面向对象程序设计（JAVA）大作业

所选题目名称：飞机大战

班 级：计算机科学与技术二班

学 号：202205720832

姓 名：何新成

一、项目概述

本项目内容为设计飞机大战游戏。实现地图展示、方向控制，敌我子弹发射，碰撞检测等基本射击游戏元素。同时完成了游戏存盘和读盘功能，排行榜功能功能，多种子弹、多种敌机、多种玩家飞机选择、子弹威力升级系统，炸弹功能，道具功能，游戏难易程度调节，游戏音乐与音效，BOSS战等功能。使游戏具有一定的可玩性，达到了飞机大战游戏基本要求。

1. 功能分析与设计及其主要代码

1.地图展示

创建了三个地图类Map1Obj、Map2Obj、Map3Obj，分别对应三个不同的难度，地图为一张高度为1800的图片，会自动增加其y值，并且会自动回调，实现了飞机在向前飞行的效果。

public void paintSelf(Graphics g) {  
 super.paintSelf(g);  
 if(GameWin.*gameState*==2) {  
 y += speed;  
 if (y >= 0) {  
 y = -900;  
 }  
 }  
}

2.方向控制

方向控制采用鼠标控制，使用addMouseMotionListener获取鼠标位置，并赋给playerX和playerY。

*playerX*=e.getX();  
*playerY*=e.getY();

3.敌我子弹发射

给每个能发射子弹的飞机都创建了他自己的子弹类和集合，采用定时器Timer来定时创建子弹并加入GameObjList来绘制。

//生成敌机子弹  
for(Enemy3Obj enemy3Obj: GameUtils.*enemy3ObjList*){  
 if(TimerOne(1000,1,GameUtils.*enemy3ObjList*.indexOf(enemy3Obj))){  
 double x = (GameUtils.*enemy3ObjList*.get(GameUtils.*enemy3ObjList*.indexOf(enemy3Obj))).getX()+63-6.5;  
 double y = (GameUtils.*enemy3ObjList*.get(GameUtils.*enemy3ObjList*.indexOf(enemy3Obj))).getY()+116;  
 GameUtils.*enemy3BulletObjList*.add(new Enemy3BulletObj(GameUtils.*enemy3bulletImg*, 13, 30, x, y, this));  
 GameUtils.*gameObjList*.add(GameUtils.*enemy3BulletObjList*.get(GameUtils.*enemy3BulletObjList*.size() - 1));//把新new出来的子弹对象添加到所有元素集合中  
 }  
}

4.碰撞检测

创建对象时初始化其长度和高度参数，使用for each配合getRec().intersects来判断要检测碰撞的两者是否有重叠区域。

if(GameWin.*invincibility*[0]==0 && GameWin.*TimerZero*(1000, 0, 0)) {  
 //玩家碰撞敌机  
 for (Enemy1Obj enemy1Obj : GameUtils.*enemy1ObjList*) {  
 if (this.getRec().intersects(enemy1Obj.getRec())) {  
 GameWin.*invincibility*[0] = 1; //玩家飞机闪烁  
 GameWin.*invincibility*[1] = 1;  
 this.hp--;  
 }  
 }

5.游戏存盘和读盘

创建了SLTool的类，用来专门执行存读档功能，将游戏数据保存在data.txt中，利用文件读写相关函数来实现。

public static void save() {  
 try ( BufferedWriter writer = new BufferedWriter(new FileWriter(*filePath*))) {  
 // 写入数据到文件  
 writer.write("@player\n");  
 writer.write("name = "+GameWin.*playerName*+"\n");  
 writer.write("x = "+GameWin.*playerX*+"\n");  
 writer.write("y = "+GameWin.*playerY*+"\n");  
 writer.write("hp = "+GameWin.*playerHp*+"\n");  
 writer.write("playerLevel = "+GameWin.*playerLevel*+"\n");  
 writer.write("playerAttack = "+GameWin.*playerAttack*+"\n");  
 writer.write("playerBulletCd = "+GameWin.*playerBulletCd*+"\n");

