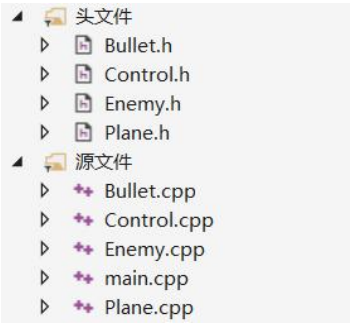
课程设计报告

一、选题简介

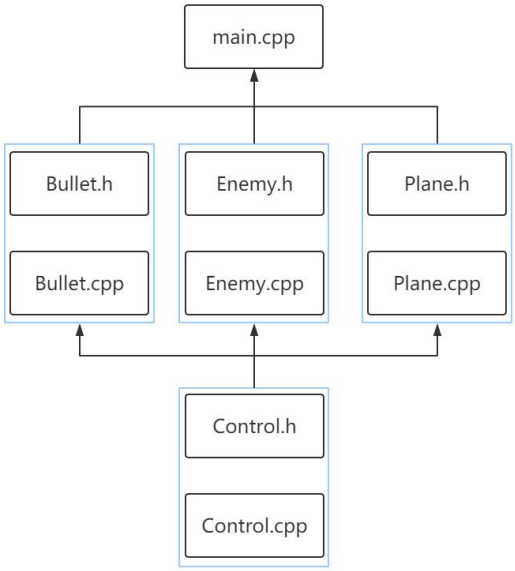
本次课程设计实现了一个基于控制台的飞机大战小游戏，可以根据键盘输入与游戏程序进行交互，如操控战机移动，发射武器等

