실습 레포트

실습명: 텀 프로젝트 (14 주차 실습)

2022-2 학기 컴퓨터프로그래밍 및 실습
이름: 김건하
학과: 컴퓨터.전자시스템공학부
학번: 201900532

2022. 12. 06

1. 총알 피하기

- 코드:

<MyCircle.h>

```
#pragmaonce
#include<SFML/Graphics.hpp>
#include<iostream>
#include<string>
#include<cmath>
#include<windows.h>

usingnamespacestd;
usingnamespacesf;

classMyCircle{
private:
        CircleShapecircle;

        doubleradius;
        doubleposX;
        doubleposY;
        doublevelocity;
        doubledx;
        doubledy;

public:
        MyCircle(); // default constructor
        MyCircle(doublex, doubley, doublerad); // constructor by x, y, rad

        voidsetVelocity(doublevel);
        doublegetVelocity();
        voidsetRadius(doublerad);
        voidsetPosition(doublex, doubley);
        doublegetPosX();
        doublegetPosY();
        voidsetColor(intr, intg, intb);
        voidmove(doubledx, doubledy);
        voidsetdxdy(doubledx, doubledy);
        doublegetdx();
        doublegetdy();
        CircleShapegetCircle();
        boolcollisionTest(MyCircleobj); // collision test -> return true if collision
};
```

<MyCircle.cpp>

```
#include"MyCircle.h"

MyCircle::MyCircle() { // default constructor
        posX = 0;
        posY = 0;
        radius = 10;
        velocity = 2;

        circle.setPosition(posX, posY);
        circle.setRadius(radius);
        circle.setFillColor(Color(0, 255, 0)); // circle color RGB
        circle.setPointCount(30);
}

MyCircle::MyCircle(doublex, doubley, doublerad) { // constructor by x, y, rad
        posX = x;
```

```cpp
    posY = y;
    radius = rad;
    velocity = 2;
    circle.setPosition(posX, posY);
    circle.setRadius(radius);
    circle.setFillColor(Color(0, 255, 0)); // circle color RGB
    circle.setPointCount(30);
}

voidMyCircle::setVelocity(doublevel) {
    velocity = vel;
}

doubleMyCircle::getVelocity() {
    returnvelocity;
}

voidMyCircle::setRadius(doublerad) {
    radius = rad;
}

voidMyCircle::setPosition(doublex, doubley) {
    posX = x;
    posY = y;
    circle.setPosition(x, y);
}

doubleMyCircle::getPosX() {
    returnposX;
}

doubleMyCircle::getPosY() {
    returnposY;
}

voidMyCircle::setColor(intr, intg, intb) {
    circle.setFillColor(Color(r, g, b));
}

voidMyCircle::move(doubledx, doubledy) {
    posX += dx;
    posY += dy;
    circle.move(dx, dy);
}

doubleMyCircle::getdx() {
    returnthis->dx;
}

doubleMyCircle::getdy() {
    returnthis->dy;
}

voidMyCircle::setdxdy(doubledx, doubledy) {
    this->dx = dx;
    this->dy = dy;
}

CircleShapeMyCircle::getCircle() {
    returncircle;
}

boolMyCircle::collisionTest(MyCircleobj) {
    doubled = sqrt(pow(((posX + radius) - (obj.posX + obj.radius)), 2) + pow(((posY +
radius) - (obj.posY + obj.radius)), 2));

    if(d <= (radius + obj.radius)) {
    returntrue;
```

```
	}
	else{
	returnfalse;
	}
}
```

