

# C# 프로젝트

## Project Galag

선동운

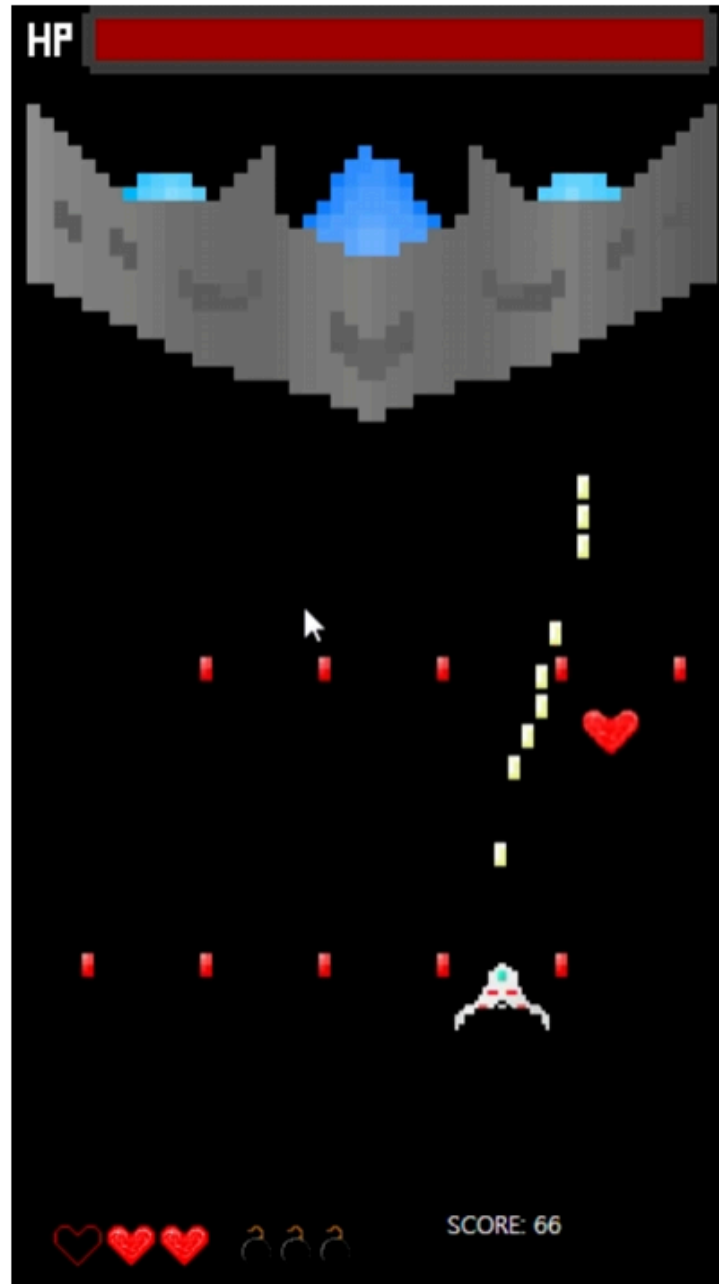
01

---

게임 소개

# 게임 소개

---



제목 : Galag

장르 : 슈팅 게임

주요 라이브러리 : Windows forms

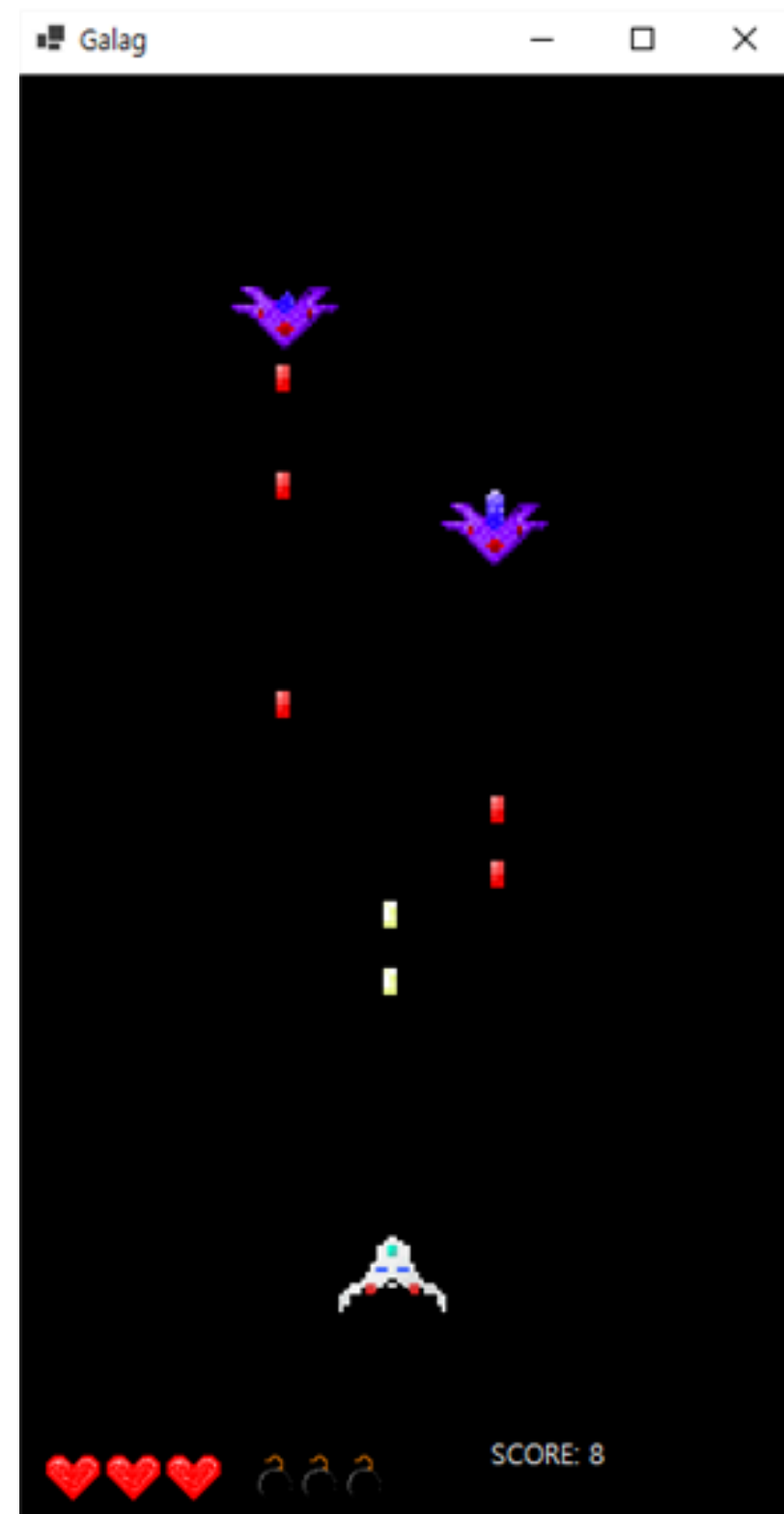
언어 : C#

개발 인원 : 1명

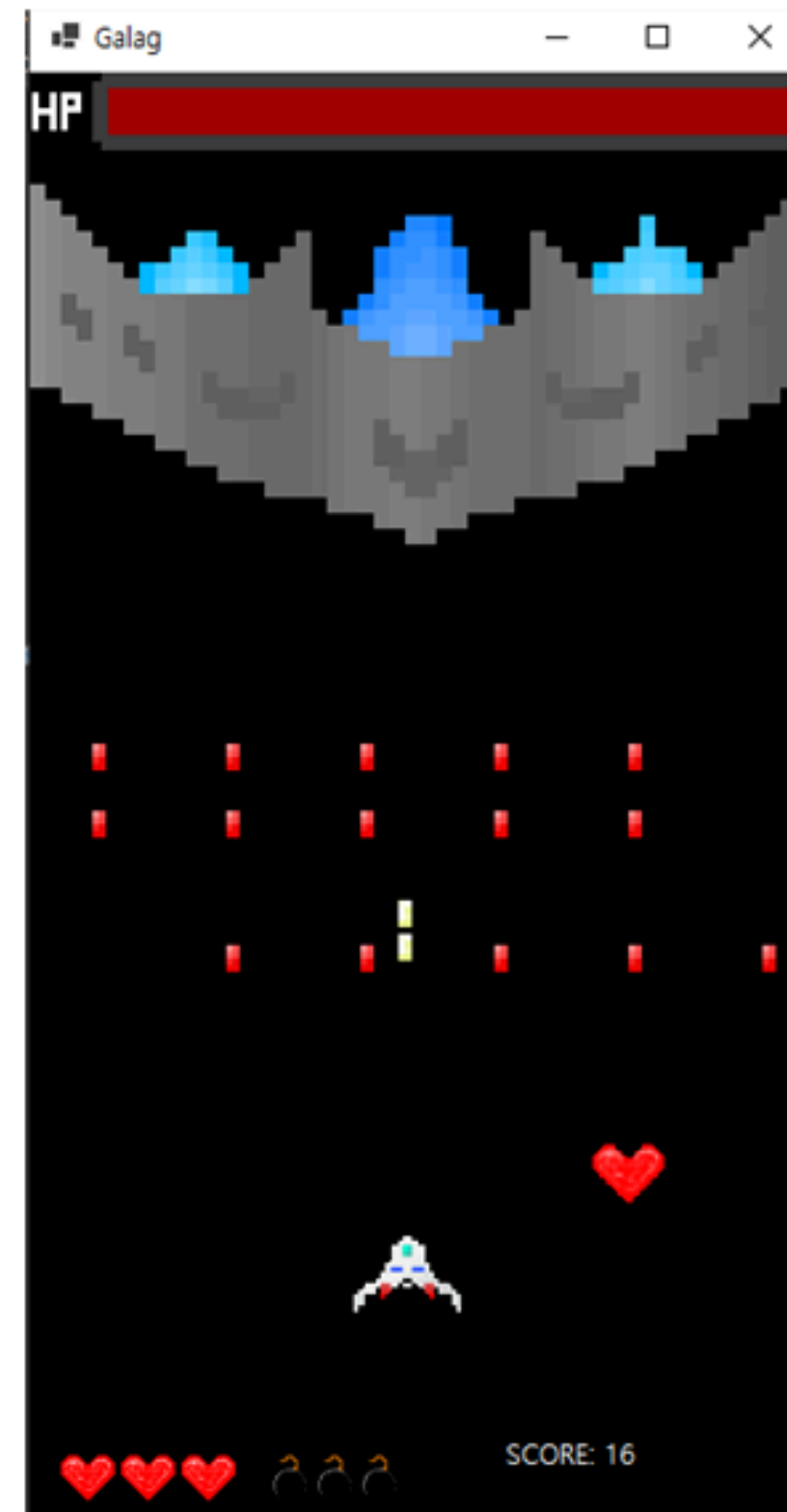
시연 영상 : [https://youtu.be/8r\\_5vPcWxvl?si=cAmTrVmBokNc4txC](https://youtu.be/8r_5vPcWxvl?si=cAmTrVmBokNc4txC)

GitHub : [https://github.com/Seon-dongun/Project\\_Galag](https://github.com/Seon-dongun/Project_Galag)

# 기본 화면 구성



[보스 등장 전 1페이지]