…  
 // 关闭writer对象  
 writer.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
}

public static void load() {  
  
 try (BufferedReader reader = new BufferedReader(new FileReader(*filePath*))) {  
 int intData;  
 double doubleData;  
 String line;  
  
 reader.readLine(); //吃掉@player  
  
 line = reader.readLine();  
 String[] tokens = line.split(" ");  
 GameWin.*playerName* = tokens[2];  
 line = reader.readLine();  
 tokens = line.split(" ");  
 doubleData = Double.*parseDouble*(tokens[2]);  
 GameWin.*playerX* = doubleData;  
 line = reader.readLine();  
 tokens = line.split(" ");  
 doubleData = Double.*parseDouble*(tokens[2]);  
 GameWin.*playerY* = doubleData;  
 line = reader.readLine();  
 tokens = line.split(" ");  
 intData = Integer.*parseInt*(tokens[2]);  
 GameWin.*playerHp* = intData;  
 line = reader.readLine();  
 tokens = line.split(" ");  
 intData = Integer.*parseInt*(tokens[2]);  
 GameWin.*playerLevel* = intData;  
 line = reader.readLine();  
 tokens = line.split(" ");  
 intData = Integer.*parseInt*(tokens[2]);  
 GameWin.*playerAttack* = intData;  
 line = reader.readLine();  
 tokens = line.split(" ");  
 intData = Integer.*parseInt*(tokens[2]);  
 GameWin.*playerBulletCd* = intData;

…

} catch (IOException ex) {  
 throw new RuntimeException(ex);  
 }  
}

6.排行榜

创建了ScoreList的类，用来专门执行排行榜功能，将排行榜数据保存在scoreList.txt中，利用文件读写相关函数来实现，与游戏存读档不同的地方在于，排行榜需要实现每次游戏结束时加入玩家成绩进行排行。本项目借助了List<Map.Entry<String, Integer>>完成。

public static void loadScoreList() {  
 Pattern pattern = Pattern.*compile*("^\\s\*(.\*?)\\s\*(\\d+)\\s\*$");  
 int No=1;  
 try (BufferedReader reader = new BufferedReader(new FileReader(*filePath*))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 Matcher matcher = pattern.matcher(line);  
 if(matcher.matches()) {  
 String[] tokens = line.split(" ");  
 String name = tokens[0];  
 int score = Integer.*parseInt*(tokens[1]);  
 GameWin.*scoreList*.put(name,score);  
 No++;  
 }  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 //GameWin.list=GameWin.scoreList;  
}  
public static void readScoreList(Graphics gImage) {  
 Pattern pattern = Pattern.*compile*("^\\s\*(.\*?)\\s\*(\\d+)\\s\*$");  
 int No=1;  
 //Pattern pattern = Pattern.compile("^\\s\*\"(.\*?)\",\\s\*(\\d+(?:\\.\\d+)?)\\s\*$");  
 try (BufferedReader reader = new BufferedReader(new FileReader(*filePath*))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 Matcher matcher = pattern.matcher(line);  
 if(matcher.matches()) {  
 String[] tokens = line.split(" ");  
 String name = tokens[0];  
 int score = Integer.*parseInt*(tokens[1]);  
 GameUtils.*drawWord*(gImage,name,Color.*white*,30,150,146+64\*No);  
 GameUtils.*drawWord*(gImage,""+score,Color.*white*,30,370,146+64\*No);  
 No++;  
 }  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
}  
public static void saveScoreList() {  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(*filePath*))) {  
 // 写入数据到文件  
 writer.write("@name @score\n");  
 int count = 0;  
 for (Map.Entry<String, Integer> entry : GameWin.*list*) {  
 if (count < 10) {  
 writer.write(entry.getKey()+" "+entry.getValue()+"\n");  
 count++;  
 } else {  
 break;  
 }  
 }  
 // 关闭writer对象  
 writer.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
}

