

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

--- ☞ 📖 ☞ ---



**BÁO CÁO ĐỒ ÁN**  
**NGÔN NGỮ LẬP TRÌNH JAVA**  
**LẬP TRÌNH GAME SPACE HERO**

**GVHD: Huỳnh Tuấn Anh**

SVTH: Đặng Phương Tân – 16521071

Đinh Trọng Tín – 16521071

Ứng Vi Vương – 16521468

Ngô Việt Cường - 16520144

**Tp. Hồ Chí Minh, 6/2019**



## This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

## MỤC LỤC

<b>I. GIỚI THIỆU .....</b>	<b>6</b>
<b>II. HIỆN THỰC .....</b>	<b>6</b>
<b>1. Ý tưởng.....</b>	<b>6</b>
<b>2. Tổ chức các đối tượng.....</b>	<b>7</b>
<b>a. Sprite.....</b>	<b>7</b>
<b>b. Hero .....</b>	<b>8</b>
<b>c. Enemy .....</b>	<b>9</b>
<b>d. Bullet.....</b>	<b>11</b>
<b>e. Item .....</b>	<b>11</b>
<b>f. Common .....</b>	<b>12</b>
<b>g. Main.....</b>	<b>12</b>
<b>h. SpaceHero.....</b>	<b>19</b>
<b>3. Cách tính điểm.....</b>	<b>19</b>
<b>III. CÀI ĐẶT, CÁCH CHƠI .....</b>	<b>19</b>
<b>IV. TÀI LIỆU THAM KHẢO .....</b>	<b>21</b>

## LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với sự hỗ trợ, giúp đỡ dù ít hay nhiều của ít hay nhiều người, của sự giúp đỡ trực tiếp hay gián tiếp. Trong suốt thời gian làm làm đồ án lần này, chúng em đã nhận được sự giúp đỡ nhiệt tình của thầy cô và bạn bè.

Nhóm chúng em xin gửi lời cảm ơn chân thành đến thầy Huỳnh Tuấn Anh, giảng viên bộ môn Ngôn ngữ lập trình Java – Trường Đại học Công nghệ thông tin người đã hướng dẫn chúng em trong suốt thời gian qua. Nếu không có sự giúp đỡ tận tình và giảng dạy những kiến thức trên lớp và giải đáp những thắc mắc của chúng em, thì đồ án lần này rất khó thực hiện được.

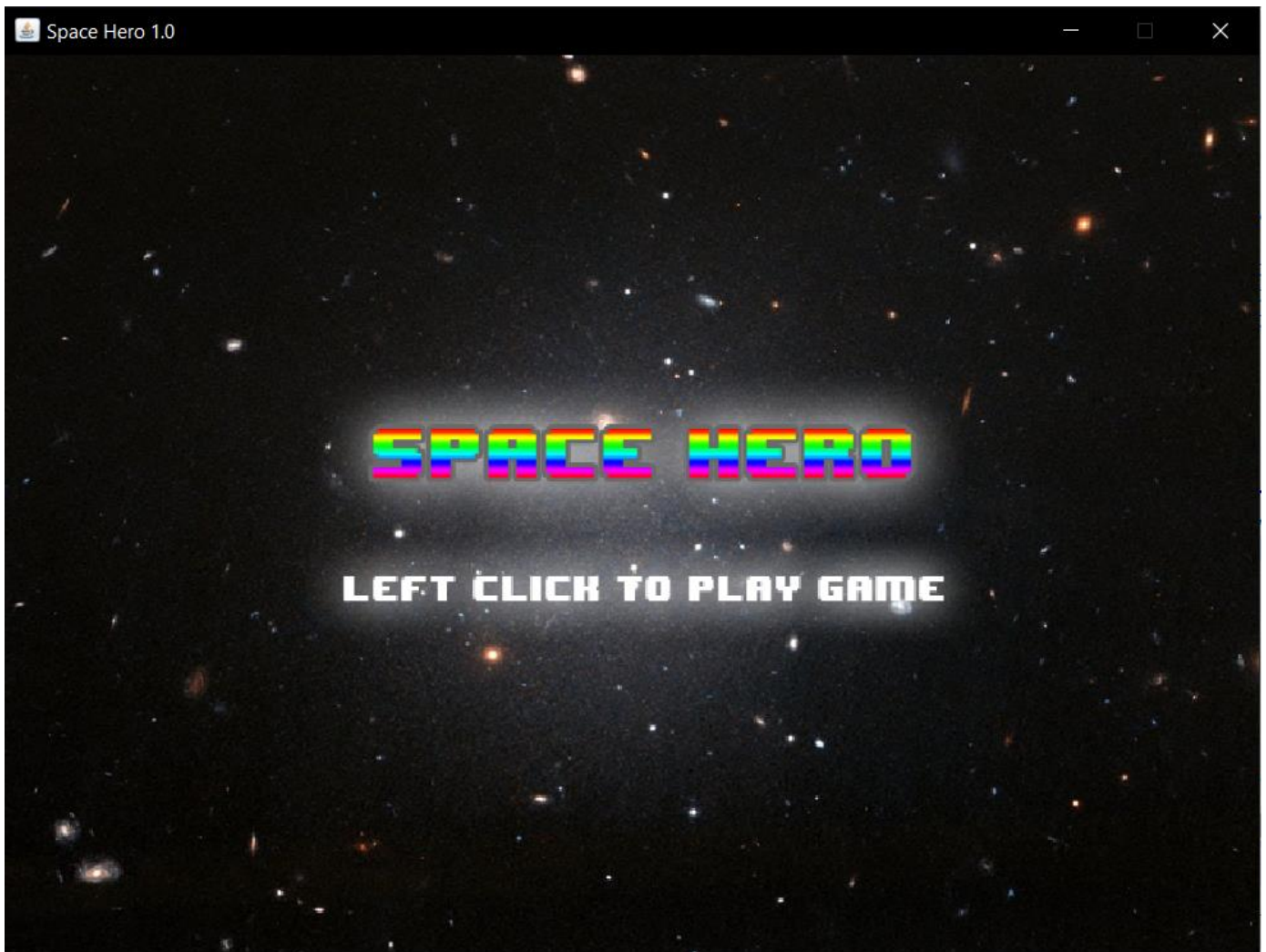
Nhóm chúng em rất mong nhận được những ý kiến đánh giá từ thầy để có những kiến thức bổ ích và hoàn thiện bản thân hơn với môn học, lĩnh vực này. Chúng em xin chân thành cảm ơn thầy.

