Predator V.S. Aliens

- SHOOT

Background

Our hero is from a peaceful and beatific village. However, it was shocked down by a group of aliens, they were aiming to eat villager. The hero protected villagers to back to the tower, his/her must shoot aliens and protect the tower far away from aliens!

The situation was urgent, aliens can beat, shoot, boom and even hold the hero. But after updated, hero can also quintic shoot, jump higher or in rampage mode.

Operation

Press "UP" to jump.

Press "LEFT" or "RIGHT" to move left or right, this can also change the direction of shooting.

Keep pressing "SPACE" to increase shooting strength, and release to shoot. But you can't always pressing, it will cause the string bounce out of hand.

Press "W" or "S" to raise or lower the angle of shooting. It can be operated whether you press space or not.

Extra skill: after updated "jump higher", you can press "SHIFT" to change the mode of jump - higher or lower.

Extra skill: after updated "more arrows", you can press "A" to shoot quintic(5) arrows in one time. But you need to wait the blue bar full.

You can pause by pressing "P" in the game.

You can exit game at any time by pressing "ESC".

Characters

Hero

Movement

Left and right movement is allowed, but if you impact the left and right walls at high speed, you will suffer inelastic collision. You can jump on the heads of aliens as well, then you get a huge gravitational potential energy.

Shoot

When shooting, you can adjust the strength of the angle, the picture of the bow changed accordingly, and the parabolic trajectory is drew of the bow and arrow, which you can predict. After shooting, the bow and arrow move in a parabola. When they collide with the aliens, the momentum is calculated and the damage is determined by the momentum. The darker the bow, the greater the momentum.

Quintic shoot

Quitic shoot let you shoot 5 arrows at once, but this skill has a cool down time. When you face a large number of aliens, it helps a lot!

Jump higher

Try to jump higher (even out of screen!) and shoot, then the arrow will damage a lot for the high momentum. It may be a good choice to stand on Slimers(see table of Aliens) for they will never fly out of screen.

Rampage mode

It's a one-time skill. Every time you spend points on it many times, you will gain triple speed and triple power in a short time.

Score

Beat an alien with the least number of shots to get a higher score. You can compare with your partner's score, and you can also see your ranking in "log/history.txt".

Aliens

	Static	Never move, never shoot. Waiting to be shot. They only appear in the easy mode.
	Horizontal	Moving horizontally, and sometimes shoot at the tower.
	Guard	Guard is a kind of normal "horizontal", but it can also follow the king up and down and guard the king.
	Slimer	Slimer likes to move to and control the hero. When it comes into contact with the hero, the hero can't move for a certain period of time, and it will slowly pass until it dies. It does not attack the tower.
	Dinosaur	Dinosaur is a kind of mild and warm alien. It protects other aliens from attack with their huge body and it's life value is big. But it can only hit the tower through its head after it is close to tower.
	Pink	Pink explodes on the ground and blasts the hero a certain distance. He can only move vertically, and very quickly, has very short health, and will die after the explosion.
	King	King is the most powerful alien who only appears when all of other aliens are dead in the difficult mode. It can not only shoot at the tower remotely, but also squash the tower and drop bombs after get closed to it. What's more, when it goes up high, it can release Slimers and Pinks. So never let it get close to your tower, its close range damage is devastating!

Codes

1. Trajectory as rediction of arrow track
   a) Get x position by $v_0 * t$
   b) Get y position by $g*t^2$

```
public void trajectory(GameArena panel) {
    if (holdStrength * holdAngle > 0) {
        int x = this.getX();
        double strength = holdStrength * 10;
        double gravity = -0.5;
        double SX = strength * Math.cos(holdAngle);
        double SY = -strength * Math.sin(holdAngle);
        int time = (int) (2 * SY / gravity);
        time += (int)Math.sqrt(-2*(panel.getHeight()-getY())/gravity);
        for (int i = 0; i <= time; i++) {
            Dot newDot = new Dot(panel, (int) (x + SX * i), (int) (gravity * i * i / 2
- SY * i) - this.getY() + panel.getHeight());
            newDot.setY(panel.getHeight() - newDot.getY());
        }
    }
}
```