7.多种子弹

四种玩家子弹，具有不同的攻击速度和攻击伤害，且分为三个等级，创建了十二个玩家子弹类。三种敌机子弹，三种BOSS子弹。一共创建了18个bullet相关类。其中不同规则的子弹使用相关数学公式实现，比如BOSS的螺旋子弹，使用了sin和cos函数实现了一个圆形区域的子弹。

if(GameWin.*gameState*==2) {  
 if(bulletNum>=0 && bulletNum<=5){  
 ySpeed=r\*Math.*cos*((Math.*PI*/2)\*((double)(bulletNum)/6));  
 xSpeed=Math.*sqrt*(r\*r-ySpeed\*ySpeed);  
 y+=ySpeed;  
 x-=xSpeed;  
 }else if(bulletNum>=6 && bulletNum<=11){  
 ySpeed=r\*Math.*sin*((Math.*PI*/2)\*((double)(bulletNum%6)/6));  
 xSpeed=Math.*sqrt*(r\*r-ySpeed\*ySpeed);  
 y-=ySpeed;  
 x-=xSpeed;  
 }else if(bulletNum>=12 && bulletNum<=17){  
 ySpeed=r\*Math.*cos*((Math.*PI*/2)\*((double)(bulletNum%12)/6));  
 xSpeed=Math.*sqrt*(r\*r-ySpeed\*ySpeed);  
 y-=ySpeed;  
 x+=xSpeed;  
 }  
 else if(bulletNum>=18 && bulletNum<=24){  
 ySpeed=r\*Math.*sin*((Math.*PI*/2)\*((double)(bulletNum%18)/6));  
 xSpeed=Math.*sqrt*(r\*r-ySpeed\*ySpeed);  
 y+=ySpeed;  
 x+=xSpeed;  
 }  
}  
if(y>gameHeight){  
 GameUtils.*removeObjList*.add(this);  
}

8.多种敌机

一共五个敌机和三个BOSS，创建了8个相关类，不同难度下的敌机和BOSS都不完全相同，而且各种敌机各有其特色。

if(GameWin.*gameState*==2) {  
 y += speed;  
 if(y>gameHeight){  
 GameUtils.*removeObjList*.add(this);  
 }  
 if (this.hp<=0 || GameWin.*boom*==1) {  
 MusicTool.*playExplodeMusic*();  
 ExplosionSmallObj explosionSmallObj = new ExplosionSmallObj(x, y);  
 GameUtils.*explosionSmallObjList*.add(explosionSmallObj);  
 GameUtils.*removeObjList*.add(explosionSmallObj);  
 if(Math.*random*()<0.05){ //百分之五的概率掉落道具  
 GameWin.*ifCreatSupplyDrop*=1;  
 GameWin.*supplyDropX*=x;  
 if(y>=0) {  
 GameWin.*supplyDropY* = y;  
 }else{  
 GameWin.*supplyDropY* = 1;  
 }  
 GameWin.*supplyDropNum*=(int)((Math.*random*()+0.25)\*4);  
 }  
 this.setX(-1000);  
 this.setY(-1000);  
 GameUtils.*removeObjList*.add(this);  
 GameWin.*score*+=1;  
 }

…

if (x < width \* 0.2) {  
 x += speed;  
} else if (x > gameWidth - 1.2 \* width) {  
 x -= speed;  
}  
if (this.hp<=0 || GameWin.*boom*==1) {  
 MusicTool.*playExplodeMusic*();  
 ExplosionMiddleObj explosionMiddleObj = new ExplosionMiddleObj(x, y);  
 GameUtils.*explosionMiddleObjList*.add(explosionMiddleObj);  
 GameUtils.*removeObjList*.add(explosionMiddleObj);  
 if(Math.*random*()<0.05){ //百分之五的概率掉落道具  
 GameWin.*ifCreatSupplyDrop*=1;  
 GameWin.*supplyDropX*=x;  
 if(y>=0) {  
 GameWin.*supplyDropY* = y;  
 }else{  
 GameWin.*supplyDropY* = 1;  
 }  
 GameWin.*supplyDropNum*=(int)((Math.*random*()+0.25)\*4);  
 }  
 this.setX(-1000);  
 this.setY(-1000);  
 GameUtils.*removeObjList*.add(this);  
 GameWin.*score*+=50;  
}