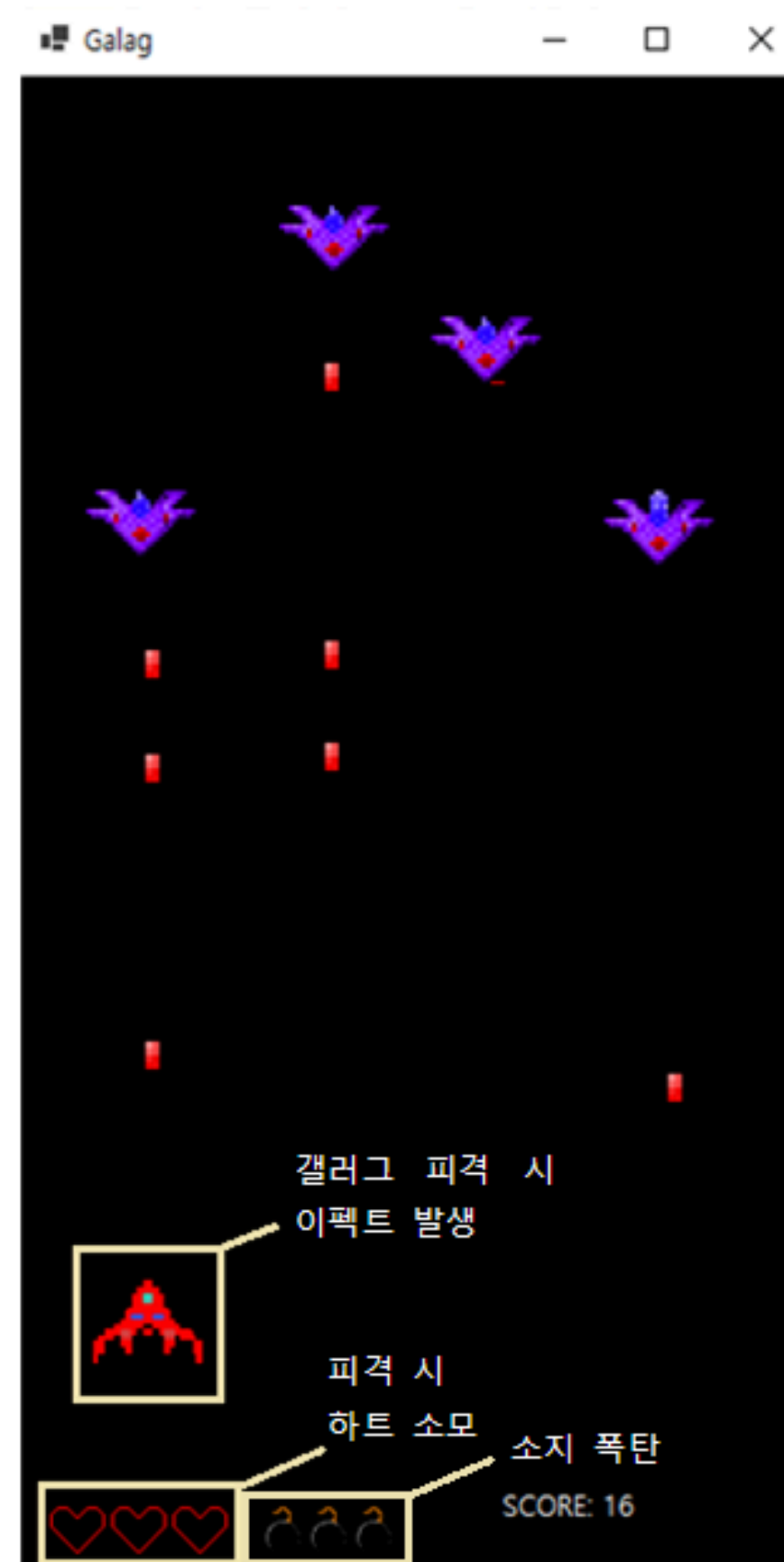
[보스 등장 후 2페이지]