.....,ngày.....,tháng.....,năm.....

## I. GIỚI THIỆU

**Thể loại game “bắn ruồi”** là một trò chơi arcade rất phổ biến trên các dạng máy thùng cũng như máy chơi game cổ điển. Thể loại này trở nên nổi tiếng và được ưa thích ngay từ khi được phát hành cho đến ngày nay.

Nhằm mục đích tái hiện lại trải nghiệm chơi game kinh điển này và cũng để hoàn thành đồ án, nhóm chúng em đã chọn đề tài lập trình game **Space Hero**.



## II. HIỆN THỰC

### 1. Ý tưởng

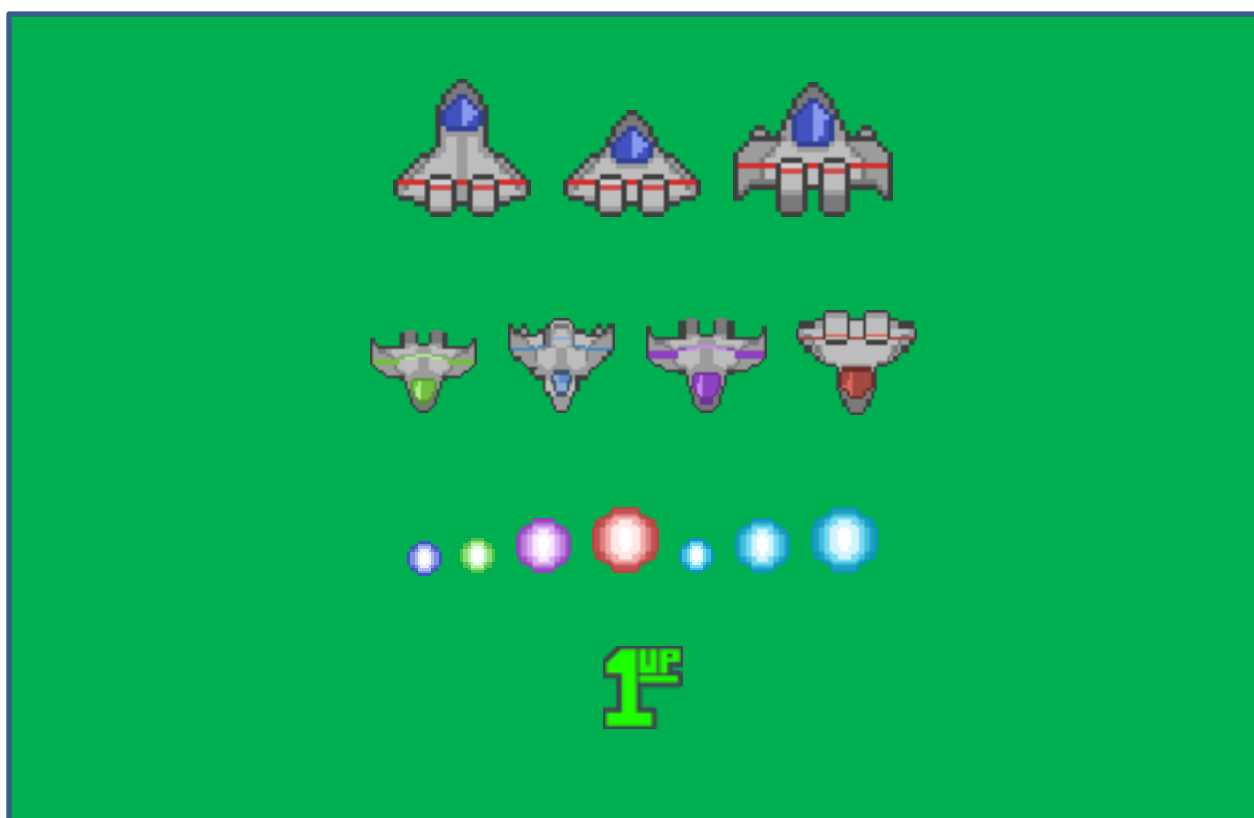
Để thực hiện trước hết cần phải tìm hiểu về các đối tượng trong game. Sau đó là tìm hiểu cách thức vận hành game, tương tác giữa các đối tượng, cách tính điểm,