二、文件划分

1. 文件清单



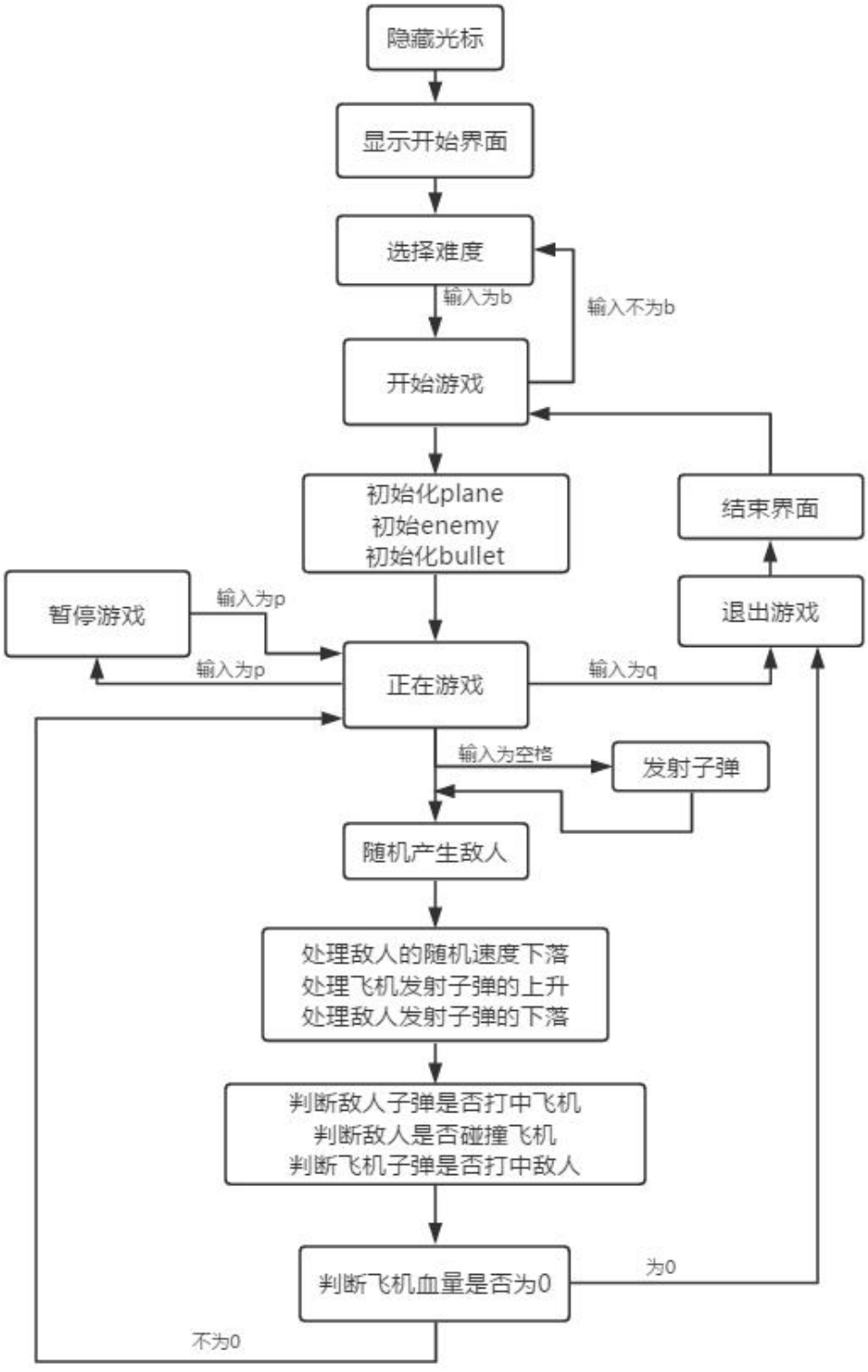
2. 文件关联



3. 内容划分

Plane　包含飞机类 -> 即玩家操纵的飞机
Enemy　包含敌人类
Bullet　包含两个子弹类 -> plane/enemy 发射的子弹
Control 包含控制类函数，如光标、随机数的产生函数等
main　主函数 -> 包含对键盘的监听、对碰撞的检测等

三、设计思路


隐藏光标
显示开始界面
选择难度
输入为b
输入不为b
开始游戏
初始化plane
初始enemy
初始化bullet
结束界面
暂停游戏
输入为p
退出游戏
输入为p
正在游戏
输入为q
输入为空格
发射子弹
随机产生敌人
处理敌人的随机速度下落
处理飞机发射子弹的上升
处理敌人发射子弹的下落
判断敌人子弹是否打中飞机
判断敌人是否碰撞飞机
判断飞机子弹是否打中敌人
判断飞机血量是否为0
为0
不为0

四、类的设计

1. 飞机类 class Plane

• 主要设计

注释中有对成员变量及成员函数的解释

```
class Plane {
private:
    int x_min;          //the border for the plane center tomove
    int x_max;
    int y_min;
    int y_max;
    int x;              //my plane's middle coordinate
    int y;
    int score;          //record the scores
    int blood;          //blood volumn
    int level;          //0 means easy, 1 means difficult
    int enemy_nums;     //max enemy_nums in the screen, easy->6, difficulty->12
public:
    Plane();                                //constrctor set the border for the plane center
    void Init_Plane();                      //init the plane
    void Draw_Plane();                      //draw my plane,(x,y) is the middle pos
    void Clean_Plane();                     //clean the cur plane
    void Move_Plane(char c);                //when input 'wasd' then move
    void Get_Pos(int &pos_x, int &pos_y);   //get the cur pos of the plane to(pos_x, pos_y)
    void Add_Score();                       //if bullet hit the enemy, score++
    int Get_Score();                        //an api for getting the player's scores
    void Decrease_Blood();                  //when collison, decrease the blood
    bool Check_die();                       //if blood==0 then die
    int Get_Blood();                        //an api for getting the player's scores
    int Get_Level();                        //an api for getting the player's level
    void Set_Level(int l);                  //set the level=l
    int Get_EnemyNum();                     //an api for getting the player's enemy nums
    void Set_EnemyNum(int l);               //set the num=l
};
```

• 主要功能

通过类记录飞机的坐标、得分、血量等，完成飞机的上下左右移动、得分的记录、绘制飞机图形、清除飞机图形等

2. 敌人类 class Enemy

• 主要设计

注释中有对成员变量及成员函数的解释

```
class Enemy {
private:
    int x;                  //the enemy's coordinate, if y==-1,invalid
    int y;
    int x_min;              //border condition
    int x_max;
    int y_min;
    int y_max;
    int move_interval;      //when up to move_interval then move
    int cnt_interval;       //to cnt for interval
public:
    Enemy();                                    //constructor set the border condition
    void Init_Enemy(int level);                 //init the enemy
    void Draw_Enemy();                          //draw the enemy
    void Clean_Enemy();                         //clean the enemy
    void Move_Down();                           //move the enemy down
    bool Enemy_Valid();                         //whether the enemy is valid
    bool Check_Move();                          //to implement different enemy move speed
    void Get_Pos(int &pos_x, int &pos_y);       //get the pos of enemy
    void Destroy_Enemy();                       //the enemy meet the plane or bullet, has destroyed
};
```