02



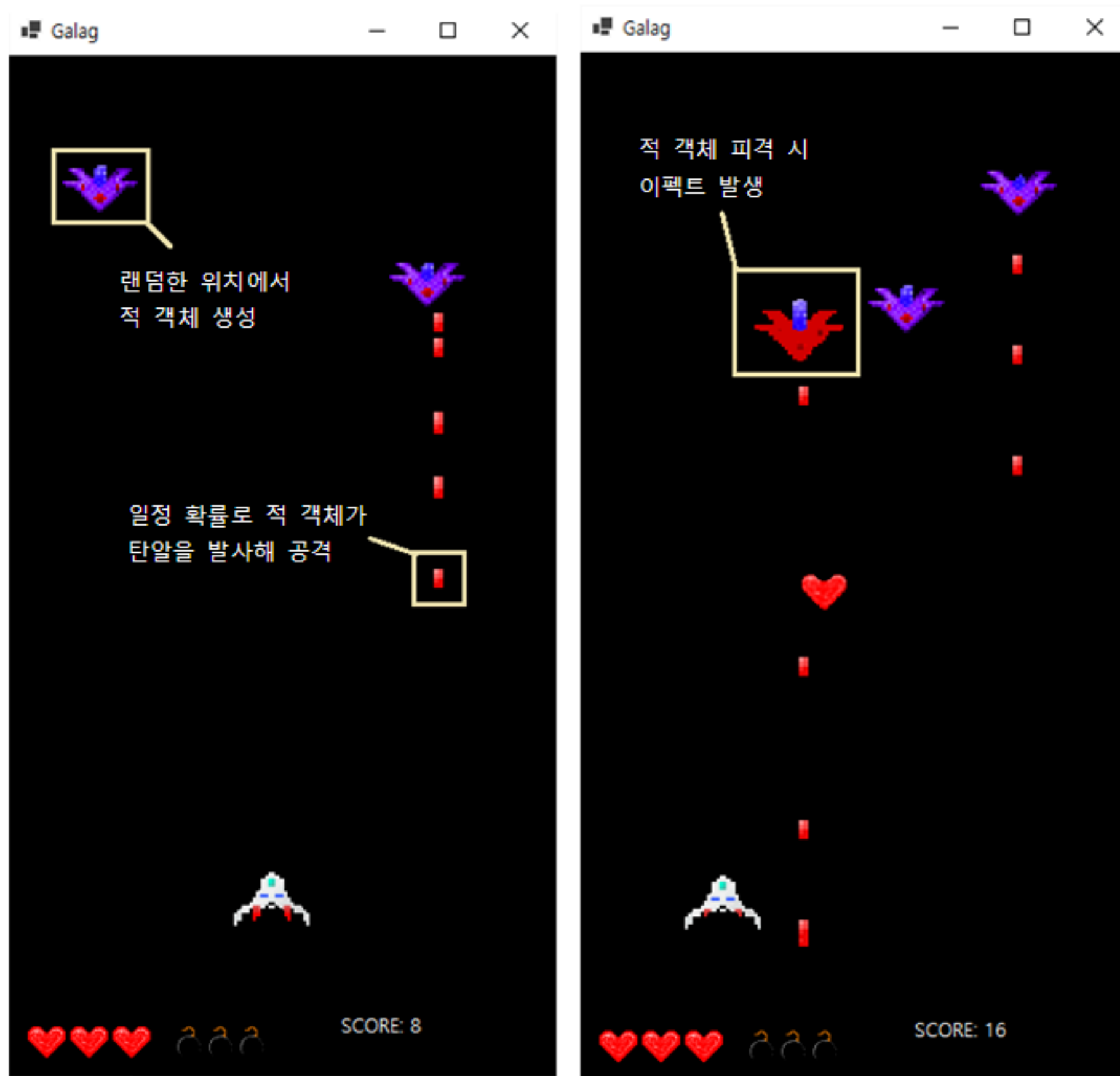
기능 구현

# 갤러그 동작 방식



- 방향키로 이동, Space 바 로 탄알 발사
- F키로 폭탄 사용 가능
- 피격 시 하트 1개 소모

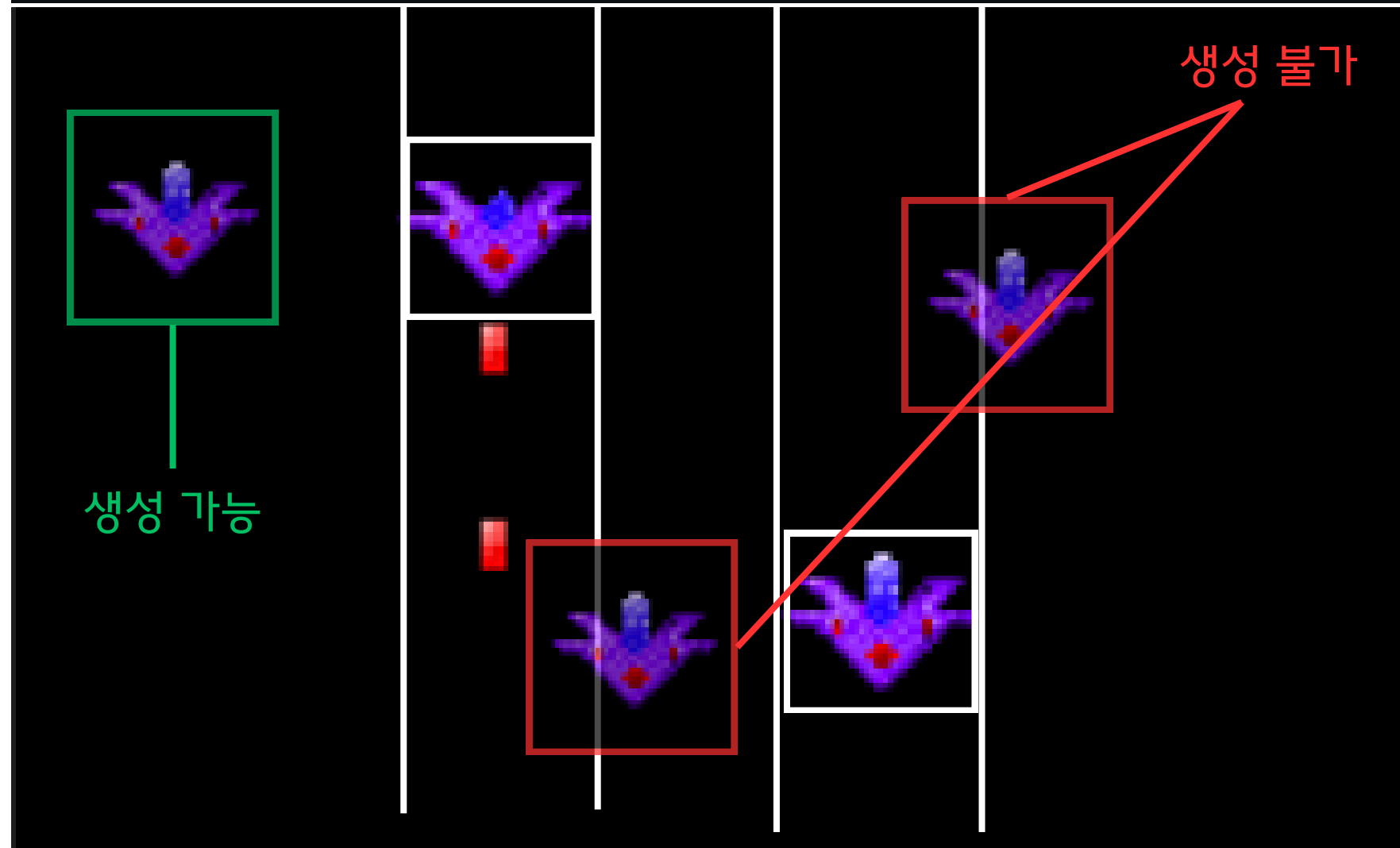
# 적 동작 방식



- 적은 일정한 공간 내에서 랜덤한 위치에서 생성
- 생성된 위치에 고정되어 있고 일정 확률로 탄알 발사
- 피격 시 1의 데미지를 받고, 총 3의 데미지를 받은 적은 제거

# 적 생성

```
// 이미 생성되어 있는 적 객체들을 확인하여 새로 생성하는 적 객체가 기존 적 객체들과 x축으로 겹치지 않는 경우만 새로운 적 객체를 생성하도록 한다
foreach (Enemy tmp in enemyArray)
{
    int x1 = tmp.enemyX - tmp.getWidth() / 2;
    int x2 = tmp.enemyX + tmp.getWidth() / 2;
    if (((x - tmp.getWidth() / 2) >= x1 && (x - tmp.getWidth() / 2) <= x2) || ((x + tmp.getWidth() / 2) >= x1 && (x + tmp.getWidth() / 2) <= x2))
        return; // 기존 적 객체의 x좌표와 새로운 적 객체의 x좌표가 겹치는 경우 새로운 적 객체를 생성하지 않고 return
}
enemyArray.Add(new Enemy(x, y, this)); // 기존 적 객체의 x좌표와 새로운 적 객체의 x좌표가 겹치지 않는 경우 새로운 적 객체를 생성
```



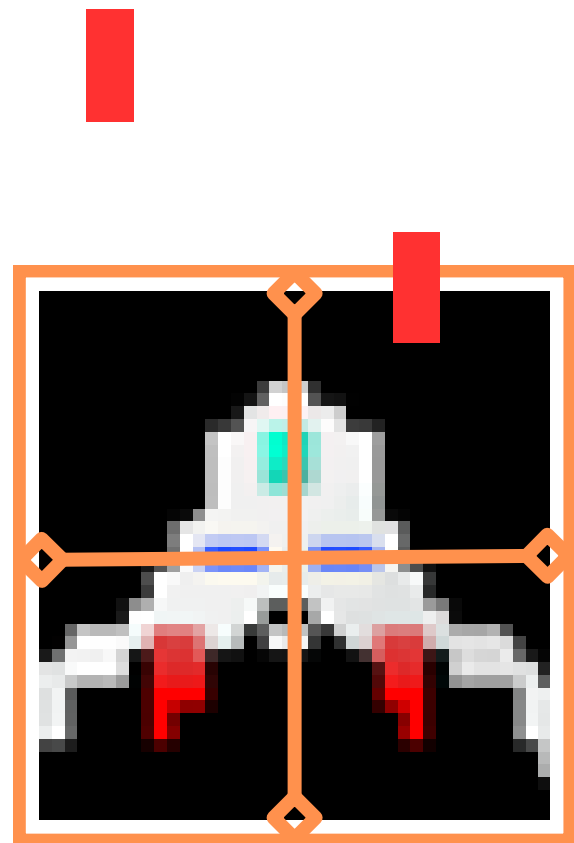
랜덤한 위치에 적 스폰 시, 기존 객체의 x축 범위와 겹치는지 검사해 객체 간 겹침 없이 스폰될 수 있도록 처리