<main.cpp>

```
#include<SFML/Graphics.hpp>
#include<iostream>
#include<string>
#include<vector>
#include<time.h>
#include<cmath>
#include<random>
#include<windows.h>
#include"MyCircle.h"

usingnamespacestd;
usingnamespacesf;

MyCirclesetEnemy(MyCircleplayer);

intmain() {
	cout <<"GAME START"<<endl;

	intnX = 1600; // display size
	intnY = 900;
	RenderWindowwindow(VideoMode(nX, nY), "Moving Ball");
	window.setFramerateLimit(100);

	// Player circle info
	doublepRadius = 10;
	doublepPosX = 800;
	doublepPosY = 450;
	doublepVelocity = 4;

	MyCircleplayer{ pPosX, pPosY, pRadius }; // set player circle
	player.setVelocity(pVelocity);
	player.setColor(52, 204, 255); // player color set blue

	// enemy list
	intenemyNum = 39; // enemy number
	vector<MyCircle> enemyLst;
	for(inti = 0; i < enemyNum; i++) {
	enemyLst.push_back(setEnemy(player));
	}
	intflag = 0; // flag for increasing enemyNum

	// magEnemy list
	doubleeRadius = 7;

	MyCirclemagEnemy{ 0, 0, eRadius };
	magEnemy.setColor(225, 50, 50); // magEnemy color set red

	intmagEnemyNum = 0; // player following enemy number
	vector<MyCircle> magEnemyLst;
	//for (int i = 0; i < magEnemyNum; i++) {
	//	magEnemyLst.push_back(magEnemy); //setmagenemy
	//}

	TexttTime; // display time
	TexttEnemy; // dispaly enemy

	Fontfont;
	intt = 0;
	inte = 0;
```

```
        if(!font.loadFromFile("C:₩₩Users₩₩qkqcu₩₩source₩₩repos₩₩Bullet Dodge₩₩Bullet
Dodge₩₩arial.ttf")) { // check font file route!
        //C:₩₩Users₩₩qkqcu₩₩source₩₩repos₩₩Bullet Dodge₩₩Bullet Dodge₩₩arial.ttf"
        return42; // Robust error handling!
        }

        // time text set
        tTime.setFont(font);
        tTime.setCharacterSize(25);
        tTime.setFillColor(Color::White);
        tTime.setPosition(1525, 860);

        // enemy num text set
        tEnemy.setFont(font);
        tEnemy.setCharacterSize(25);
        tEnemy.setFillColor(Color::Magenta);
        tEnemy.setPosition(0, 0);

        clock_ttime = clock();

        // game loop
        while(window.isOpen()) {
        // check event
        Evente;
        while(window.pollEvent(e)) {
        if(e.type == Event::Closed)
        window.close();
        }

        // move player circle by keyboard
        if(Keyboard::isKeyPressed(Keyboard::Up)) {
        player.move(0, -player.getVelocity());
        }
        elseif(Keyboard::isKeyPressed(Keyboard::Down)) {
        player.move(0, player.getVelocity());
        }
        if(Keyboard::isKeyPressed(Keyboard::Left)) {
        player.move(-player.getVelocity(), 0);
        }
        elseif(Keyboard::isKeyPressed(Keyboard::Right)) {
        player.move(player.getVelocity(), 0);
        }


        // move enemy -> follows player
        for(inti = 0; i < magEnemyNum; i++) {
        doublel = sqrt(pow(player.getPosX() - magEnemyLst[i].getPosX(), 2) +
pow(player.getPosY() - magEnemyLst[i].getPosY(), 2));
        doubledx = (player.getPosX() - magEnemyLst[i].getPosX()) / (l/2);
        doubledy = (player.getPosY() - magEnemyLst[i].getPosY()) / (l/2);
        magEnemyLst[i].setdxdy(dx, dy);
        magEnemyLst[i].move(magEnemyLst[i].getdx(), magEnemyLst[i].getdy());
        }

        // move enemy -> toward player
        for(inti = 0; i < enemyNum; i++) {
        enemyLst[i].move(enemyLst[i].getdx(), enemyLst[i].getdy());
        if(enemyLst[i].getPosX() >= window.getSize().x || enemyLst[i].getPosX() <= 0 ||
enemyLst[i].getPosY() >= window.getSize().y || enemyLst[i].getPosY() <= 0) {

        // enemy 가 화면 밖으로 나갈 시 객체 소멸
        enemyLst.erase(enemyLst.begin() +i);
        enemyNum -= 1;

        // 새로운enemy 객체 생성
        enemyLst.push_back(setEnemy(player));
        enemyNum += 1;
        }
        }

        //collision test . enemy