9.多种玩家飞机选择

有四种玩家飞机可供选择，第一个飞机拥有适中的攻击速度和攻击伤害，第二个飞机攻速快但伤害低，第三个飞机攻速慢但伤害高，第四个为激光战机，拥有子弹穿透的特殊机制。创建了四个plane相关类。

if(GameWin.*gameState*==1){  
 if(GameWin.*chosePlane*==1){  
 this.x=253.5-55;  
 this.y=571;  
 }else{  
 this.x=-2000;  
 this.y=-2000;  
 }  
}  
if(GameWin.*gameState*==2 && GameWin.*chosePlane*==1) {  
 if (GameWin.*invincibility*[0]==0 || GameWin.*invincibility*[0]==2) { //如果不闪烁  
 this.x = GameWin.*playerX* - 55;  
 this.y = GameWin.*playerY* - 60;  
 }else{  
 if(GameWin.*invincibility*[1] == 0){ //开始闪烁  
 this.x = GameWin.*playerX* - 55;  
 this.y = GameWin.*playerY* - 60;  
 if (GameWin.*TimerZero*(100, 0, 1)) {  
 GameWin.*invincibility*[1] = 1;  
 this.x = -2000;  
 this.y = -2000;  
 }  
 }else if(GameWin.*TimerZero*(100, 0, 1)){  
 GameWin.*invincibility*[1] = 0;  
 }  
 }

…

10.子弹威力升级系统

创建了playerLevel变量来控制玩家等级，分别在得分到达100和500时会自动升级子弹，攻击速度和攻击伤害会随等级升高而提升，并且相应的子弹图片也会不同。

//飞机升级  
if(score>=100 && playerLevel==1) {  
 playerLevel=2;  
 playerAttack\*=2;  
 playerBulletCd/=2;  
}  
if(score>=500 && playerLevel==2){  
 playerLevel=3;  
 playerAttack\*=2;  
 playerBulletCd/=2;  
}

11.炸弹功能

创建了玩家可以捡拾的炸弹，按下空格后会自动发射，爆炸后会清除场上的敌机。

if(*supply*[1]==1){ //0为获得炸弹,1时创建对象,2时可发射炸弹,3炸弹发射  
 GameUtils.*supply1ObjList*.add(new Supply1Obj(GameUtils.*supply1Img*,24,75,0,52,this));  
 *supply*[1]=2;  
}

12.道具功能

创建了四种道具及其相关的supply类，分别是炸弹、狂暴、回血和护盾，道具会在敌机爆炸或者攻击BOSS时随机掉落并移动，撞到屏幕后还会反弹，玩家在捡到道具后就可以使用。

if(GameWin.*gameState*==2) {  
 x += xSpeed;  
 y += ySpeed;  
 if (x <= 0 || x >= gameWidth - width) {  
 xSpeed \*= -1;  
 }  
 if (y <= 0 || y >= gameHeight - height) {  
 ySpeed \*= -1;  
 }  
 if (y < 0) {  
 y = 1;  
 }  
}

13.游戏难易程度调节

在选择飞机界面可以自由选择游戏难度，不同难度有不同的地图、敌机以及BOSS等。

//选择难度  
if (*difficultyState* == 1) {  
 *difficulty* = 1;  
} else if (*difficultyState* == 2) {  
 *difficulty* = 2;  
} else if (*difficultyState* == 3) {  
 *difficulty* = 3;  
}

…

if(*difficultyState*==1 || *difficulty*==1){  
 gImage.drawImage(GameUtils.*difficulty12Img*, 343, 200, null);  
}  
if (*difficultyState*==2 || *difficulty*==2){  
 gImage.drawImage(GameUtils.*difficulty22Img*, 343, 306, null);  
}  
if (*difficultyState*==3 || *difficulty*==3){  
 gImage.drawImage(GameUtils.*difficulty32Img*, 343, 412, null);  
}

…

if (*difficulty* >= 2) {  
 if (Math.*random*()\*100>0 && Math.*random*()\*100<=3 || (*score*%100==0 && *score*!=0 && Math.*random*()<=0.5)){ //类型三的飞机  
 GameUtils.*enemy4ObjList*.add(new Enemy4Obj(GameUtils.*enemy4Img*, 150, 110, (Math.*random*() \* (*gameWidth*-150)), -110, this));  
 GameUtils.*gameObjList*.add(GameUtils.*enemy4ObjList*.get(GameUtils.*enemy4ObjList*.size() - 1));  
 }  
 if (*score* > 1000) { //到达一千分加大刷新高级敌机概率  
 if (Math.*random*()\*100>=0 && Math.*random*()\*100<=6 || (*score*%100==0 && *score*!=0 && Math.*random*()<=0.5)){ //类型三的飞机  
 GameUtils.*enemy4ObjList*.add(new Enemy4Obj(GameUtils.*enemy4Img*, 150, 110, (Math.*random*() \* (*gameWidth*-150)), -110, this));  
 GameUtils.*gameObjList*.add(GameUtils.*enemy4ObjList*.get(GameUtils.*enemy4ObjList*.size() - 1));  
 }  
 }