# 피격 구현

```
// 갤러그의 히트박스에 적 객체의 탄알이 만나면 갤러그가 피격된 것으로 판단  
if ((tmp.enemy_armoX >= (galagX - galag.Width / 2) + 30 && tmp.enemy_armoX <= (galagX + galag.Width / 2) + 22) && (tmp.enemy_armoY > galagY - galag.Height / 2) && (tmp.enemy_armoY < galagY + galag.Height / 2))
```

```
// 적 객체의 히트박스에 갤러그 탄알이 만나면 적 객체를 피격된 것으로 판단  
if (tmp.armoY < enemy.enemyY && (enemy.enemyX - (enemy.getWidth() / 2) + 30 <= tmp.armoX && tmp.armoX <= (enemy.enemyX + enemy.getWidth() / 2) + 22))
```



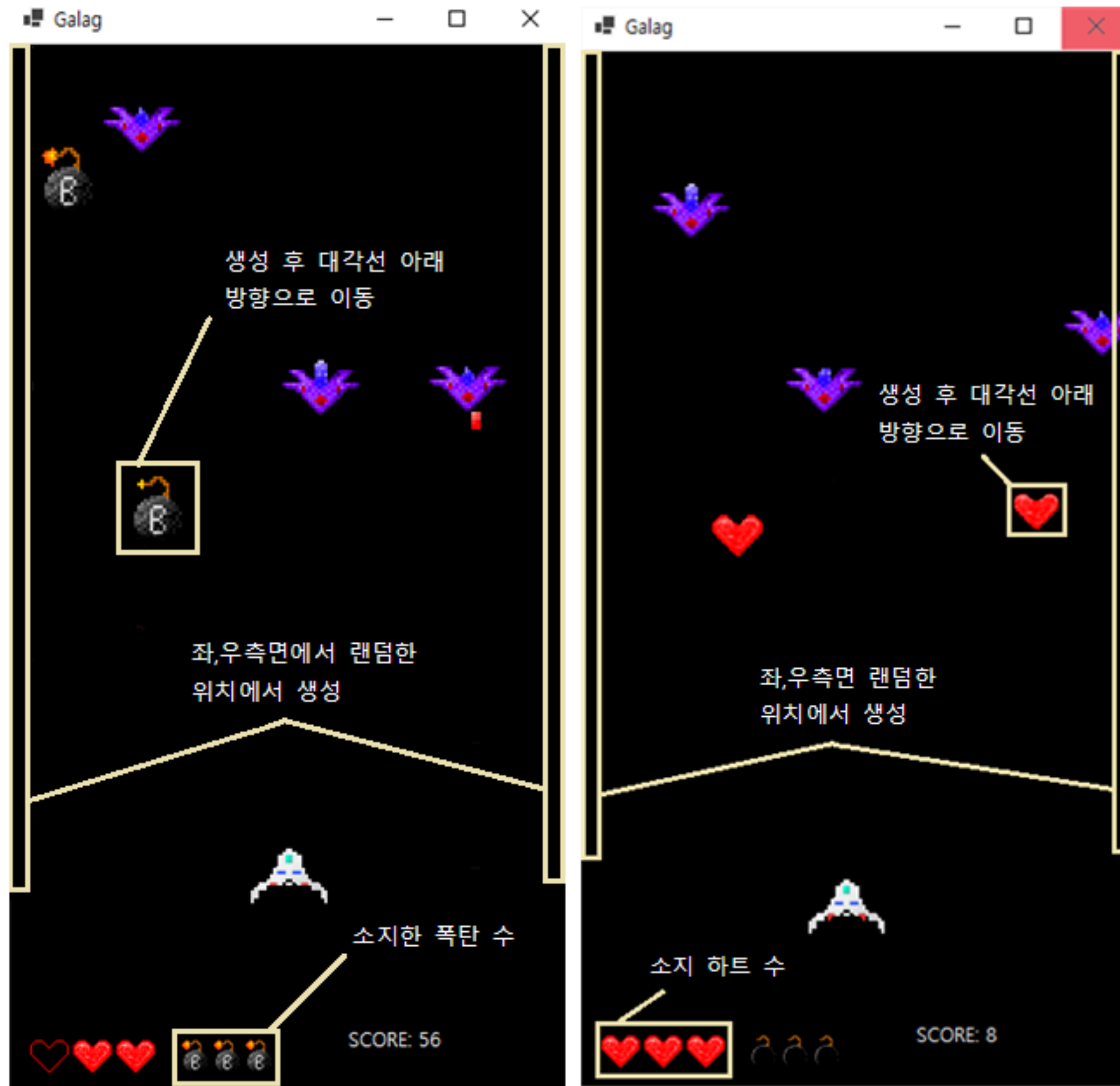
[갤러그 피격 범위]



[적 피격 범위]