2. Reset ground when there is an alien below the player

```
if ((player.getY() <= item.getY() - 20) & (Math.abs(player.getX() - item.getX()) < 30)
        & (getHeight() - item.getY() + 50 > resetGround)) {
    resetGround = getHeight() - item.getY() + 50;
}
```

3. Make the background rolling

```
public void paint(Graphics g) {
    super.paint(g);
    if (dynamicBackgroundFlag) {
        backGroundSpeed -= 1;
        titleIndex += 0.25;
        if (titleIndex >= 10) {
            titleIndex = 1;
        }
    }
    try {
        TL = ImageIO.read(new File("img/title/title" + (int) titleIndex + ".png"));
    } catch (IOException e) {
        e.printStackTrace();
    }
    // The following is how to achieve dynamic background
```

```
    // Scroll two identical pictures back and forth
    g.drawImage(background, backGroundSpeed, 0, screenSize.width, screenSize.height,
null);
    g.drawImage(background, backGroundSpeed + screenSize.width, 0, screenSize.width,
screenSize.height, null);
    g.drawImage(TL, screenSize.width / 2 - 200, screenSize.height / 100 - 100, 400, 400,
null);
    if (backGroundSpeed <= -screenSize.width) {
        backGroundSpeed = 0;
    }
}
```

4. Draw the tail of arrow

```
if (item instanceof Arrow) {
    double width = ((Arrow) item).getLastX() - item.getX();
    double height = -((Arrow) item).getLastY() + item.getY();
    double alfa = Math.atan(width / height);
    int dy = (int) (50 * Math.cos(alfa));
    int dx = -(int) (50 * Math.sin(alfa));
    if (height < 0) {
        dy = -dy;
        dx = -dx;
    }
    Graphics2D g2 = (Graphics2D) g;
    g2.setStroke(new BasicStroke(3));
    int colorStrength = 2 * (int) ((Arrow) item).getSpeed();
    if (colorStrength > 254) {
        colorStrength = 254;
    }
    g2.setColor(new Color(255, 140, 0, colorStrength));
    int originX = ((Arrow) item).getLastX() - dx;
    int originY = ((Arrow) item).getLastY() - dy;
    g2.drawLine(originX, originY, item.getX(), item.getY()); // draw arrow
}
```

5. Control different types of alien to launch different types of rock.

```
/**
 * Store index of alien that enable to throw rocks.
 */
private ArrayList<int[]> indexCase = new ArrayList<>();
// Update the picture of the bomb dropped by king alien every 10 frames (produce
// animation effect).
if (item instanceof Rock && ((Rock) item).getMode() == 1) {
    String num = String.valueOf(time % 20 / 10);
```

```
        item.setSource("img/characters/boom" + num + ".png");
    }
    // Record the index of the alien that needs to launch rock.
    if (((Alien) item).getMoveMode() == 2) {
        int n = items.indexOf(item);
        int nx[] = new int[1];
        nx[0] = n;
        indexCase.add(nx);
    }
    if (time % 200 == 0) {
        for (int i = 0; i < indexCase.size(); i++) {
            new Rock(((Alien) items.get(indexCase.get(i)[0])), this);
        }
    }
    indexCase.clear();
    if (random.nextInt(30) == 1 && kingIndex > 0) {
        if (Math.abs(kingAlien.getX() - tower.getX()) < 20) {
            Rock boom = new Rock((Alien) items.get(kingIndex), this);
            boom.setSource("img/characters/boom0.png");
            boom.setSize(80);
            boom.setMode(1);
        }
    }
```

6. Earthquake effect.

```
    if (shaking > 0) {
        if (shaking > 3) {
            background.setY(background.getY() - 6);
        }
        background.setY(background.getY() + 3);
        shaking--;
    } else {
        background.setY(screenSize.height / 2);
    }
```