14.游戏音乐与音效

在菜单时会循环播放菜单的背景音乐，在战斗中也会播放战斗时音乐，还有各种点击、爆炸、捡到道具等的音效。创建了一个MusicTool类来实现相关功能。

public class MusicTool {  
 private static String *file* = "0";  
 static URL *battleUrl*;  
 static {  
 try {  
 *battleUrl* = new File("res/media/battle.wav").toURL();  
 } catch (MalformedURLException e) {  
 throw new RuntimeException(e);  
 }  
 }  
  
 static URL *bossUrl*;  
 static {  
 try {  
 *bossUrl* = new File("res/media/boss.wav").toURL();  
 } catch (MalformedURLException e) {  
 throw new RuntimeException(e);  
 }  
 }  
  
 static URL *clickUrl*;  
  
 static {  
 try {  
 *clickUrl* = new File("res/media/click.wav").toURL();  
 } catch (MalformedURLException e) {  
 throw new RuntimeException(e);  
 }  
 }

…  
  
 static AudioClip *menuAudioClip* = Applet.*newAudioClip*(*menuUrl*);  
 static AudioClip *clickAudioClip* = Applet.*newAudioClip*(*clickUrl*);  
 static AudioClip *goAudioClip* = Applet.*newAudioClip*(*goUrl*);  
 static AudioClip *battleAudioClip* = Applet.*newAudioClip*(*battleUrl*);

…

public static void playMenuMusic() {  
 *menuAudioClip*.loop();  
 }  
 public static void playClickMusic() {  
 *clickAudioClip*.play();  
 }  
 public static void playBattleMusic() {  
 *battleAudioClip*.loop();  
 }  
  
 *…*

public static void stopBattleMusic() {  
 *battleAudioClip*.stop();  
 }  
 public static void stopBossMusic() {  
 *bossAudioClip*.stop();  
 }  
}

15.BOSS战

每通过击落敌机来获得一千分时会根据所选难度生成不同的BOSS，且会播放不同的战斗音乐。创建了三个BOSS相关类和creatBoss函数以及creatBossBullet函数来实现该功能。

if(GameWin.*gameState*==2) {  
 if(GameWin.*bossIfLive*==1 && GameWin.*bossCome*==1){  
 hp = 500;  
 x = 253.5-175;  
 y = -288;  
 GameWin.*bossCome*=0;  
 }  
 if(GameWin.*bossIfLive*==1){  
 //血条  
/\* g.setColor(Color.white);  
 g.fillRect(gameWidth/2-100,50,200,10);\*/  
 g.setColor(Color.*red*);  
 g.fillRect(gameWidth/2-100,70,hp\*200/500,10);  
 }  
 if(y<=0) {  
 y += speed;  
 }  
 if (this.hp <= 0) {  
 MusicTool.*playExplodeMusic*();  
 ExplosionBigObj explosionBigObj = new ExplosionBigObj(x, y);  
 GameUtils.*explosionBigObjList*.add(explosionBigObj);  
 GameUtils.*removeObjList*.add(explosionBigObj);  
 this.setX(-1000);  
 this.setY(0);  
 this.hp=500;  
 GameWin.*bossIfLive*=0;  
 GameWin.*score*+=100;  
 MusicTool.*stopBossMusic*();  
 MusicTool.*playBattleMusic*();  
 }