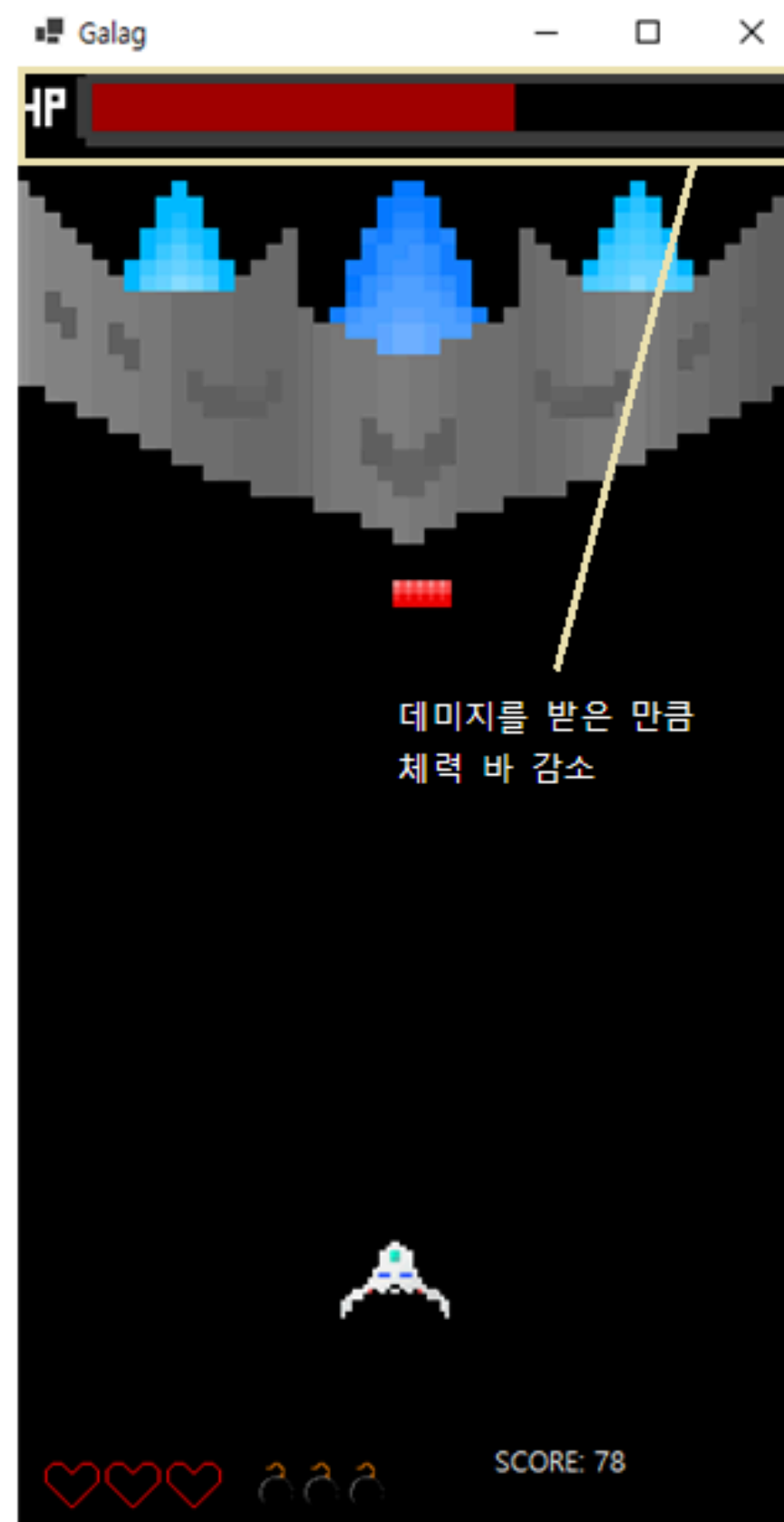
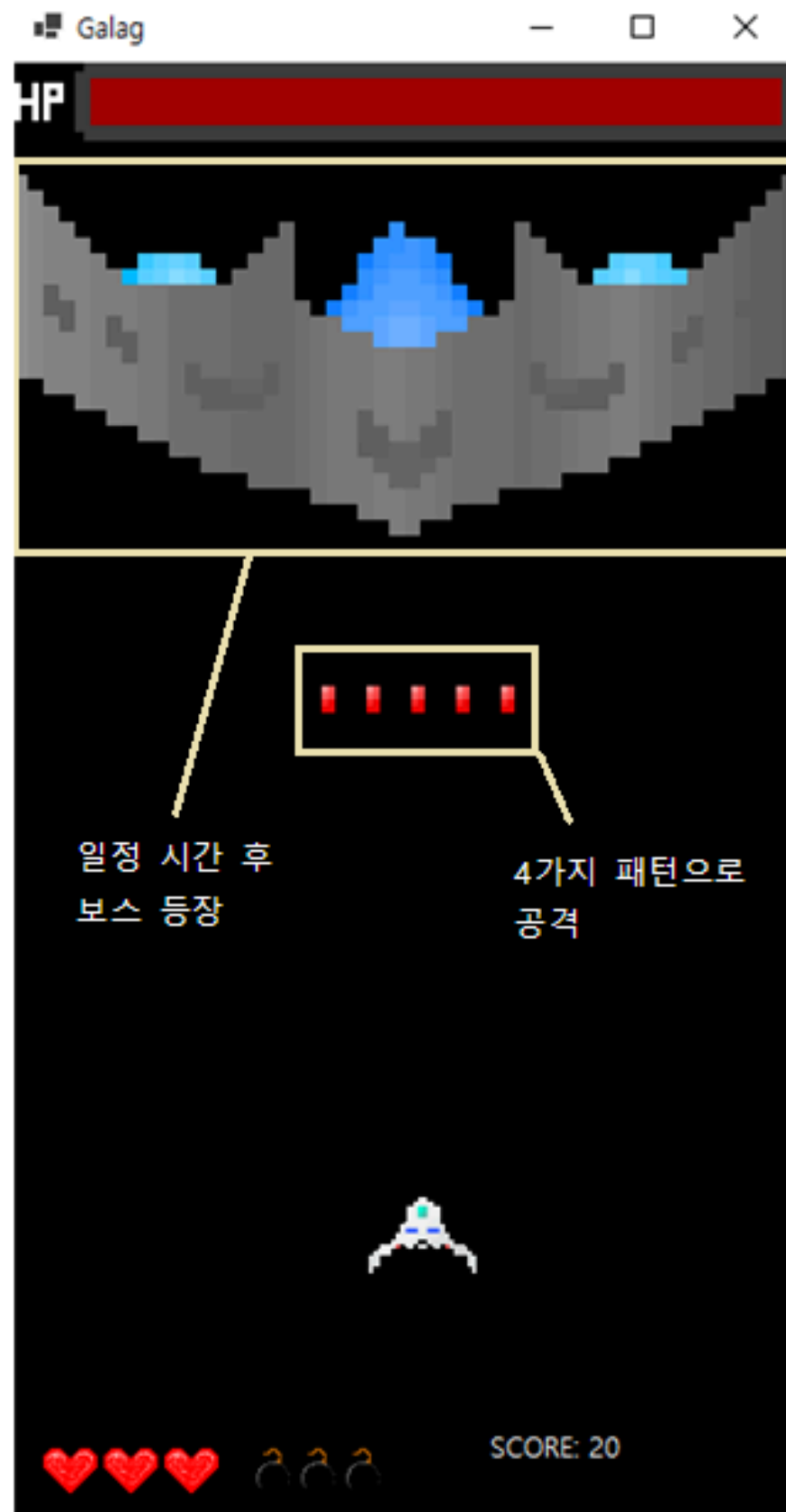
객체와 탄알 간의 오버랩을 판단해 피격 구현

