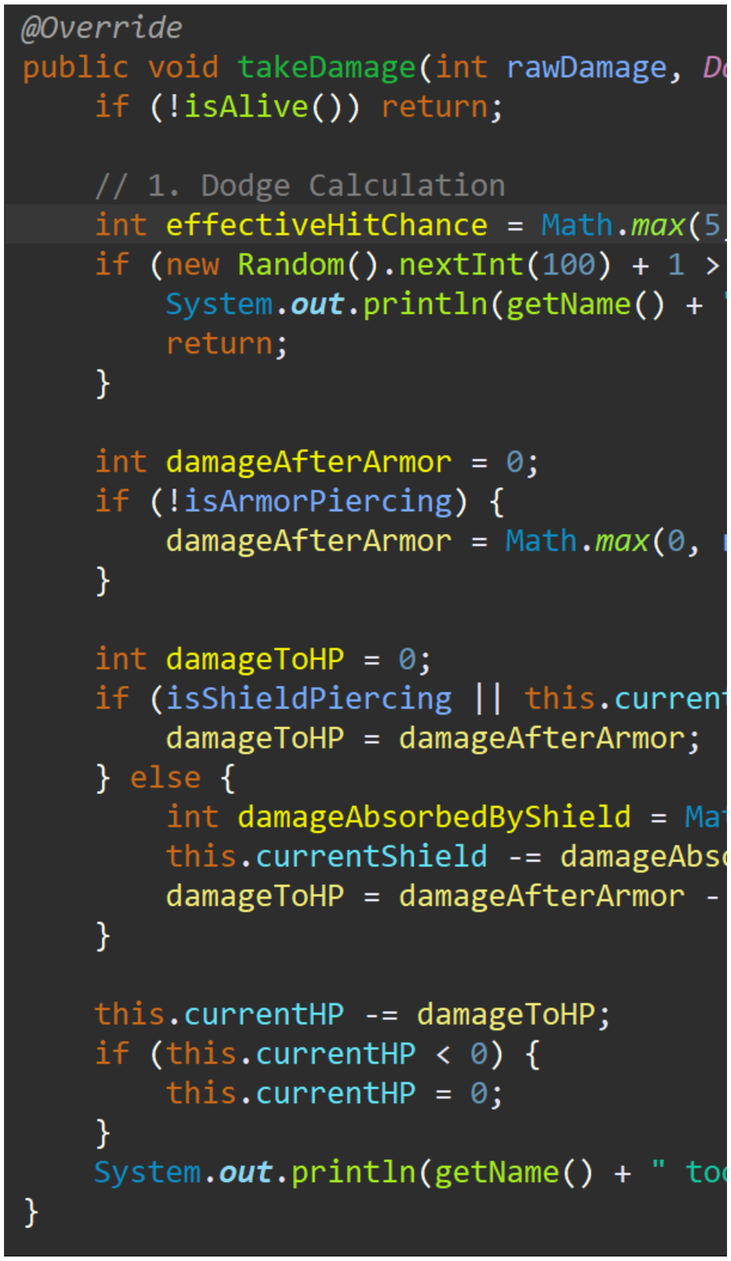
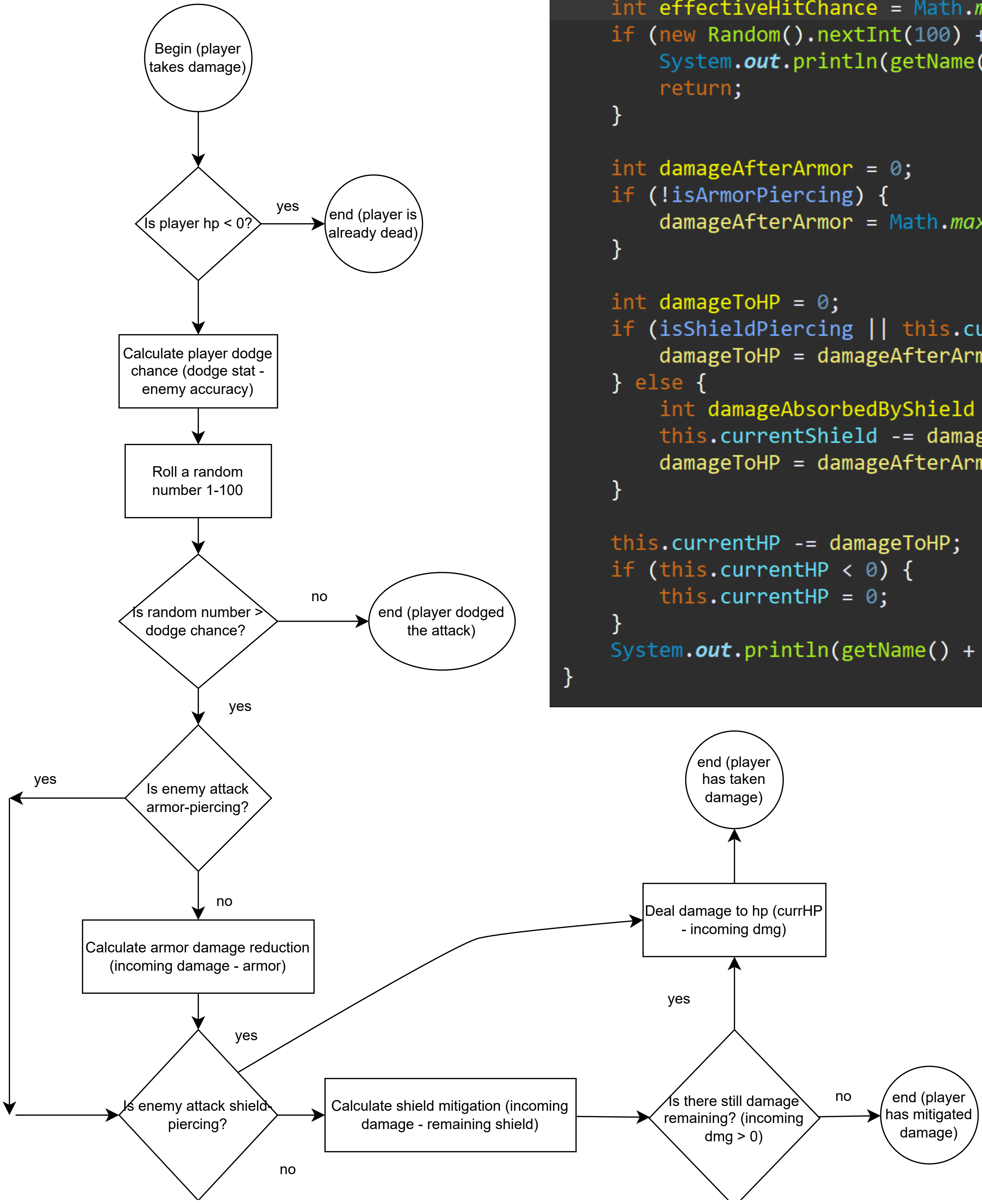
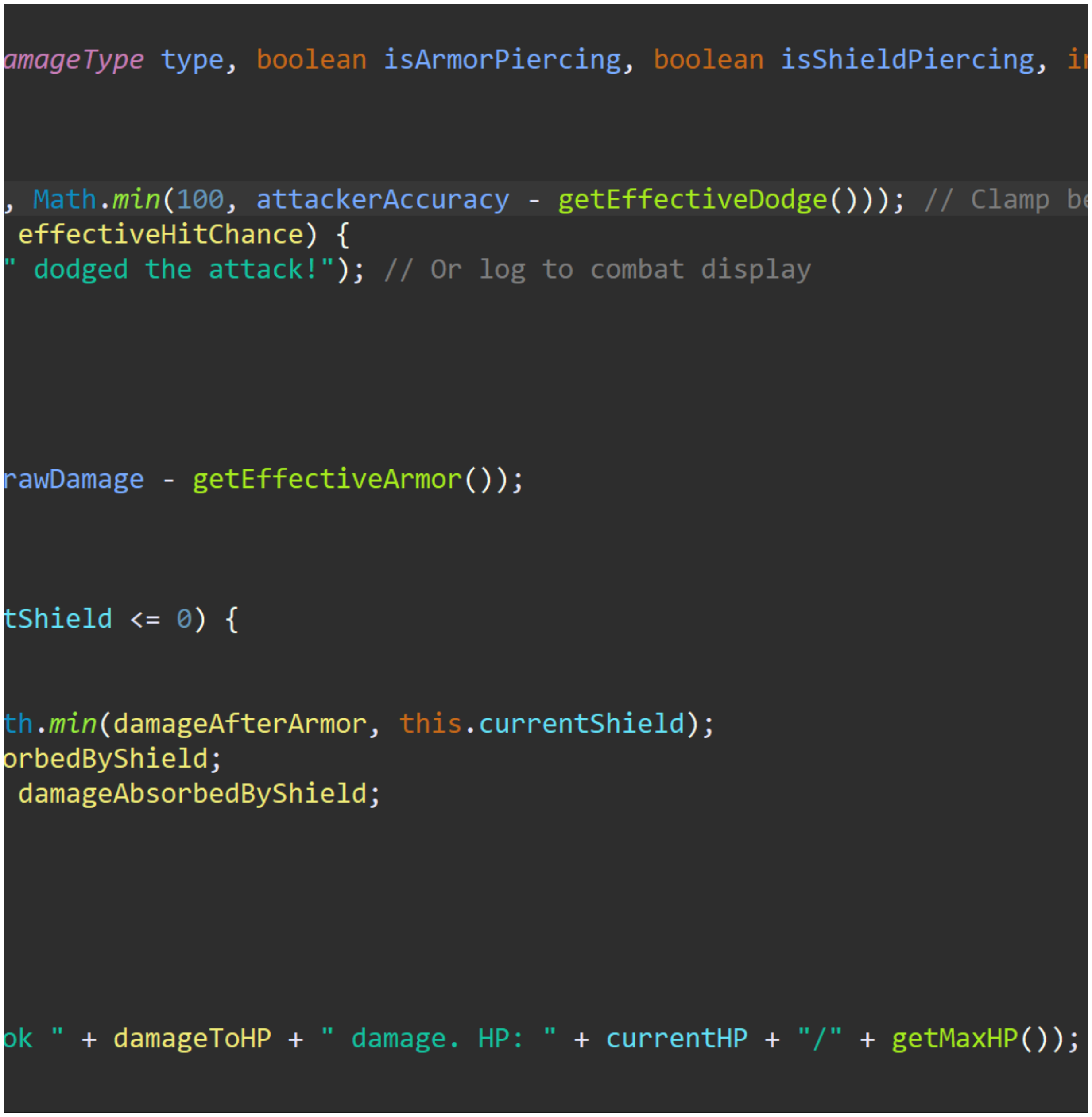


```
@Override
public void takeDamage(int rawDamage, Da
    if (!isAlive()) return;

    // 1. Dodge Calculation
    int effectiveHitChance = Math.max(5,
    if (new Random().nextInt(100) + 1 >
        System.out.println(getName() +
        return;
    }

    int damageAfterArmor = 0;
    if (!isArmorPiercing) {
        damageAfterArmor = Math.max(0,
    }

    int damageToHP = 0;
    if (isShieldPiercing || this.current
        damageToHP = damageAfterArmor;
    } else {
        int damageAbsorbedByShield = Ma