        for(inti = 0; i < enemyNum; i++) {
        if(player.collisionTest(enemyLst[i])) {
        // game over
        cout <<"GAME OVER"<<endl;
        cout <<"Your score is "+to_string(time) <<endl;
        window.close();
        }
        }
        //collision test . magEnemy
        for(inti = 0; i < magEnemyNum; i++) {
        if(player.collisionTest(magEnemyLst[i])) {
        // game over
        cout <<"GAME OVER"<<endl;
        cout <<"Your score is "+to_string(time) <<endl;
        window.close();
        }
        }

        time = clock();
        time = time / CLOCKS_PER_SEC;
        tTime.setString(to_string(time) +" sec");
        tEnemy.setString("Enemy: "+to_string(enemyNum + magEnemyNum));

        // 15초가 지나면magEnemy 생성
        if(time == 15 && magEnemyNum == 0) {
        magEnemyLst.push_back(magEnemy);
        magEnemyNum += 1;
        }

        // 5초마다enemy 객체 추가
        if(time % 5 == 0 && flag == 0) {
        enemyLst.push_back(setEnemy(player));
        enemyNum++;
        flag++;
        }

        // enemy 생성 제한
        if(time % 5 == 1) {
        flag = 0;
        }


        // erase monitor
        window.clear();

        // draw enemy
        for(inti = 0; i < enemyNum; i++) {
        window.draw(enemyLst[i].getCircle());
        }

        // draw magEnemy
        for(inti = 0; i < magEnemyNum; i++) {
        window.draw(magEnemyLst[i].getCircle());
        }

        // draw player
        window.draw(player.getCircle());

        // draw text
        window.draw(tTime);
        window.draw(tEnemy);

        // display monitor
        window.display();
        }
```

```
        return0;
 }

 // function set enemy
 MyCirclesetEnemy(MyCircleplayer) { // player 의 위치를 입력받아enemy의vector를player를
향하도록 생성
         //generate random device
         random_devicerd;
         mt19937gen(rd());
         uniform_int_distribution<int> startposX(20, 1580); // starting x point seed
         uniform_int_distribution<int> startposY(20, 880); // starting y point seed
         uniform_int_distribution<int> randdirection(0, 3); // raandom direction 0 ~ 3
         uniform_int_distribution<int> randomVel(100, 200); // random velocity

         doubleeRadius = 4;
         doubleeVelocity = 2;

         MyCircleenemy{ 0, 0, eRadius }; // set enemy circle
         enemy.setColor(255, 255, 0); // enemy color set yellow

         intdir = randdirection(gen);
         if(dir == 0) {
        enemy.setPosition(startposX(gen), 10); // set enemy position top
         }
         elseif(dir == 1) {
         enemy.setPosition(1590, startposY(gen)); // set enemy position right
         }
         elseif(dir == 2) {
         enemy.setPosition(startposX(gen), 890); // set enemy position bottom
         }
         elseif(dir == 3) {
         enemy.setPosition(10, startposY(gen)); // set enemy position left
         }

         // set enemy vector tdoward player
         doublel = sqrt(pow(player.getPosX() - enemy.getPosX(), 2) +
pow(player.getPosY() - enemy.getPosY(), 2));
         doubledx = (player.getPosX() - enemy.getPosX()) / (l / (randomVel(gen)/ 50));
         doubledy = (player.getPosY() - enemy.getPosY()) / (l / (randomVel(gen)/ 50));
         enemy.setdxdy(dx, dy);

         returnenemy;
 }
```

- 진행상황:

기본 게임 구현 완료
enemy 개수 40 개로 시작
화면 밖 random 한 위치에서 player 를 향해 발사
random 한 속력을 가짐
벽에 충돌시 소멸 후 다시 random 한 위치에서 생성
5 초에 1 개씩 enemy 증가
직선운동이 아닌 player 를 따라가는 magEnemy 객체 구현
15 초에 1 개 생성되어 계속 player 를 따라다님
player 와 enemy 충돌시 게임종료 후 GAME OVER, 점수(시간) 출력
난이도 적절함

- 계획:

player, enemy 모양?
player 의 생명 추가?

2. 느낀점

단순히 몇일동안 간단한 프로그램을 만드는 것이 아닌, 3 주라는 긴 시간동안 지금까지 만들어온 프로그램보다는 복잡한 게임을 직접 구현에 보면서 여러 가지 어려운 점들이나 구현하기 힘들었던 부분들을 천천히 고민해보고 다양한 방법으로 해결해나가면서 흥미를 느낄 수 있었다.