…

void creatBossObj(){  
 if((*score*-*hitBossScore*)%1000>=950 && (*score*-*hitBossScore*)%1000<=1000 && *score*!=0 && *bossIfLive*==0){  
 MusicTool.*stopBattleMusic*();  
 MusicTool.*playWarningMusic*();  
 MusicTool.*playBossMusic*();  
 if(*difficulty*==1){  
 *bossCome*=1;  
 *bossWidth*=350;  
 *bossHeight*=288;  
 GameUtils.*gameObjList*.add(boss1Obj);  
 }else if(*difficulty*==2){  
 *bossCome*=2;  
 *bossWidth*=375;  
 *bossHeight*=304;  
 GameUtils.*gameObjList*.add(boss2Obj);  
 }else if(*difficulty*==3){  
 *bossCome*=3;  
 *bossWidth*=400;  
 *bossHeight*=374;  
 GameUtils.*gameObjList*.add(boss3Obj);  
 }  
 *bossIfLive*=1;  
 }  
 //创建BOSS子弹  
 if(TimerOne(10000, 0, 4) && *bossIfLive*==1)  
 {  
 if(*difficulty* == 1 && boss1Obj.getY()>=0){  
 *ifCreatBossBullet*=1;  
 }else if(*difficulty* == 2 && boss2Obj.getY()>=0){  
 if(Math.*random*()\*100 >= 0 && Math.*random*()\*100 <= 50) {  
 *ifCreatBossBullet*=1;  
 }  
 else{  
 *ifCreatBossBullet*=2;  
 }  
 }else if(*difficulty* == 3 && boss3Obj.getY()>=0){  
 if(Math.*random*()\*100 >= 0 && Math.*random*()\*100 <= 30){  
 *ifCreatBossBullet*=1;  
 }else if(Math.*random*()\*100 >= 0 && Math.*random*()\*100 <= 50) {  
 *ifCreatBossBullet*=2;  
 }  
 else{  
 *ifCreatBossBullet*=3;  
 }  
 }  
 }  
 if(*ifCreatBossBullet*==1 && TimerOne(800,0,5)) {  
 *bossBulletNum*++; //该BOSS子弹数量自增  
 if(*bossBulletNum*>=3){  
 *bossBulletNum*=0;  
 *ifCreatBossBullet*=0;  
 }  
 for(int i=0;i<(int)(Math.*random*()\*7)+3;i++){ //每次发射三到十发子弹  
 …  
 }  
 }  
 if(*ifCreatBossBullet*==2 && TimerOne(100,0,5)) {  
 *bossBulletNum*++; //该BOSS子弹数量自增  
 if(*bossBulletNum*>=16){  
 *bossBulletNum*=0;  
 *ifCreatBossBullet*=0;  
 }  
 double x = *gameWidth*/2 - 20; //减去子弹宽度的一半  
 double y = *bossHeight*; //修正位置  
 GameUtils.*bossBullet2ObjList*.add(new BossBullet2Obj(GameUtils.*bossBulletImg*, 40, 39, x, y, *bossBulletNum*, this));  
 GameUtils.*gameObjList*.add(GameUtils.*bossBullet2ObjList*.get(GameUtils.*bossBullet2ObjList*.size() - 1));  
 }  
 if(*ifCreatBossBullet*==3 && TimerOne(100,0,5)) {  
 *bossBulletNum*++; //该BOSS子弹数量自增  
 if(*bossBulletNum*>=48){  
 *bossBulletNum*=0;  
 *ifCreatBossBullet*=0;  
 }  
 double x = *gameWidth*/2 - 25; //修正位置  
 double y = *bossHeight*-6;  
 …  
 }  
}