điều kiện thắng thua, âm thanh... Sâu hơn nữa ta cần tìm hiểu về cách tổ chức các đối tượng chính, các đối tượng hỗ trợ, quản lí sao cho hợp lí mà vẫn hiệu quả về mặt xử lí cũng như lưu trữ. Trong các phần sau em sẽ nêu tư tưởng cũng như cách tổ chức quan trọng chứ không dẫn toàn bộ code do sẽ rất dài dòng và khó hiểu.

## 2. Tổ chức các đối tượng

### a. Sprite

- Đây là lớp cơ bản của hầu hết những gì cần thể hiện cho người xem.
- Tại lớp này chúng ta sẽ xử lí những gì chung nhất của hầu hết các đối tượng để tiết kiệm thời gian và công sức của người lập trình.
- Lớp có chức năng lưu trữ thông tin tọa độ, hình ảnh, kích thước, máu của đối tượng. Về xử lí thì lớp này có chức năng Load hình ảnh, lấy kích thước, trả về tọa độ, cập nhật máu cũng như kiểm tra xem đối tượng đó còn máu hay không.
- Dưới đây là Sprite của các đối tượng:



## b. Hero

- Đây là đối tượng đại diện cho người chơi. Để di chuyển Hero thì người chơi chỉ cần di chuyển con trỏ chuột và nháy chuột trái để bắn.
- Lớp Hero chứa các dữ liệu phục vụ cho việc ở trên:
  - + **dx, dy**: Tọa độ hiện tại của con trỏ chuột.
  - + **level**: Cấp độ của Hero.
  - + **flickering**: Kiểm tra Hero có đang được bắt tử sau khi bị thương không.
  - + **shooting**: Kiểm tra Hero có đang bắn không.
  - + **flickerTime**: Thời gian Hero hết bắt tử.
- Để Hero di chuyển từ vị trí hiện tại tới vị trí con trỏ chuột thì với mỗi khung hình ta cộng tọa độ cho Hero. Nếu Hero tới đủ gần thì gán thẳng tọa độ Hero bằng tọa độ chuột.

```
if ((Math.abs(x-dx) < HERO_SPEED) && ((Math.abs(y-dy) < HERO_SPEED))) {  
    x = dx;  
    y = dy;  
}
```

- Khi Hero nhặt được Item tăng cấp độ thì Hero luôn được lượng máu tối đa, thay đổi ngoài hình cho đến khi đạt cấp 3 thì ngừng thay đổi.

```
if (level < 3) {  
    level++;  
    loadImage("images/hero/hero_level_" + level +  
".png");  
}  
health = HERO_MAX_HEALTH;
```



- Khi Hero bị thương thì chuyển sang trạng thái bất tử trong vài giây và bị trừ máu.

```
flickering = true;  
    flickerTime = Calendar.getInstance();  
    flickerTime.add(Calendar.SECOND, 3);  
    health--;
```