• 主要功能

记录敌人的坐标等，完成敌人的移动、绘制图形、消灭敌人等

• 随机速度

成员变量中的 move_interval 根据不同的难度设置为不同的值，每当 cnt_interval 达到 move_interval 时，Check_Move 函数返回 true；否则 cnt_interval++，返回 false。因而，可以实现每个敌人的速度都是根据难度设置的随机移动速度

```
if (level == 0) {   //easy->slow
    move_interval = Rand_Num(10000, 160000);
}
else {          //difficult->fast
    move_interval = Rand_Num(4400, 16000);
}
```

3. 子弹类 class Bullet / class Enemy_Bullet

①class Bullet

• 主要设计

注释中有对成员变量及成员函数的解释

```
class Bullet {
private:
    int x;              //the bullet's coordinate,y=-1 means no existing
    int y;
    int y_min;          //the min y or the bullet will display
    int bullet_speed;   //to control the speed of the bullet
public:
    Bullet();                                       //constructor
    void Init_Bullet(int pos_x, int pos_y);         //init the bullet,(x,y) is the center of plane
    void Clean_Bullet();                            //clean the bullet
    void Draw_Bullet();                             //draw the bullet
    void Bullet_Up();                               //Move the bullet up
    void Bullet_Down();                             //Move the bullet down
    bool Bullet_Valid();                            //whether bullet is valid
    void Get_Pos(int &pos_x, int &pos_y);           //get the pos of bullet
    void Destroy_Bullet();                          //if bullet meets the enemy, then destroy
    int Get_Speed();                                //get the intervals of bullet move speed
};
```

• 具体实现

通过记录每个子弹的坐标和速度控制变量，实现子弹的自由下落、或者上升、绘制图形、销毁子弹等

②class Enemy_Bullet

• 主要设计

注释中有对成员变量及成员函数的解释

```
class Enemy_Bullet:public Bullet {
private:
    int y_max;          //the max y or the bullet will disappear
public:
    void Init_EnemyBullet(int pos_x, int pos_y);    //init the enemy bullet
    void EnemyBullet_Down();                        //enemy bullet will move down
};
```

• 类的继承

Enemy_Bullet 继承了 Bullet 类，Bullet 为飞机 plane 发射的子弹，而 Enemy_Bullet 为敌人 enemy 发射的子弹，默认每个 Enemy 在出现时发射子弹，并且要控制子弹速度大于敌人的移动速度。

其中 Bullet 类中子弹向上移动，而 Enemy_Bullet 类中子弹向下移动，并且初始位置是在敌人的下方，故实现了如上两个成员函数。

五、其他实现

1. Control 中

```
void Set_Cursor(int x, int y);                    //set cursor to (x,y)
void Hide_Cursor();                               //hide the cursor
void Display_Cursor();                            //display the cursor
int Rand_Num(int head, int tail);                 //produce rand num
void Draw_Row(int x1, int x2, int y, char c);     //draw a row, from(x1,y)->(x2,y),with c
void Draw_Column(int x, int y1, int y2, char c);  //draw a column, from (x,y1)->(x,y2),with c
void Press_Wait(char press);                      //waitfor press down the keboard "press"
void Game_Cover();                                //the cover of the game
void Game_Begin();                                //when enter the space,the game begin
void Draw_Outline(int level);                     //draw the background of the game
void Set_Score(int score);                        //set the cur scores
void Game_Over(int score);                        //the cover of the game over
void Set_Suspend();                               //set state to suspend
void Set_Continue();                              //set state to suspend
void Set_Blood(int blood);                        //set the cur blood
```

- 通过<windows.h>和<conio.h>库调用，实现对界面光标的操作
- 通过 kbhit()和 getch()实现对键盘的监听
- 通过各个类提供的 API 获取类的信息，从而在屏幕上打印出相关信息