# 아이템 구현



- 일정 간격으로 화면 양측면 중 한 곳에서 랜덤 생성
- 탄알 피격과 동일한 원리로, 갤러그와 아이템 충돌 시 아이템 획득 가능

# 보스 동작 방식

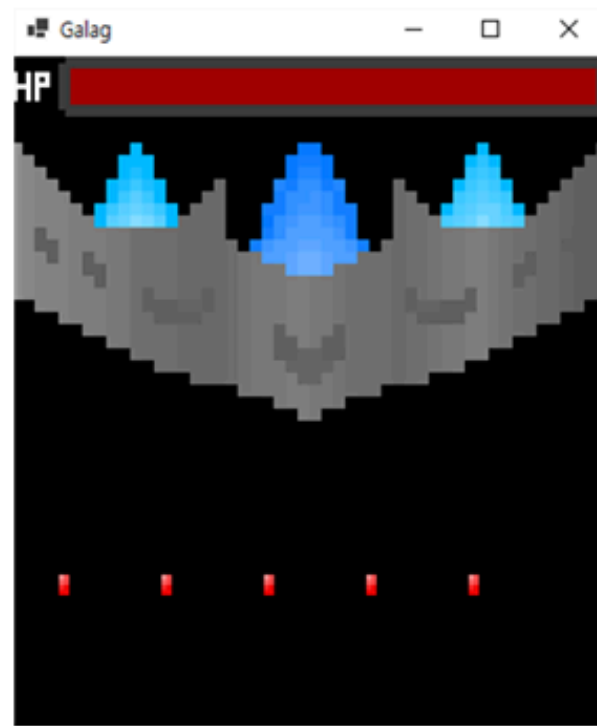


- 일정 시간 동안 일반 적의 공격을 버티면 보스 생성
- 피격된 데미지에 따라 UI 이미지를 교체해 체력 바 감소 구현
- 총 50 데미지를 받게 되면 보스 사망 및 게임 클리어

# 보스 동작 방식



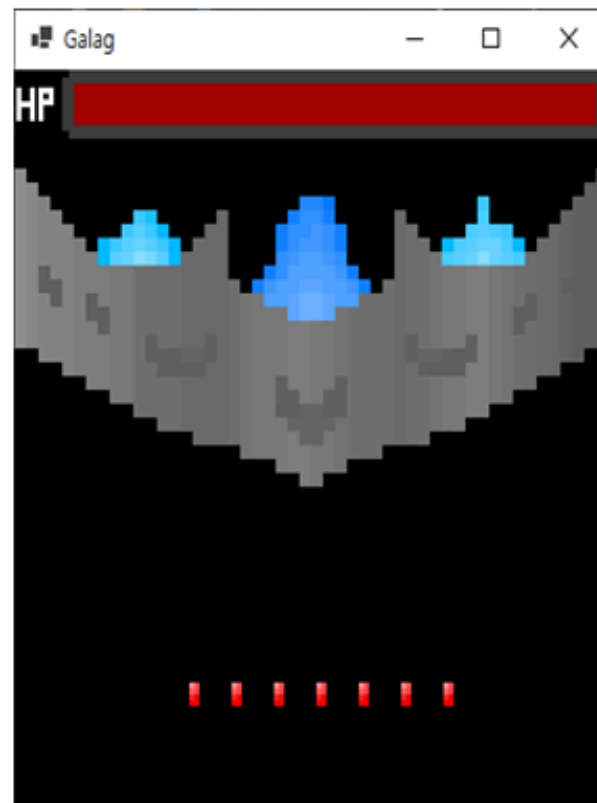
[우측 공격 패턴]



[좌측 공격 패턴]



[방사형 공격 패턴]



[더 빠른 방사형 공격 패턴]

```
Random rand = new Random();
int bossArmoAppear= rand.Next(0, 1000); // 랜덤변수를 통해 보스 객체의 공격 확률을 설정
if (bossArmoAppear < 120) // 보스 객체가 공격을 수행하는 경우
{
    Random pattern = new Random();
    int bossPattern = rand.Next(0, 100); // 랜덤변수를 통해 보스 객체의 공격 패턴을 선택하여 공격을 수행한다.
    if (bossPattern < 25) // 25% 확률로는 방사형으로 움직이는 탄알을 발사하는 패턴
    {
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, -10, 17));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, -5, 17));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, 0, 17));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, 5, 17));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, 10, 17));
    }
    else if (bossPattern < 60) // 35% 확률로는 우측에서 생성되고 직진하며 움직이는 탄알을 발사하는 패턴
    {
        enemyArmoArray.Add(new EnemyArmo(240, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(180, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(120, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(60, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(0, 200, this, 0, 15));
    }
    else if (bossPattern < 85) // 35% 확률로는 좌측에서 생성되고 직진하며 움직이는 탄알을 발사하는 패턴
    {
        enemyArmoArray.Add(new EnemyArmo(60, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(120, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(180, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(240, 200, this, 0, 15));
        enemyArmoArray.Add(new EnemyArmo(300, 200, this, 0, 15));
    }
    else // 15% 확률로는 더 빠르게 방사형으로 움직이는 탄알을 발사하는 패턴
    {
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, -15, 20));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, -10, 20));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, -5, 20));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, 0, 20));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, 5, 20));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, 10, 20));
        enemyArmoArray.Add(new EnemyArmo(150, 200, this, 15, 20));
    }
}
```

일정 확률에 따라 4가지의 공격 패턴 중 선택해 공격

# 게임 클리어 / 게임 오버



- 클리어 여부에 따라 게임 클리어 / 게임 오버 표시
- Replay 버튼으로 게임 재시작
- Exit 버튼으로 게임 종료

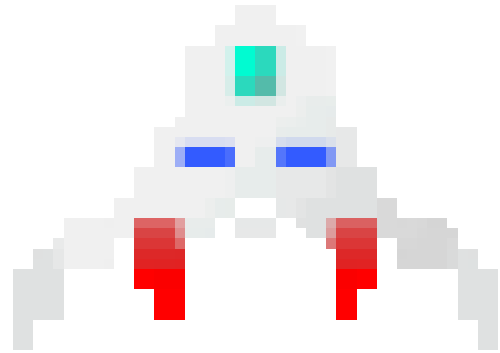
03



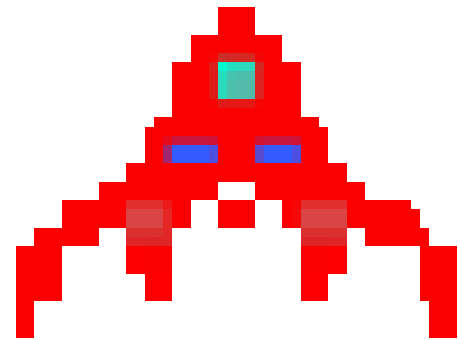
리소스 파일  
제작

# 리소스 파일 제작

- Pikel-0.14.0 프로그램으로 제작



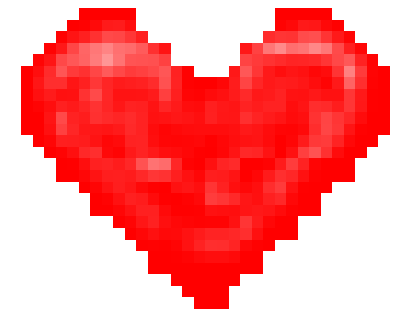
[galag]



