ETH zürich

Informatik I

Exercise session 10

Autumn 2019

Homework

- Questions?

Use case

Videogame

with skeletons and zombies.

Videogame

Videogame

You're writing a videogame. It has enemies: skeletons and zombies.
Each enemy has a certain amount of health
(integer health points).

Skeletons are vulnerable to being hit with a sword, but not with fire.
Zombies are only vulnerable to fire.

Step 1

Write an `abstract class Enemy` that supports:

- keeping track of health (`int health`);
- storing the enemy's name (`String name`);
- being dealt a sword hit (reduces the health by its parameter, `damage`);
- reporting info about itself (name, health, whether the enemy is still alive: health >= 0).

Step 1

```
abstract class Enemy {
    protected int health;
    private String name;

    public void dealDamage(int damage) {
        this.health -= damage;
    }

    public String getInfo() {
        return this.name + ": " + this.health + "HP, alive: " +
            (this.health > 0);
    }
}
```

Part 2

Write a constructor for `Enemy` that initialises `health` and `name`.

Part 2

```
abstract class Enemy {
    protected int health;
    private String name;

    protected Enemy(String name, int health) {
        this.name = name;
        this.health = health;
    }
```

Part 3

Write a `Skeleton` class that extends the `Enemy` class.
Its constructor should take a parameter that corresponds to the enemy's level (`int level`).
The name of this enemy should be "Level X skeleton".
And its health should be `5 * level`.

Part 3

```
class Skeleton extends Enemy {
    public Skeleton(int level) {
        super("Level " + level + " skeleton", level * 5);
    }
}
```

Part 4

Try it out in `Main`.

```
Skeleton skeleton = new Skeleton(2);
Out.println(skeleton.getInfo());
skeleton.dealDamage(10);
Out.println(skeleton.getInfo());
```

Part 5

Write a `Zombie` class that extends the `Enemy` class.
A zombie should not be vulnerable to normal attacks. It has 30 health points when spawned.
It can be dealt fire damage with the method `dealFireDamage`.

Part 5

```
class Zombie extends Enemy {
    public Zombie() {
        super("Generic zombie", 30);
    }

    @Override public void dealDamage(int damage) {
        // no damage
    }

    public void dealFireDamage(int power) {
        assert power >= 0 && power <= 10;
        this.health = this.health * power / 10 - power;
    }
}
```

Part 6

Try it out in `Main`.

```
Zombie zombie = new Zombie();
Out.println(zombie.getInfo());
zombie.dealDamage();
Out.println(zombie.getInfo());
zombie.dealFireDamage(5);
Out.println(zombie.getInfo());
```

Part 7

Assign both enemies to an `enemies` array (by leveraging polymorphism).
Then iterate over the array and call `getInfo` on each enemy.

Part 7

```
Enemy enemies[] = new Enemy[2];
enemies[0] = zombie;
enemies[1] = skeleton;

for (int i = 0; i < enemies.length; i++) {
    Out.println(enemies[i].getInfo());
}
```