void next_display() 다음을 보여줍니다. << 좌표 값이 다름 void hold_display() 홀드된 것을 보여줍니다. << 좌표 값이 다름 void set_empty(bool empty_) { non_empty = empty_; } << non_empty를 설정함 void move_up() 때로는 위로 올라가야하는 상황이 나오기에 필요함 void move_only_left() 왼쪽으로 이동할 수 있는지를 검사하지 않고 그냥 왼쪽으로 한 칸 이동시킵니다. void move_only_right() 오른쪽으로 이동할 수 있는지를 검사하지 않고 그냥 오른쪽으로 한 칸 이동시킵니다. bool is_it_hit_block(Frame game_frame[][20], int next_spin) field에 이미 차 있는 block과 부딪히는지를 출력해줍니다. Spin을 위해 필요합니다. bool is_it_bottom(Frame gameframe[][20]) 맨 밑인지를 확인해줍니다. bool is_it_okay_move_left(Frame gameframe[][20]) 왼쪽으로 이동해도 되는지를 확인합니다. bool is_it_okay_move_right(Frame gameframe[][20]) 오른쪽으로 이동해도 되는지를 확인합니다. bool is_it_okay_move_up(Frame gameframe[][20]) 위로 올립니다. string getMsg() {return msg;} msg를 받아갑니다. Color getColor() const {return color;} color를 받아갑니다. void set_spin(int spin) { tempt_spin = spin; } 스핀 값을 설정합니다. void setX_Y(int x_[][4], int y_[][4]) X,Y 좌표를 설정합니다. void setMsg(string msg_) {msg = msg_[0];} msg를 설정합니다. void setColor(Color color_) {color = color_;} color를 설정합니다. Block operator+(const Block &pt) 사용되지 않지만 조교님이 구현을 해둔 것이라서 뒀습니다 virtual ~Block() {}

A. 먼저 이번 Block class를 보면 모든 다른 하위 블록에서 사용될 메소드들이 구성이 되어있습니다. Clock Spin과 counter spin의 경우 아무래도 T 스핀과 I의 스핀의 경우 구현이 다르게 되어있기 때문에 virtual 키워드가 붙었습니다.

b. Frame

Member 변수	int x : field의 x 값을 저장해둡니다. int y : field의 y 값을 저장해둡니다. string msg : 특수 문자를 기억해둡니다.

	Color color : 그 블록이 무슨색으로 출력되야하는지를 알려줍니다. bool non_empty : 비어있으면 false 채워져있으면 true를 넣어둡니다.

Member 함수	Frame() { non_empty = false; }
	int getX() { return x; }
	int getY() { return y; }
	bool get_empty() { return non_empty; }
	void setX_Y(int x_, int y_) { x = x_; y = y_; }
	void setMsg(string msg_) { msg = msg_[0]; }
	void setColor(Color color_) { color = color_; }
	Color getColor() const { return color; }
	void set_empty(bool empty_) { non_empty = empty_; }
	void show_frame(int score);
	void display();

A. 이 클래스의 경우 Field의 틀을 구성할 클래스이기 때문에 아무래도 Block과 유사하지만 다른 점은 private에서 x,y가 배열이 아니고, spin 값을 가지고 있지 않다는 점입니다. 이 이유는 이 클래스의 경우 실제 main 함수에서 [10][20]으로 구현이 되어 있어서 실제 블록이 맨 밑에 닿여서 저장되어야 하는 곳에 이 field에 맞는 index에 해당하는 곳에 color 와 non_empty 등을 변경해줘서 필드를 출력할 수 있도록 해주는 것을 담당하고 있는 클래스입니다.

c. Tetrimino_I,O,Z,S,T,L,J 클래스

A. 모두 public으로 Block을 상속 받는 클래스이다.

Tetrimino_I,T함수	tetrimino_I();
	tetrimino_T();
	int spin_clock(Frame game_frame[][20]);
	int spin_counter_clock(Frame game_frame[][20]);

Tetrimino_Z,S,O,L,J 함수	tetrimino_O();
	tetrimino_Z();
	tetrimino_S();
	tetrimino_L();
	tetrimino_J();

B. 이 클래스의 경우 block을 상속 받기 때문에 많은 method들이 들어가 있지 않다. 이 경우 생성자를 통해서 현재 블록의 x,y 값을 설정하는 작업을 진행하게 되고, 그 외에 것들에 대해서는 이미 block의 생성자가 모든 것들을 해둔 뒤에 이루어지기 때문에 많은 것들을 할 필요는 없다. 하지만 spin_clock 과 spin_counter_clock이 I, T에 따로 구현이 되어 있는 이유는 I 미노의 경우 spin에서의 SRS 시스템이 다른 블록들과 다르기 때문이고, T의 경우 T 스핀 판별을