7. Explode the player
   a) Let the player get the opposite velocity in the directions X and y of the explosion.
   b) The closer the player is to the blast, the greater the resulting blast effect.

```
    if (Math.abs(this.getX() - player.getX()) < 100 & Math.abs(this.getY() - player.getY()) <
    50) {
        panel.player.setSpeedX(panel.player.getSpeedX() + (1000 - Math.abs(getX() -
    panel.player.getX()))
                * (getX() - panel.player.getX() > 0 ? -10 : 10));
        panel.player.setSpeedY(panel.player.getSpeedY() + (1000 - Math.abs(getY() -
```

```
panel.player.getY()));
}
```

8. Slimer move towards player

```
double alfa;
int distanceX = getX() - panel.player.getX();
int distanceY = -getY() + panel.player.getY();
if (distanceY != 0) {
    alfa = Math.atan(distanceX / distanceY);
} else {
    alfa = 0;
}
if (getSpeedX() > 2) {
    setSpeedX(getSpeedX() - 0.2);
} else {
    setSpeedX(2);
}
if (getSpeedY() > 2) {
    setSpeedY(getSpeedY() - 0.2);
} else {
    setSpeedY(2);
}
int dy = (int) (getSpeedY() * Math.cos(alfa));
int dx = -(int) (getSpeedX() * Math.sin(alfa));

if (distanceY < 0) {
    dy = -dy;
    dx = -dx;
}
if (distanceX * getWidth() < 0) {
    setWidth(-getWidth());
}
int toX = getX() + dx;
int toY = Math.max(getY() + dy, 100);
this.setX(toX);
this.setY(toY);
```

9. Different aliens have different judgements of being attacked

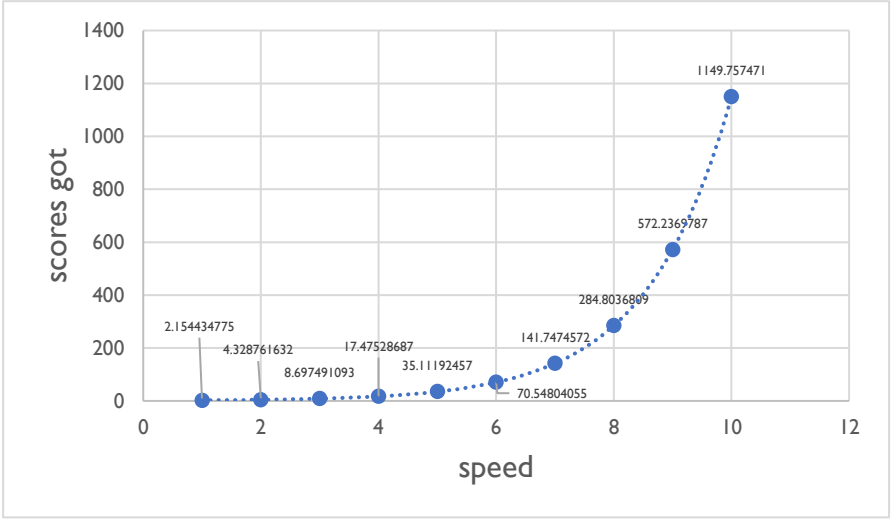
a) Dinosaur is two square for its big body 