1. 功能展示与说明
2. 游戏菜单

进入游戏首先映入眼帘的是游戏菜单，可以选择开始游戏进入飞机选择界面，选择继续游戏继续上次存档，选择排行榜查看游戏排行榜，选择关于查看游戏说明。



1. 选择飞机

这里有四种飞机可以选择，会自动演示飞机子弹，还要选择难度才能开始游戏。



1. 游戏中

飞机自动射出子弹，鼠标控制飞机移动，捡到炸弹道具后空格可以发射炸弹，在游戏中按下Esc键可以暂停。



1. 暂停游戏

按下Esc键进入暂停界面，游戏元素都停止活动，在这里可以选择保存游戏或者返回菜单界面。



1. 游戏结束

玩家被击毁后进入游戏结束界面，自动统计玩家本次游戏得分并按顺序插入排行榜。



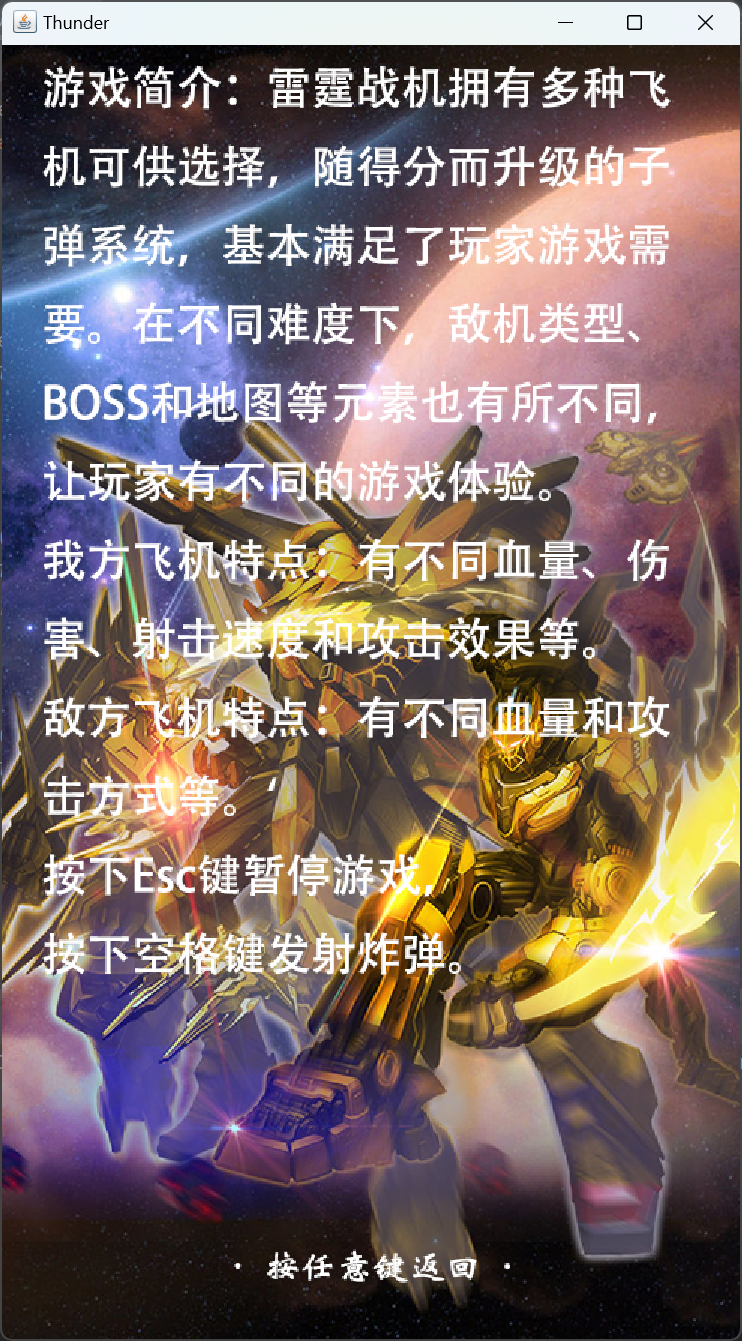
1. 排行榜

按顺序输出排行榜玩家昵称及其分数，按下任意键可以返回菜单。



1. 关于

这里是游戏内容简介与游戏操作说明，按下任意键可以返回菜单。



四、设计心得与体会

1. 设计过程中的问题及其解决方案

（1）透明贴图问题。刚开始用的掩码图，每一张素材都需要自己PS掩码图，做起来繁琐且实际使用效果不好。在网上大量搜索资料之后找到了一套完整合适的透明图片素材及游戏音乐音效，解决了此问题。

（2）子弹创建位置的问题。使用相关数据用位置加上二分之一宽度即可。

（3）子弹包括种类、等级和编号三个特征使用二维数组无法解决的问题。将每个不同的子弹都各自创建了一个类解决了该问题。

（4）玩家无敌时间闪烁问题。该问题的解决需要设置闪烁时长及闪烁状态，解决方案为定义了invincibility[2]数组invincibility[0]控制闪烁时长，invincibility[1]控制闪烁。

（5）循环播放的音乐关闭不掉的问题。后来经过检查发现是每次调用音乐类的时候都为其创建了一个新的对象，导致无法关闭之前循环播放的音乐的对象，将每个音乐对象先创建好，每次调用时直接调用解决了这个问题。