위한 wallkick 으로 인해 return 값이 다르기 때문이다.

d. 전반적인 코드에 대한 설명

A. 먼저 구조를 보면 이번에 본인이 구현해둔 코드는 Block 밑에 7개의 Derived 클래스가 존재하고, Frame은 별개의 클래스로 존재하는 코드가 된다. 또한 Block에서 필요한 모든 method들을 구현해두고, 이 클래스를 상속받아서 진행을 하게 된다. 또한 main.cpp에 대해서 조금 살펴보자면 main.cpp에서는 game을 진행할 때 7개의 동적 할당 받은 Block들을 차례대로 전달을 해주면서 게임이 진행이 되며, score를 얻었거나, T스핀을 통해서 구현이 되어 있거나와 같은 것들에 따라서 game 진행중인 함수에서 return 값이 달라지는 것을 통해서 combo나 Back-To-Back 추가 점수를 구현할 수 있도록 해두었다. 또한 main.cpp에서 파일 출력을 위해서 동적할당을 받고, 원하는 값까지 저장할 수 있도록 해두었다.

e. 추가 기능

A. T-Spin detect

i. 먼저 T스핀 미니와 싱글의 차이점에 대한 기준을 잘 모르겠어서 본인이 판단을 한 기준으로 T-spin 미니와 싱글에 대한 판별을 진행하였다. 하지만 이 판별이 옳은 것인지에 대한 것은 아직 의문으로 작용을 하고 있다.

ii. 일단 T 스핀 미니는 월킥이 발생했을 때의 경우인 것 같아 T 스핀 미니의 기준이 한 줄을 제거하는데 월킥이 발생했을 때의 기준으로 잡았고, T 스핀 싱글의 경우 한 줄을 제거하는데 월킥이 발생하지 않았을 때의 기준으로 잡았다.

iii. T 스핀 싱글과 미니가 잘못되었을 수도 있다. 하지만 T 스핀 더블 부터는 T 스핀이 발생했을때를 기준으로 했다.

iv. T 스핀 트리플 역시 T 스핀이 발생했을때를 기준으로 해서 3줄이 없어진 경우이다. 또한 T 스핀 트리플이 발생하려면 사이드 블록(T자의 중심으로부터 대각선을 이루는 블록)들이 무조건 채워져 있는 경우에만 발생하기 때문에 사이드가 채워져 있는지를 체크하는 함수에서 4가 반환이 되어야지 T스핀 트리플이 발생했다고 볼 수 있다.

B. Combo

i. 콤보의 경우 점수가 났을 경우에는 콤보를 유지하고, 점수가 나지 않았을 때는 다시 콤보 값을 0으로 반환하는 방식으로 구현을 해 뒀다.

ii. 점수가 났는지는 tetrimino가 바꼈을때를 기준으로 삼았는데 이 tetrimino가 바뀔 때 점수가 났을때와 나지 않았을 때의 return 값을 다르게 구현을 해두

어서 이 것을 판별할 수 있도록 해뒀다.

C. Back-to-Back

i. 백투백의 경우 T 스핀이 발생했을때의 return 값을 달리하여 이 것이 두 번 연속으로 발생했을 경우에만 Back-To-Back이 나타나도록 구현을 해뒀다.

ii. 백투백의 경우 combo와 동시에 발생했을 경우에 둘 다 점수가 오르도록 구현을 해두었다.

D. Hold 기능

i. 홀드 기능은 테트리스에서 존재하는 모든 홀드 기능과 동일하다 현재 블록을 저장을 해주고, 만약 방금 홀드되있던 블록에서 들고왔던 블록을 다시 홀드를 하는 위치 바꾸기를 하려고 하면 그 경우에는 바뀌지 않는다.

ii. 하지만 현재 하나의 단점이 있다면 첫번째 홀드된 블록의 경우에는 위의 조건을 만족하지 않고, 계속 바꿀 수있다는 에러가 존재한다.

3. 토론 및 개선

□ 이번 코드에서 한계점은 일단 출력이 제대로 안되는 문제이다. 출력이 제대로 되지만 반짝거리는 것 때문에 눈이 아프게 출력이 되고 만다. 또한 150/ 1+(0.03 * (x/1000)) = 1000 / 1 + (0.1 * (x/1000)) 이 값을 만족하는 x 점수부터는 키를 클릭이 안되게 구현이 되어있다. 이는 키를 어느정도 이상 눌러야 키를 입력했다고 받아들이는 코드로 구현을 해뒀기 때문인데 이는 0.03 또는 0.1을 조정을 해가면서 더 높은 점수까지 구현이 가능하도록 할 수 있다. 이 코드가 있는 이유는 이 코드가 없으면 a,w,s,d 와 같은 입력에서 너무 낮은 감도로 인해서 d를 한 번 누르는 입력을 원했는데 실제 입력은 10번이 넘게 되는 사태를 막기 위해서 있는 코드이다. 그렇기 때문에 뒤로 갈수록 입력에 대한 감도가 짧아진다는 단점이 존재하긴 하지만 이 보정값이 없다면 점수가 80000점 정도만 되도 이 코드는 입력 불가로 인해 더 이상 게임을 플레이 하지 못하기 때문에 이러한 보정 값들을 넣어두었다.

