Involved classes:

* Projectile: Mainly for utilising the piercing projectile's functions
* CartesianAndPolar: Converts between cartesian and polar coordinates to reorient after a knockback
* HitSystem: Increments the number of hits after enemy is damaged

**Enemy class**

create float baseSpeed and set it to 2  
create float typeSpeedReduction and set it to 0.2

create float piercingCooldown and set it to 0.7  
create float kbDuration and set it to 0.5  
create float splitArcHalf and set it to 30

create public gameObject enemyTriangle  
create public gameObject enemySquare

create int enemyID and set it to 1  
create int health and set it to 1  
create float finalSpeed and set it to 2

create float currPiercingCooldown and set it to 0

UpdateSkin function:

Look at cases for health:  
case 0:  
 change this sprite's colour to Clear  
case 1:  
 change this sprite's colour to Red  
case 2:  
 change this sprite's colour to Green  
case 3:  
 change this sprite's colour to Blue  
case 4:  
 change this sprite's colour to Black

CreateEnemySettings public function:  
parameters: int newID, int newHealth

set enemyID to newID  
set finalSpeed to baseSpeed - typeSpeedReduction \* (newID - 1)  
set health to newHealth  
execute UpdateSkin() function

Update function:

translate this gameObject by finalSpeed \* Time.deltaTime in the direction of the local y axis  
if (currPiercingCooldown < piercingCooldown):  
 add Time.deltaTime to currPiercingCooldown

OnCollisionEnter function:  
parameter: Collision other

if (other's tag == "Projectile):  
 execute TakeDamage function  
else if (other's tag == "Piercing Proj" AND currPiercingCooldown >= piercingCooldown):  
 execute TakeDamage function  
 set currPiercingCooldown to 0

TakeDamage function:

In Hit Display's HitSystem component, execute IncrementHit()  
if (enemyID > 1 && health > 1):  
 execute SplitEnemy()  
else if (enemyID = 1 && health > 1):  
 subtract health by 1  
 execute UpdateSkin()  
 start Coroutine Knockback  
else if (health <= 1):  
 start Coroutine KillEnemy

SplitEnemy function:

Repeat twice:  
 Look at cases for enemyID - 1:  
 case 1:  
 create GameObject newEnemy and Instantiate it as enemyTriangle at this gameObject's position  
 case 2:  
 create GameObject newEnemy and Instantiate it as enemySquare at this gameObject's position  
 In newEnemy's Enemy component, execute CreateEnemySettings(enemyID -1, health-1)  
 Rotate newEnemy by a random range between -splitArcHalf and +splitArcHalf  
 In newEnemy's Enemy component, start the Coroutine Knockback  
Start Coroutine KillEnemy

Knockback public coroutine:  
create float kbTime and set it to 0  
create float array newPolarCoOrds

set finalSpeed to 0  
Rotate enemy 180 degrees  
while (kbTime <= kbDuration):  
 translate this gameObject by finalSpeed \* Time.deltaTime \* (1/kbDuration) in the direction of the local y axis  
 Add Time.deltaTime to kbTime  
 wait Time.deltaTime seconds  
set finalSpeed to baseSpeed - typeSpeedReduction \* (newID - 1)  
set newPolarCoOrds to the result of CartesianAndPolar.ConvertToPolar(this objecty's x position, this object's y position)  
set this object's rotation to newPolarCoOrds + 90 degrees

KillEnemy coroutine:  
 set health to 0  
 execute UpdateSkin()  
 Play child particle system  
 Wait 0.9 seconds  
 Destroy this object