（6）输入昵称不显示画面的问题。将输入昵称调节到跳转行下面解决了该问题。

（7）游戏速度过快的问题。改变了绘制帧数解决了该问题。（10）有的事件需要第一次调用就触发如鼠标点击判定而有的事件不能第一次调用触发且都用的一个定时器的问题。解决方案为设置两个定时器，一个TimerZero()和TimerOne()分别处理这两种情况。

（11）敌机子弹过于密集玩家不便躲避且疏散敌机子弹或者减小玩家贴图都会造成游戏不美观的问题。将玩家判定区域缩小解决了这个问题。

（12）道具掉落在屏幕之外的错误。产生原因是敌机还未完全出现在屏幕中便被击落，而产生的道具位置为敌机位置。加上一个判断敌机是否完全出现在屏幕中，若未完全出现则产生道具的纵坐标为0得以解决该错误。

（13）特殊敌机刷新条件为gamer.score % 100 == 0 导致游戏刚开始就刷新且不断刷新的错误。产生原因是刚开始分数就为0，再补充上gamer.score != 0得以解决改错误。

（14）BOSS子弹三的圆周不匀称的错误。创建子弹时，子弹y轴速度成等差数列，但圆周周长却不成等差数列导致。用ySpeed=r\*Math.sin((Math.PI/2)\*((double)(bulletNum%18)/6))等数学公式解决了该错误。

（15）游戏暂停时双方虽然不会动但是子弹还在创建的错误。产生原因是游戏暂停的函数中调用了相关子弹产生函数，删除得以解决错误。

（16）激光战机子弹一碰敌机就死亡的错误。产生原因是激光一直存在，而击中判定是是否有交集，即没有判定CD，导致伤害超标问题，加上判定CD即为激光战机射击速度解决了该错误。

（17）刚刚打完BOSS又产生一个BOSS的错误。产生原因是BOSS创建条件为玩家每获得一千分创建BOSS而击中BOSS也能获得积分，导致刚打完又产生了BOSS。设置一个hitbossScore变量，用来记录玩家打BOSS的得分，判定产生BOSS时用玩家积分减去打BOSS的积分解决该问题。

（18）狂暴或者护盾第二次获得时持续时间为0的错误。产生原因为用的TimerOne()函数判定是否到一定时间使道具效果消失，但是未获得道具时持续时间也会流逝，所以设计了每次捡到道具都会初始化其Init数组，解决了这个错误。

（19）鼠标只点击了一次，界面跳转好几次的错误。产生原因是鼠标按下有短暂的持续时间，而这个判定没有间隔，错把一次点击判定为好几次，使用TimeZero()函数，为每次鼠标点击判定加上了一个间隔得以解决该错误。

1. 体会和收获

在本次课程设计作业中，我体会到了完成一个游戏制作过程的复杂和困难。从最开始一个空白窗口到后面多彩的界面，我感触颇深，对最后运行的结果比较满意，感觉这一个月以来的努力没有白费。我也在这次制作过程中获得了很多收获。我编程能力获得了极大的提升，制作游戏需要有较强的逻辑思维和动手能力，由于游戏包含内容多，更加综合，需要更加细致和谨慎，同时也需要学习一些新的东西所以我通过制作游戏，锻炼了自己的编程能力，切实地提升我的代码实现能力，此外，制作这样的大程序，代码规范是必不可少的，经过此次游戏设计之后，我的代码也更加整洁规范。而且我还提高了我的创造力，制作游戏需要有独立思考和创新能力，需要去探索新的玩法、新的内容，比如我的雷霆战机地子弹规则，虽然之前的游戏也有过类似的设计，但是没有相关的代码可供参考，我完全通过自己的思考创作出了这些子弹规则，很有成就感。并且我也熟悉游戏开发流程，在制作游戏的过程中，需要遵循一系列的开发流程和规范，比如需求分析、设计、开发、测试等等。我自己一个人亲身经历了这些步骤，体会到了其中的困难与快乐。总之，制作JAVA游戏是一个有趣的、具有挑战性和充满收获的过程，可以提高自己的编程能力，同时也可以让自己学习更多新的知识，在收获过程中充实并快乐自我。