□ 이번 과제를 통해서 가장 아쉬웠던 점은 출력의 문제이다. 내가 테트리스를 하고 있는데 현재와 같은 커서의 움직임이 계속 보이거나, 또는 반짝반짝 거리는 블록을 보면서 플레이를 하고 있으면 눈이 아파서 오랜 시간 하지 못할 것 같았다. 하지만 이 문제는 해결할 수 있는 선에 존재하는 문제가 아니었다. 동작이 된 후 블록을 출력을 하면 visual studio 에서는 코드가 정상적으로 돌아가지만 Mingw 에서 컴파일을 돌릴 경우 컴파일이 이상하게 되는 에러가 발생한다. 왜 이런 현상이 발생하는지는 확인하지 못했다. 또한 이번 어싸인에서 pdf 상에서 나와있는 정보가 너무 모호하다는 생각을 했다. SRS 시스템이나, wall_kick 또는 T 스핀에 대한 정보는 아무것도 나

와있지 않았다. 테트리스를 어느정도 한 사람이나 알법한 그런 회전에 관련된 시스템들에 대한 정보를 하나도 주지 않고, 어싸인을 구현을 하게 한다는 것은 테트리스에 친근하지 않은 사람에게는 상속에 대한 C++ 공부보다는 오히려 테트리스 공부를 먼저 해야한다는 단점으로 작용하게 됐던 어싸인이라고 생각이 된다.

- 위에서 언급한 것과 같이 이번 어싸인을 통해서 상속에 대한 공부를 많이 했다고 느껴지지는 않는다. 상속이 필요한 부분은 사실 block에서 모든 메소드를 구현을 할 수 있는 구조였고, 오히려 구현에 어려움을 느꼈던 것들은 위에서 언급된 wall kick 또는 SRS 시스템과 같은 것들이었다. 그렇기 때문에 아무래도 계속해서 검색을 해봐야했던 것들은 상속과 관련된 C++ 문법들이 아닌 테트리스 그 자체에 대한 것들이었다. 과연 우리가 공부해야 했던 것은 C++ 문법이었는지 아니면 테트리스였는지에 대해서 생각을 해본다면 당연히 C++ 문법에 대해서 공부를 해야한다고 생각하기에 이 어싸인이 유의미한 어싸인이었는지에 대해서 의문을 갖고 구현을 했던 것 같다.

- 마지막으로 이제 곧 나오게 될 어싸인 3는 클래스들의 hierarchy를 이용하는 것이라고 했는데 이번 어싸인을 더 발전시켜서 만들 수 있는 것들을 생각해보면 facebook의 개인 글 포스팅하기, 댓글, 좋아요, 친구추가와 같은 것들이 구현이 되어있으므로, 페이스북 그룹 만들기 또는 페이지 만들기와 같은 것들이 만들어지지 않을까? 하는 예상을 하며 이번 보고서를 마무리한다. << 이는 내가 어싸인 2 보고서에서 언급을 한 내용이다. 이러한 것과 같은 어싸인을 만들었다면 오히려 더 깔끔한 상속에 대한 공부를 할 수 있는 어싸인이 되지 않았을까?

- 추가 기능에 대한 기준이 모호하다는 아쉬운점 역시 남는 것 같다. 특히 T 스핀의 경우 T 스핀 미니와 싱글의 기준을 솔직히 아직까지 무슨 차이인지를 모르겠다. 이러한 것과 같이 테트리스 내에서도 확실한 기준이 나오지 않는 것에 대해서는 조교님이 좀 더 정확한 기준을 줬으면 좋겠다는 생각이 든다.

- 마지막으로 이번 어싸인과 같은 어싸인이 나온 후 조교님이 번복을 하는 사태가 다시 발생하지 않기 위해서는 조교님이 어싸인에 대한 아이디어만을 구상해 볼 것이 아니라 이번 어싸인 구현을 완벽하게 해보고 이번 어싸인에서 필요한 부분이 mingw 환경과 잘 호환이 되는지와 같은 것을 확인해본 후 어싸인을 내는 것이 맞지 않나라는 생각이다.

4. 참고 문헌

- https://www.youtube.com/watch?v=_Pu4wHzncnw&feature=youtu.be
- https://hse30.tistory.com/82
- https://hse30.tistory.com/82
- https://blog.naver.com/jeff_jks/221254074573

- https://m.blog.naver.com/power2845/50143021565