- Khi người chơi di chuyển chuột (sự kiện mouseMoved) thì cập nhật tọa độ con trỏ chuột.

```
public void mouseMoved(MouseEvent e) {  
    dx = e.getX() - width / 2;  
    dy = e.getY() - height / 2;  
}
```

### c. Enemy

- Tổ chức Enemy tương đối đơn giản.
- Để quản lý Enemy ta cần 2 dữ liệu là loại Enemy (có 4 loại là xanh dương, xanh lá, tím và đỏ) và biến cờ kiểm tra Enemy có đang bắn không.
- Mỗi loại Enemy có một lượng máu và loại đạn khác nhau. Xanh dương có máu là 1, xanh lá là 2, tím là 3 và đỏ là 4.

```
switch (type) {  
    case 0:  
  
        loadImage("images/enemies/enemy_blue.png");  
        health = 1;  
        break;  
    case 1:  
  
        loadImage("images/enemies/enemy_green.png");  
        health = 2;  
        break;  
    case 2:  
  
        loadImage("images/enemies/enemy_purple.png");  
        health = 3;  
        break;  
    case 3:  
  
        loadImage("images/enemies/enemy_red.png");  
        health = 4;  
        break;  
}
```

- Việc di chuyển Enemy đơn giản là ta cộng tọa độ theo trạng thái. Có 3 trạng thái di chuyển là qua trái, qua phải và đi xuống.

```

switch (status) {
    case 0:
        x += ENEMY_SPEED_X;
        break;
    case 1:
        x -= ENEMY_SPEED_X;
        break;
    case 2:
        y += ENEMY_SPEED_Y;
        break;
}

```

#### d. Bullet

- Bullet có 2 biến dữ liệu là kiểu đạn (đạn do Hero bắn hay Enemy bắn) và cấp độ của đạn (dùng cho trường hợp Hero bắn).
- Với trường hợp đạn do Hero bắn thì kích thước đạn phụ thuộc vào cấp độ của Hero. Đạn do Enemy bắn thì tùy lại Enemy mà bắn đạn khác nhau, kích thước tăng dần theo loại Enemy.
- Tốc độ bay của đạn phụ thuộc vào cấp độ Hero và không khác biệt giữa các Enemy.

```

if (type == 0) {
    y -= BULLET_SPEED + level - 1;
} else {
    y += BULLET_SPEED;
}

```

#### e. Item

- Lớp Item chỉ đơn giản là được khởi tạo tại vị trí Enemy chết và rơi xuống sau mỗi khung hình.

## **f. Common**

- Đúng hơn thì đây là một Interface có chức năng lưu trữ các hằng số quan trọng giúp cho việc xử lý được đồng nhất, dễ quản lý hơn.
- Interface này gồm các dữ kiện:
  - + **WIDTH, HEIGHT**: Chiều dài và rộng của cửa sổ game.
  - + **ENEMY\_SPEED\_X, ENEMY\_SPEED\_Y**: Tốc độ đi ngang và dọc của Enemy.
  - + **HERO\_SPEED**: Tốc độ của Hero (dù Hero đi theo con trỏ chuột nhưng vẫn cần tốc độ di chuyển).
  - + **ITEM\_SPEED**: Tốc độ rơi của Item.
  - + **BULLET\_SPEED**: Tốc độ bay của đạn.
  - + **DELAY**: Độ delay giữa các khung hình.
  - + **INIT\_HERO\_X, INIT\_HERO\_Y**: Tọa độ đầu tiên của Hero khi mới bắt đầu game.
  - + **HERO\_MAX\_HEALTH**: Máu tối đa của Hero.

## **g. Main**

- Đây là lớp xử lý chính của game bao gồm các dữ liệu sau:
  - + **items**: Danh sách Item.
  - + **enemies**: Danh sách Enemy.
  - + **bullets**: Danh sách Bullet.
  - + **hero**: Biến đại diện Hero.
  - + **timer**: Timer để chạy khung hình.
  - + **inGame**: Kiểm tra game còn đang diễn ra không.
  - + **point**: Tính điểm.
  - + **enemiesStatus**: Quản lý trạng thái di chuyển của Enemy.