2. main 中

```
//find one unuse bullet to shoot, the center of plane is(x,y)
void Shoot_Bullet(Bullet *bullet,int x,int y){ ... }

//bullet will up by nature
void Handle_Bullet(Bullet *bullet,int &bullet_speed_cnt){ ... }

//enemy bullet will down by nature
void Handle_EnemyBullet(Plane &plane,Enemy_Bullet *bullet, int &bullet_speed_cnt){ ... }

//every enemy_produce_time produce enemy, and the time is changing
void Produce_Enemy(Plane &plane, Enemy *enemy, int &time_interval, int &enemy_produce_time,Enemy_Bullet*enemy_bullet){ ... }

//enemy will down by nature
void Handle_Enemy(Plane &plane,Enemy *enemy){ ... }

//check one plane and one enemy whether is overlap, 1 means overlap
bool Check_Plane_Enemy(Plane *plane, Enemy *enemy){ ... }

//check enemy bullet whether is overlap my plane, 1means overlap
bool Check_Bullet_Plane(Enemy_Bullet *bullet, Plane *plane){ ... }

//if plane and enemy overlap, handle it, return -1 and enemy bullet hit my plane
int Handle_Plane_Enemy(Plane *plane, Enemy *enemy,Enemy_Bullet*enemy_bullet){ ... }

//check one bullet and one enemy whether is overlap, 1 means overlap
bool Check_Bullet_Enemy(Bullet *bullet, Enemy *enemy){ ... }

//if bullet and enemy overlap, handle it, return -1
void Handle_Bullet_Enemy(Bullet *bullet, Enemy *enemy,Plane &plane){ ... }

//when playing normal, -> wasd and space to control, 1->suspend 2->lose 2->quit_game
int Keyboard_Control(Plane &plane,Bullet *bullet,Enemy *enemy,Enemy_Bullet*enemy_bullet){ ... }

//choose the difficuly of the game
void Choose_Level(Plane &plane){ ... }
```

- 实现子弹、敌人等的自由下落
- 通过获取坐标检查敌人、子弹、飞机之间是否发生了碰撞
- Produce_Enemy 通过类似于计数的方式实现随机间隔时间的产生敌人

3. main 中

Plane plane;

Bullet *bullet = new Bullet[BULLET_MAX];

Enemy *enemy = new Enemy[plane.Get_EnemyNum()];

Enemy_Bullet *enemy_bullet = new Enemy_Bullet[plane.Get_EnemyNum()];

