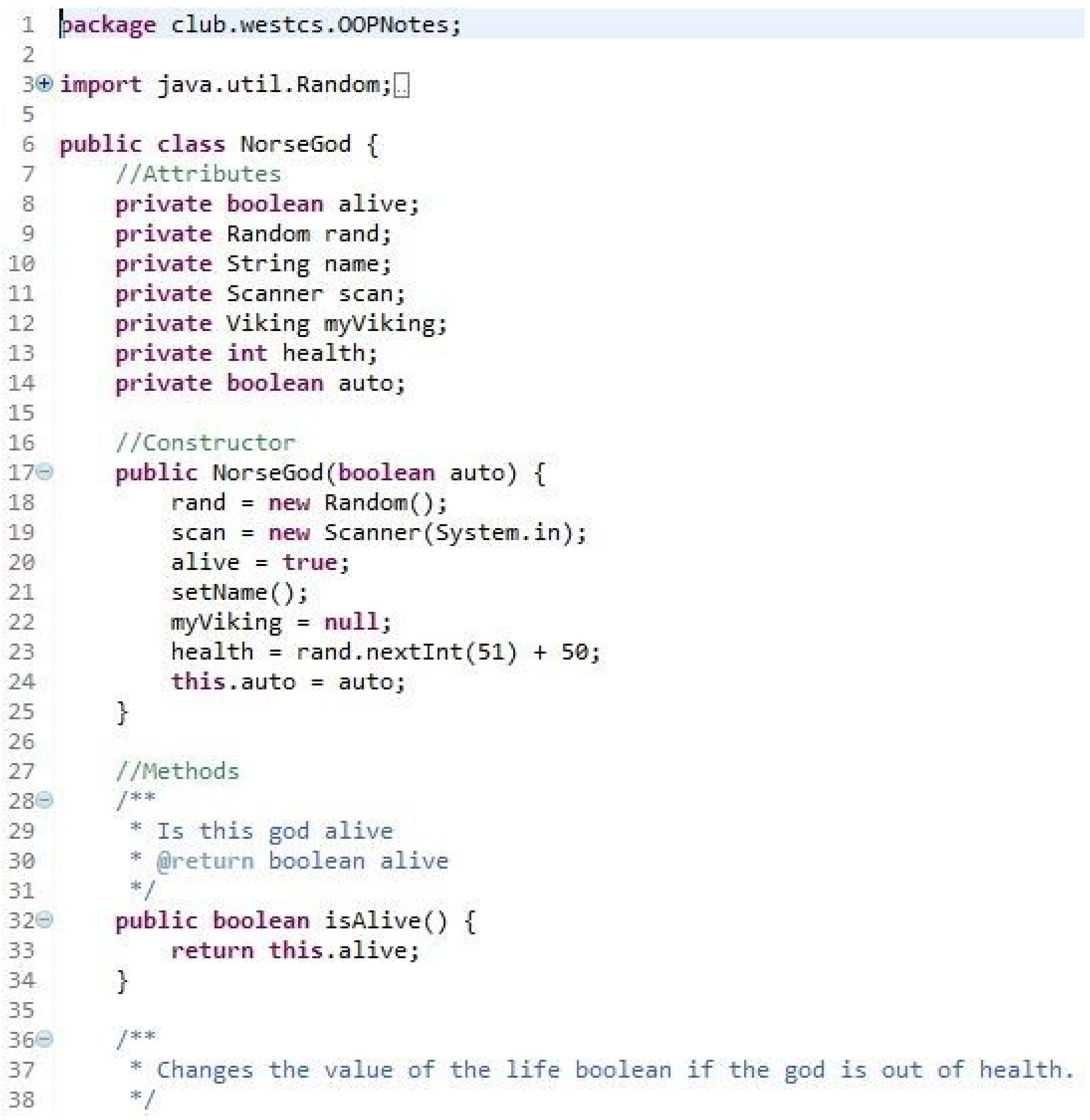
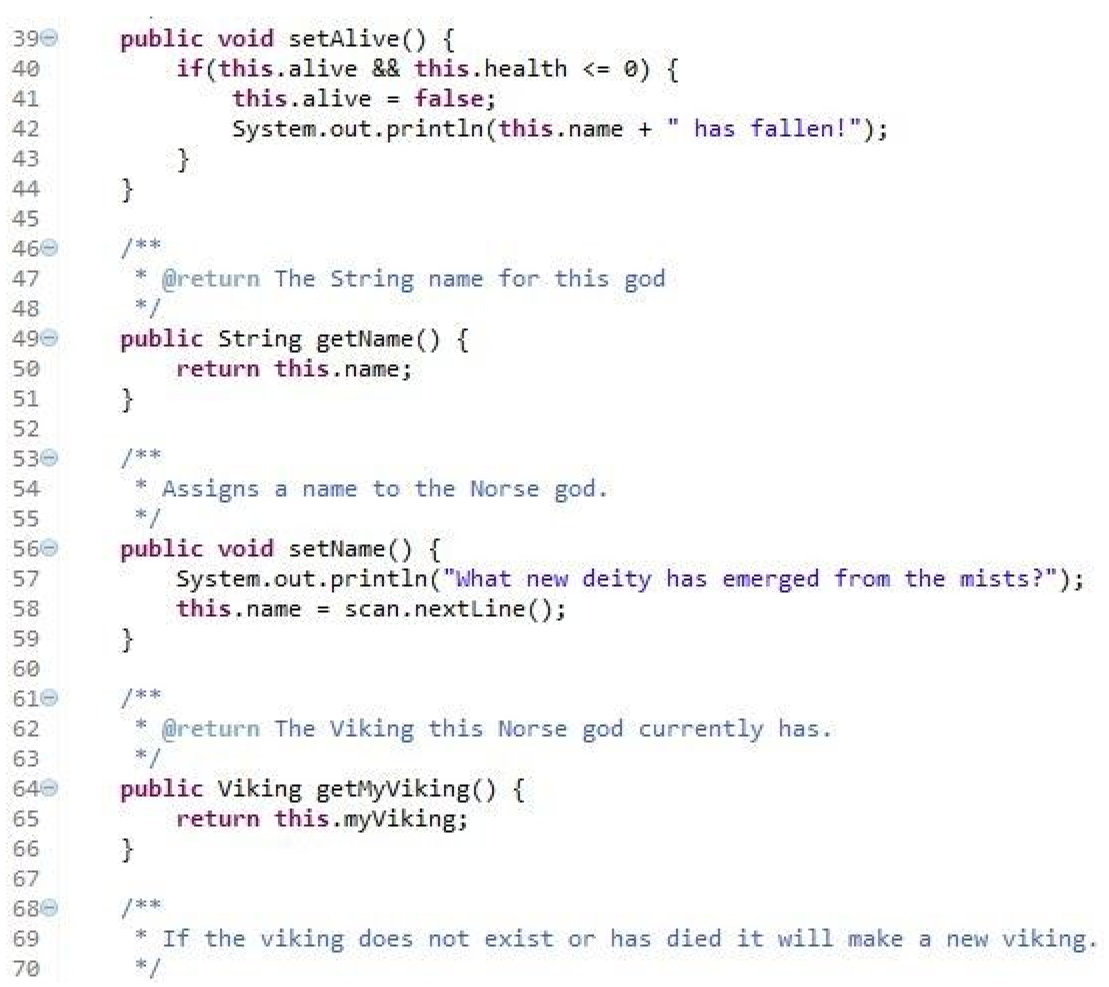