- + `stage`: Màn hiện tại.
- + `nextMoveDownTime`: Thời gian di chuyển xuống kế tiếp của Enemy khi có Enemy nằm phía trên màn hình.
- + `changingStage`: Kiểm tra xem có đang chuyển màn không.
- + `changeStageTime`: Thời gian chuyển màn.
- + `backgroundImage`: Ảnh nền game.
- + `titleImage, titleImage2, titleImage3, titleImage4`: Ảnh các tiêu đề của game.
- + `inMenu`: Kiểm tra xem có đang ở Menu game không.
- Khởi đầu của phần xử lý luôn cần khởi tạo dữ liệu, hàm `initMain()` khởi tạo các dữ liệu cơ bản nhất của giao diện game từ menu đến Game Over, bao gồm cả việc load fonts và ẩn con trỏ chuột:

```

private void initMain() {
    inMenu = true;
    setFocusable(true);
    setBackground(Color.BLACK);
    setDoubleBuffered(true);
    timer = new Timer(DELAY, this);
    timer.start();
    backgroundImage = new
ImageIcon("images/backgrounds/background_1.jpg").getImage();

    titleImage = new
ImageIcon("images/backgrounds/title.png").getImage();
    titleImage2 = new
ImageIcon("images/backgrounds/title_2.png").getImage();
    titleImage3 = new
ImageIcon("images/backgrounds/title_3.png").getImage();
    titleImage4 = new
ImageIcon("images/backgrounds/title_4.png").getImage();
    addMouseMotionListener(new MAdapter());
    addMouseListener(new MAdapter());

    // Add new fonts
    try {
        //create the font to use. Specify the
size!
        Font customFont =
Font.createFont(Font.TRUETYPE_FONT, new File("fonts/8-
bit wonder.ttf")).deriveFont(12f);
        GraphicsEnvironment ge =
GraphicsEnvironment.getLocalGraphicsEnvironment();
        //register the font
        ge.registerFont(customFont);
    } catch (IOException e) {
        e.printStackTrace();
    } catch (FontFormatException e) {
        e.printStackTrace();
    }
}

```

```

        // Transparent 16 x 16 pixel cursor image.
        BufferedImage cursorImg = new BufferedImage(16,
        16, BufferedImage.TYPE_INT_ARGB);

        // Create a new blank cursor.
        Cursor blankCursor =
        Toolkit.getDefaultToolkit().createCustomCursor(
            cursorImg, new Point(0, 0), "blank cursor");

        // Set the blank cursor to the JFrame.
        this.setCursor(blankCursor);
    }

```

- Khi khởi động game thì màn hình Menu luôn hiển thị đầu tiên, khi nhận sự kiện click chuột thì chuyển trạng thái sang `inGame` và khởi tạo các dữ liệu cần thiết cho quá trình chơi game:

```

private void initGame() {
    inGame = true;
    point = 0;
    hero = new Hero(INIT_HERO_X, INIT_HERO_Y);
    stage = 0;
    enemies = new ArrayList<Enemy>();
    items = new ArrayList<Item>();
    bullets = new ArrayList<Bullet>();
    setStageChange();
}

```

- Trong quá trình chơi, game sẽ cập nhật liên tục trạng thái của Hero, Enemy, Bullet, Item cũng như sự tương tác và chạm giữa chúng:

```

// Hero vs Enemies
Rectangle heroBound = hero.getBound();

if (!hero.isFlickering()) {
    for (Enemy enemy: enemies) {
        Rectangle enemyBound =
enemy.getBound();
        if (heroBound.intersects(enemyBound)) {
            point += (enemy.getType() + 1) *
10;

            enemy.setHealth(0);
            hero.getHit();
            break;
        }
    }
}

// Hero vs Enemy bullets
if (!hero.isFlickering()) {
    for (Bullet bullet: bullets)
        if (bullet.getType() > 0) {
            Rectangle bulletBound =
bullet.getBound();
            if
(heroBound.intersects(bulletBound)) {
                bullet.setHealth(0);
                hero.getHit();
                break;
            }
        }
}

// Hero vs Items
for (Item item: items) {
    Rectangle itemBound = item.getBound();
    if(heroBound.intersects(itemBound)) {
        point += 10;
        item.setHealth(0);
    }
}