b) The King is in a ellipse 

c) And square for other aliens 

```
if (item instanceof Arrow) {
    ((Arrow) item).update(time);
    // When the arrow fired by the player is off the screen, delete the arrow.
    if (item.getY() > this.getHeight() | item.getX() > this.getWidth()) {
        removeItems.add(item);
    }
    for (Item item2 : items) {
        if (item2 instanceof Alien) {
            // When the arrow fired by the player hits alien, delete the arrow, alien will
            // be attacked and play the sound effect, and the player will gain points.
            if (((Alien) item2).getMoveMode() == 4 && ((Math.abs(item2.getX() -
(item.getX() - 50)) < 50
                    & Math.abs(item2.getY() - (item.getY() - 75)) < 75) ||
Math.abs(item2.getX() - (item.getX() + 50)) < 50
                    & Math.abs(item2.getY() - (item.getY() + 75)) < 75)) {
                score += (int) (1.07226722 * Math.exp((((Arrow) item).getSpeed() *
0.06977531))) * window.getDifficulty();
                removeItems.add(item);
                new Music().play("sound/hit.wav", false);
                ((Alien) item2).Attacked(((Arrow) item).getSpeed());
            } else if (item2 instanceof KingAlien) {
                if ((Math.pow((item2.getX()) - item.getX(), 2) - (7396 - 0.6582 *
Math.pow((item2.getY()) - item.getY(), 2)) <= 0)) {
                    removeItems.add(item);
                    new Music().play("sound/hit.wav", false);
                    ((Alien) item2).Attacked(((Arrow) item).getSpeed());
                }
            } else if (Math.abs(item2.getX() - item.getX()) < 25 & Math.abs(item2.getY() -
item.getY()) < 25) {
                score += (int) (1.07226722 * Math.exp((((Arrow) item).getSpeed() *
0.06977531)));
                removeItems.add(item);
                new Music().play("sound/hit.wav", false);
                ((Alien) item2).Attacked(((Arrow) item).getSpeed());
            }
        }
    }
}
```

10. Score is calculated by:
    a) Hurt
    b) Level of difficulty
    c) So the more one-time injuries (arrow speed) and the more difficult the game, the more points you get. In other words, the fewer arrows used to kill an alien, the higher the score
    d)




```
score += (int) (1.07226722 * Math.exp((((Arrow) item).getSpeed() * 0.06977531))) *
window.getDifficulty();
```

11. Ranking

```
try {
    BufferedWriter out;
    out = new BufferedWriter(new OutputStreamWriter(
            new FileOutputStream("log/history.txt", true)));
    String str = (window.male ? window.player1 : window.player2) + "," + panel.time + ","
+ panel.score + "\n";
    out.write(str);
    out.close();
} catch (FileNotFoundException e) {
    e.printStackTrace();
}

InputStream in;
InputStreamReader ir;
BufferedReader br;
//player's name and score would be stored in a text file named "history"
in=new BufferedInputStream(new FileInputStream("log/history.txt"));
ir=new InputStreamReader(in,"utf-8");
br= new BufferedReader(ir);

String line="";
```

```
while((line=br.readLine())!=null){
    playerScore.add(line);
}
br.close();
ir.close();
in.close();
Object[][] rowData = {
        {"Player", "Time", "Score"},
        {0,0,0},
        {0,0,0},
};
//if there is more than two player's information stored in "history",then read the top two
player's information
if(playerScore.size() > 2) {
    rowData[1][0] = playerScore.get(playerScore.size() - 2).split(",")[0];
    rowData[1][1] = playerScore.get(playerScore.size() - 2).split(",")[1];
    rowData[1][2] = playerScore.get(playerScore.size() - 2).split(",")[2];
    rowData[2][0] = playerScore.get(playerScore.size() - 1).split(",")[0];
    rowData[2][1] = playerScore.get(playerScore.size() - 1).split(",")[1];
    rowData[2][2] = playerScore.get(playerScore.size() - 1).split(",")[2];
}
if(playerScore.size() == 1) {
    rowData[1][0] = playerScore.get(playerScore.size() - 1).split(",")[0];
    rowData[1][1] = playerScore.get(playerScore.size() - 1).split(",")[1];
    rowData[1][2] = playerScore.get(playerScore.size() - 1).split(",")[2];
}

table = new JTable(rowData, columnNames);
((DefaultTableCellRenderer) table.getTableHeader().getDefaultRenderer())
        .setHorizontalAlignment(DefaultTableCellRenderer.CENTER);
```

About us

Mingchong Li – 19726006 – Game frame design and planning
Yusen Luo – 19726013 - Window design and construction
Yunjia Tian – 19722062 – Aliens design
Yunxuan Xiao – 19726069 – Player design
Zhizhou Cheng – 19722087 - Structure optimization and integration
Tingen Yu – 19722033 - Game testing and parameter adjustment

spent 18 days to create your joy!