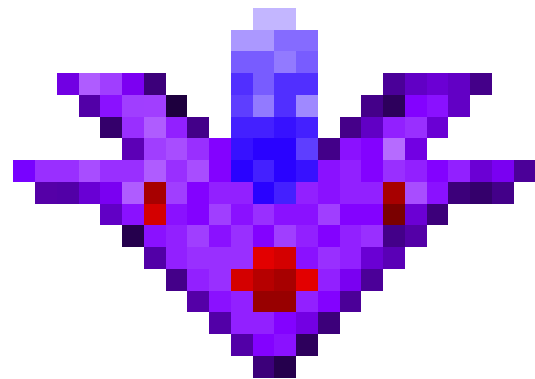
[galagHit]



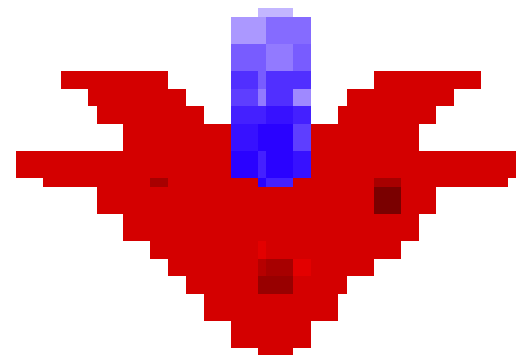
[boom]



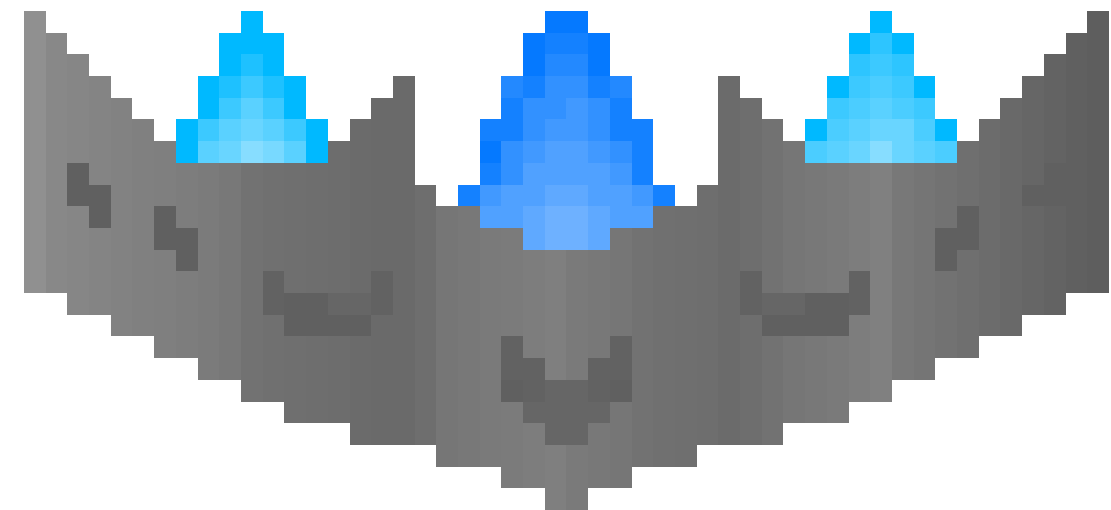
[heart]



[enemy]



[enemyHit]



[boss]

[이미지 이름 클릭 시 GIF 파일 확인 가능]

# 리소스 파일 제작



[boombar0~3]



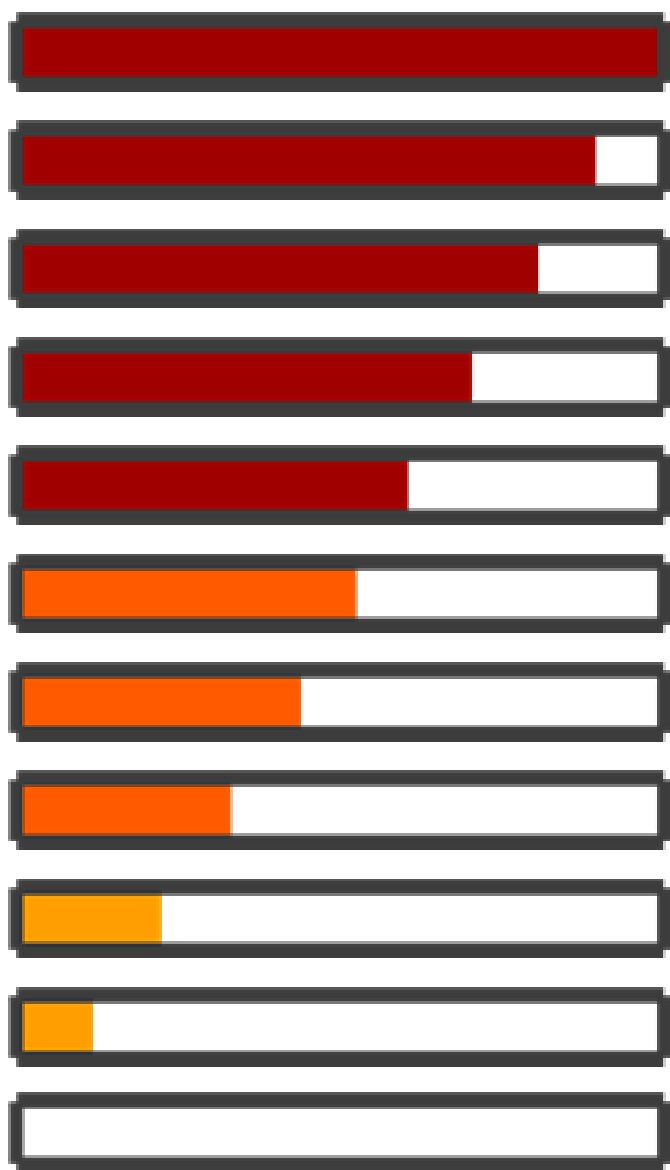
[HeartBar0~3]



[galagArmo]



[enemyArmo]



[bossHP0~50]