        this.currentShield -= damageAbs
        damageToHP = damageAfterArmor -
    }

    this.currentHP -= damageToHP;
    if (this.currentHP < 0) {
        this.currentHP = 0;
    }
    System.out.println(getName() + " to
}
```



Begin (player takes damage)
Is player hp < 0?
yes
end (player is already dead)
Calculate player dodge chance (dodge stat - enemy accuracy)
Roll a random number 1-100
Is random number > dodge chance?
no
end (player dodged the attack)
yes
Is enemy attack armor-piercing?
yes
no
Calculate armor damage reduction (incoming damage - armor)
yes
Is enemy attack shield-piercing?
no
Calculate shield mitigation (incoming damage - remaining shield)
Is there still damage remaining? (incoming dmg > 0)
yes
no
end (player has mitigated damage)
Deal damage to hp (currHP - incoming dmg)
end (player has taken damage)



```
amageType type, boolean isArmorPiercing, boolean isShieldPiercing, i



, Math.min(100, attackerAccuracy - getEffectiveDodge())); // Clamp be
 effectiveHitChance) {
" dodged the attack!"); // Or log to combat display






rawDamage - getEffectiveArmor());



tShield <= 0) {


th.min(damageAfterArmor, this.currentShield);
orbedByShield;
 damageAbsorbedByShield;







ok " + damageToHP + " damage. HP: " + currentHP + "/" + getMaxHP());
```