```



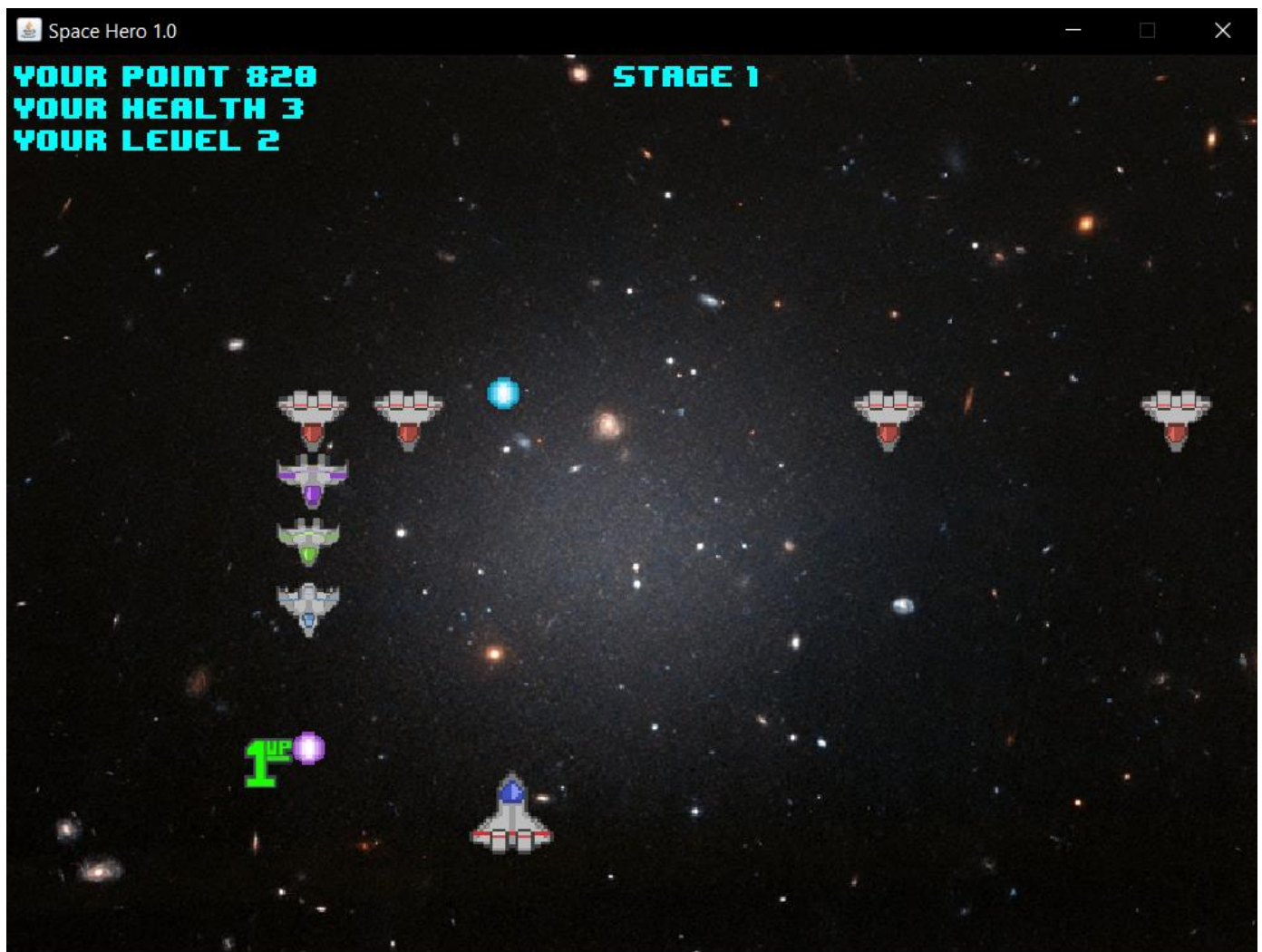
```

        hero.levelUp();
    }
}

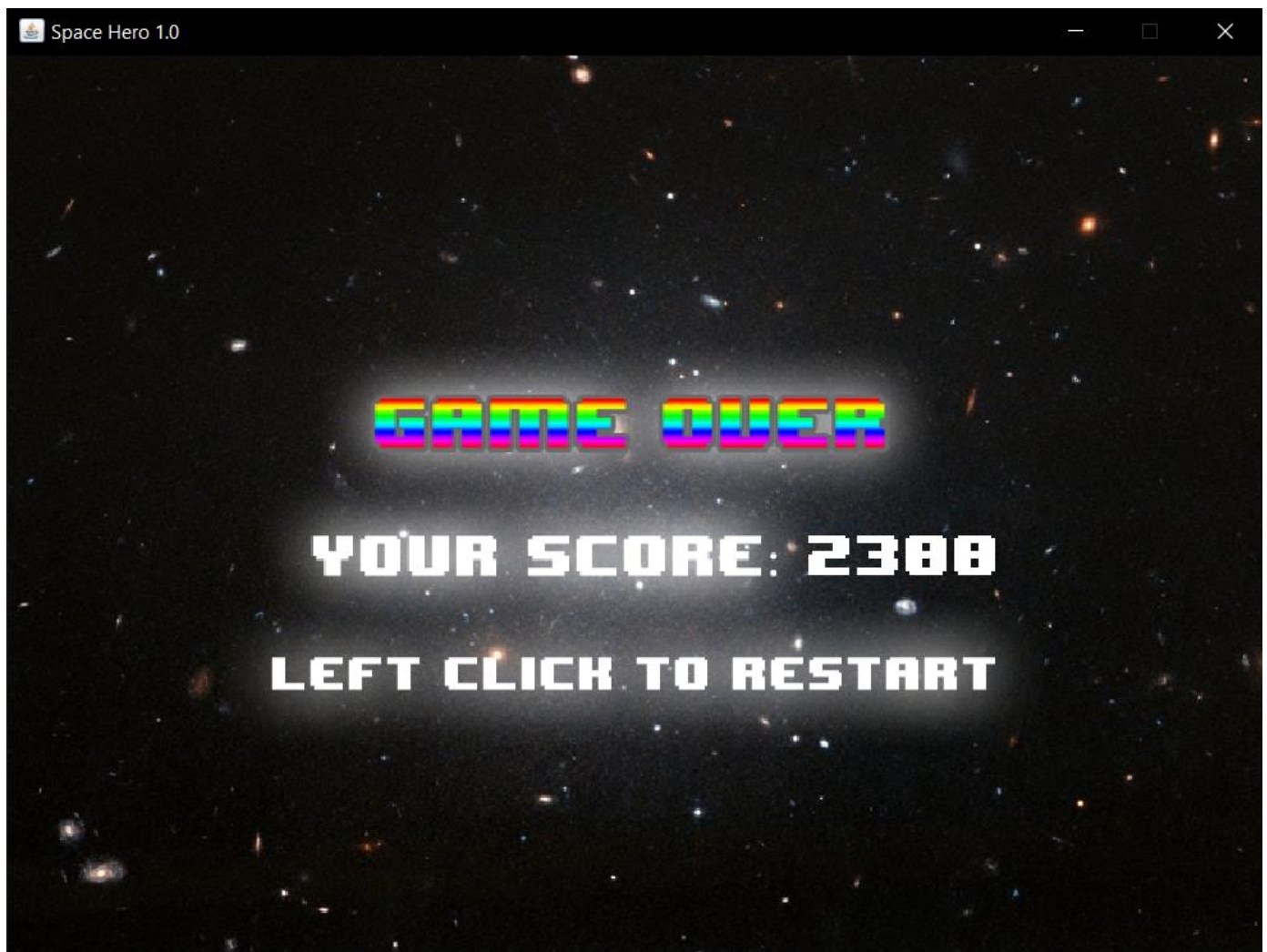
// Hero bullets vs Enemy
for (Bullet bullet: bullets)
    if (bullet.getType() == 0) {
        Rectangle bulletBound =
bullet.getBound();
        for (Enemy enemy: enemies) {
            if (!enemy.isAlive()) continue;
            Rectangle enemyBound =
enemy.getBound();
            if
(bulletBound.intersects(enemyBound)) {
                point +=
Math.min(hero.getLevel(), enemy.health) * 10;
                enemy.setHealth(enemy.health -
bullet.getLevel());
                bullet.setHealth(0);
                break;
            }
        }
    }
}