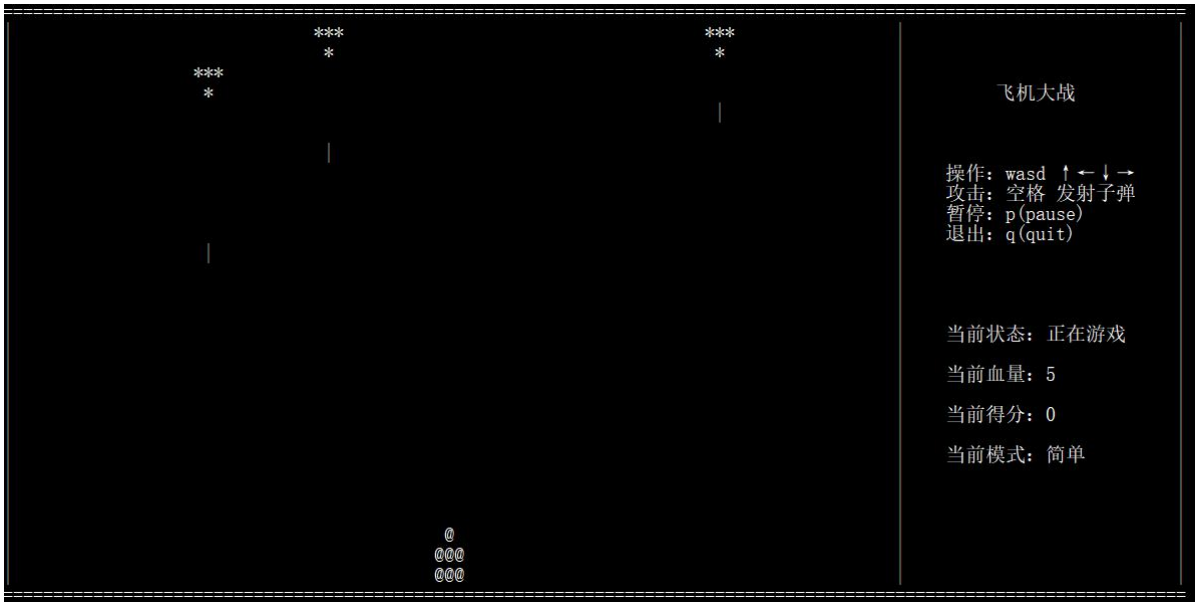
以上为类的实例化，利用实例化的对象实现小游戏

六、功能实现

1. 开始界面

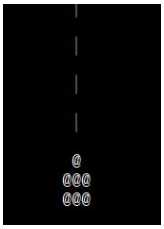
2. 难度选择

3. 游戏界面



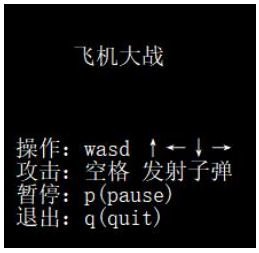
4. 游戏设定

①plane 由玩家操控，可以通过按 wasd 上下左右移动，按空格键发送子弹



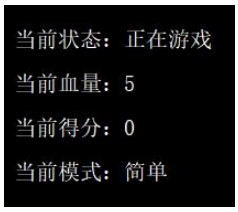
②enemy 由系统自动控制，随机时间产生，随机移动速度，会发射子弹攻击 plane

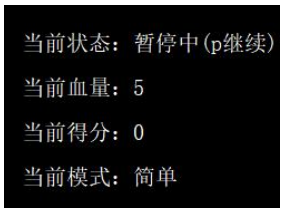
③提示区域



④状态切换

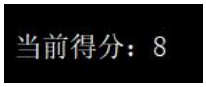
暂停游戏状态/正常游戏状态/退出





⑤得分计算

每次 plane 发射的子弹击中 enemy，得 1 分



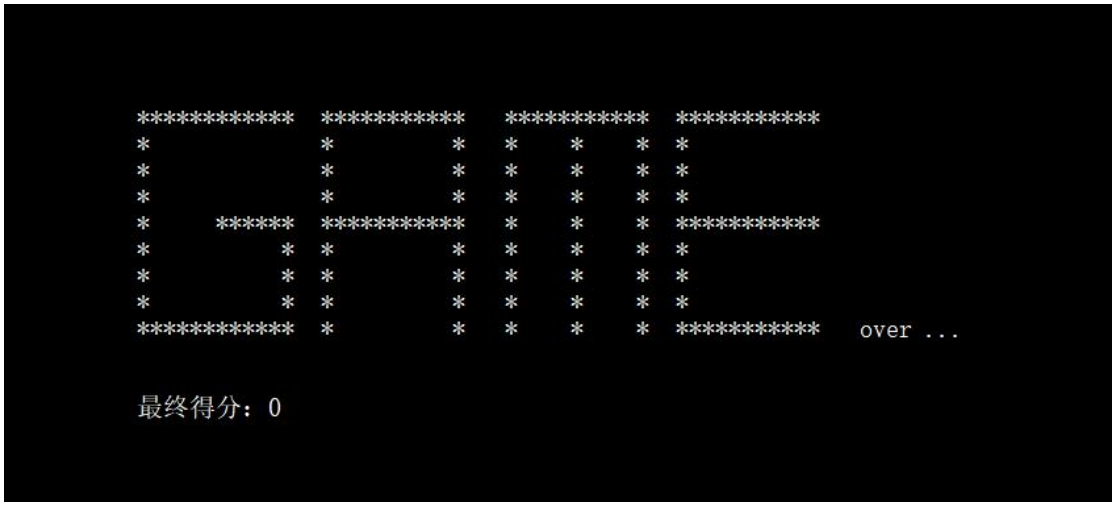
⑥血量计算

初始血量设置为了 5

每次 plane 与 enemy 碰撞，血量减 1，直至为 0 失败

每次 plane 被 enemy 发射的子弹击中，血量减 1，直至为 0 失败

5. 游戏结束



七、实验小结

1. 增强了应用知识的实践能力
2. 学习熟练使用面向对象设计的设计方法
3. 进一步理解了封装与抽象

注：源代码附在压缩包中，代码中附有详细注释