```

- Khi Hero nháy chuột sẽ bắn ra đạn, các Enemy cũng có xác suất bắn ra đạn ở mỗi khung hình. Khi Enemy chết sẽ có tỉ lệ rơi ra Item tăng level cho Hero...



- Nếu Hero chết thì `inGame` sẽ trở thành **false** và màn hình Game Over sẽ được hiển thị. Người chơi có thể nháy chuột trái để khởi tạo lại lớp này và chơi như game mới.



#### **h. SpaceHero**










- Lớp này có chức năng tạo và chỉnh sửa frame cho game cũng như cung cấp phương thức chạy game.

#### **3. Cách tính điểm**

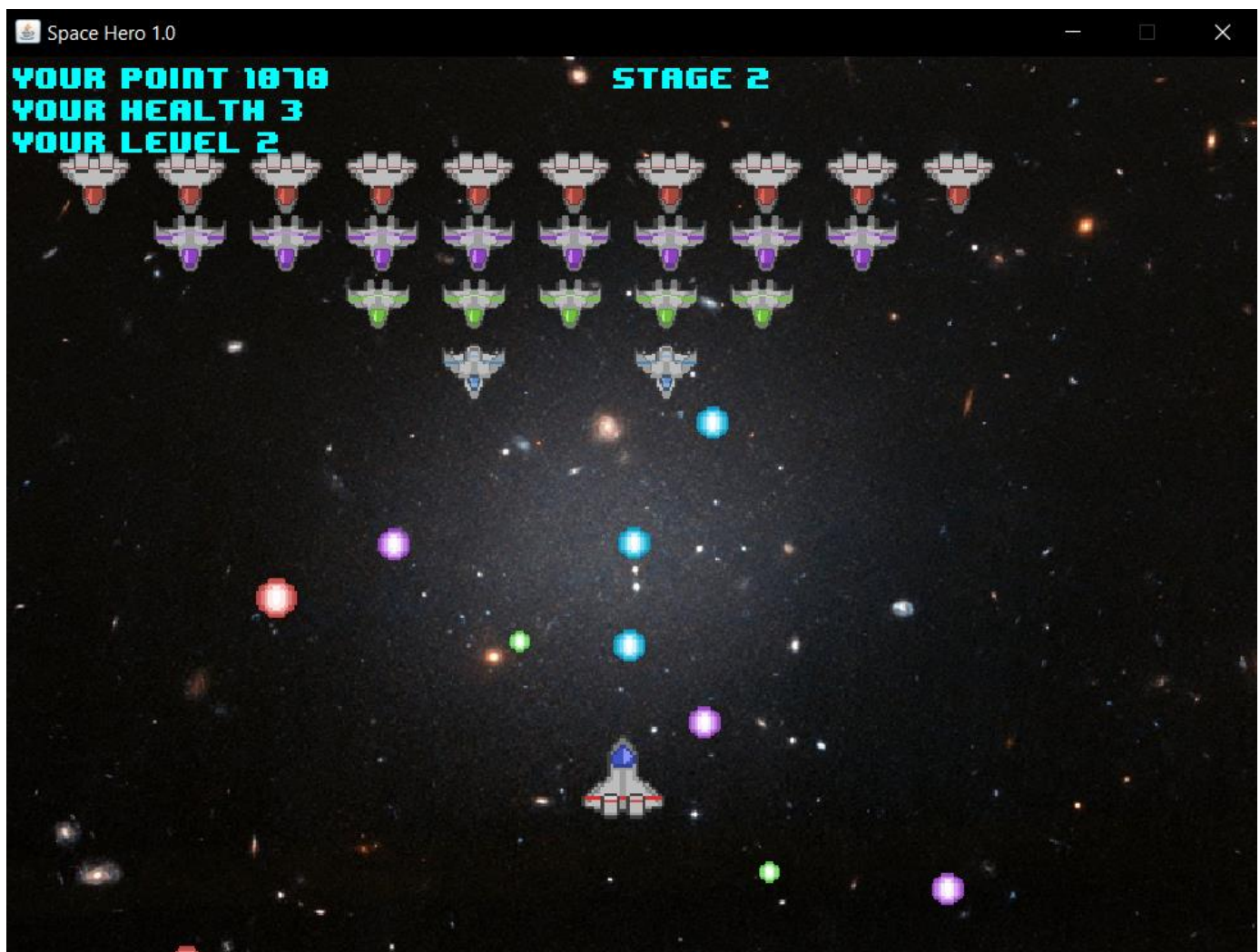
- Với mỗi 1 máu Enemy mất sẽ tăng 10 điểm.
- Với mỗi Item ăn được sẽ được cộng 10 điểm.

### **III. CÀI ĐẶT, CÁCH CHƠI**

- Để chơi game bạn chỉ cần vào file Source của game có tên là “Space Hero” sau đó chọn file “Space Hero.jar” (đòi hỏi phải có Java(TM) Platform SE binary) để chạy được.

	.settings	24/06/2019 12:45 ...	File folder	
	bin	29/06/2019 2:10 A...	File folder	
	fonts	28/06/2019 12:18 ...	File folder	
	images	27/06/2019 1:37 A...	File folder	
	src	25/06/2019 9:52 PM	File folder	
	.classpath	23/06/2019 11:50 ...	CLASSPATH File	1 KB
	.gitignore	23/06/2019 11:50 ...	Text Document	1 KB
	.project	23/06/2019 11:50 ...	PROJECT File	1 KB
	Space Hero.jar	29/06/2019 2:11 A...	Executable Jar File	15 KB

- Sau khi chạy file “Space Hero.jar” mà không gặp bất cứ lỗi nào thì bạn đã có thể tận hưởng trò chơi!



#### **IV. TÀI LIỆU THAM KHẢO**

- [1] Slide tài liệu Ngôn ngữ lập trình Java – Đại học Công nghệ thông tin.
- [2] <https://stackoverflow.com/>
- [3] <https://www.wikihow.com/>

- Hết -