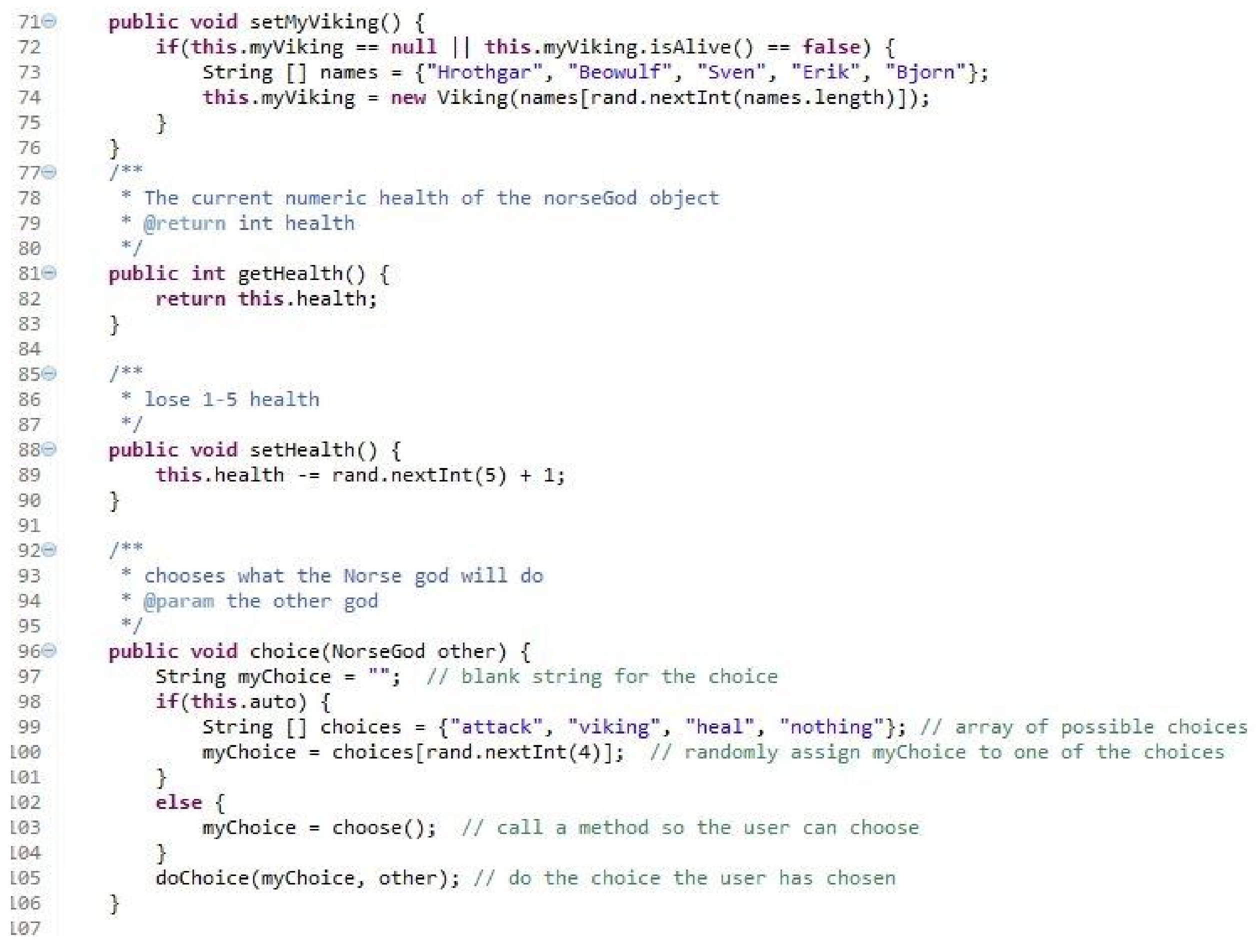


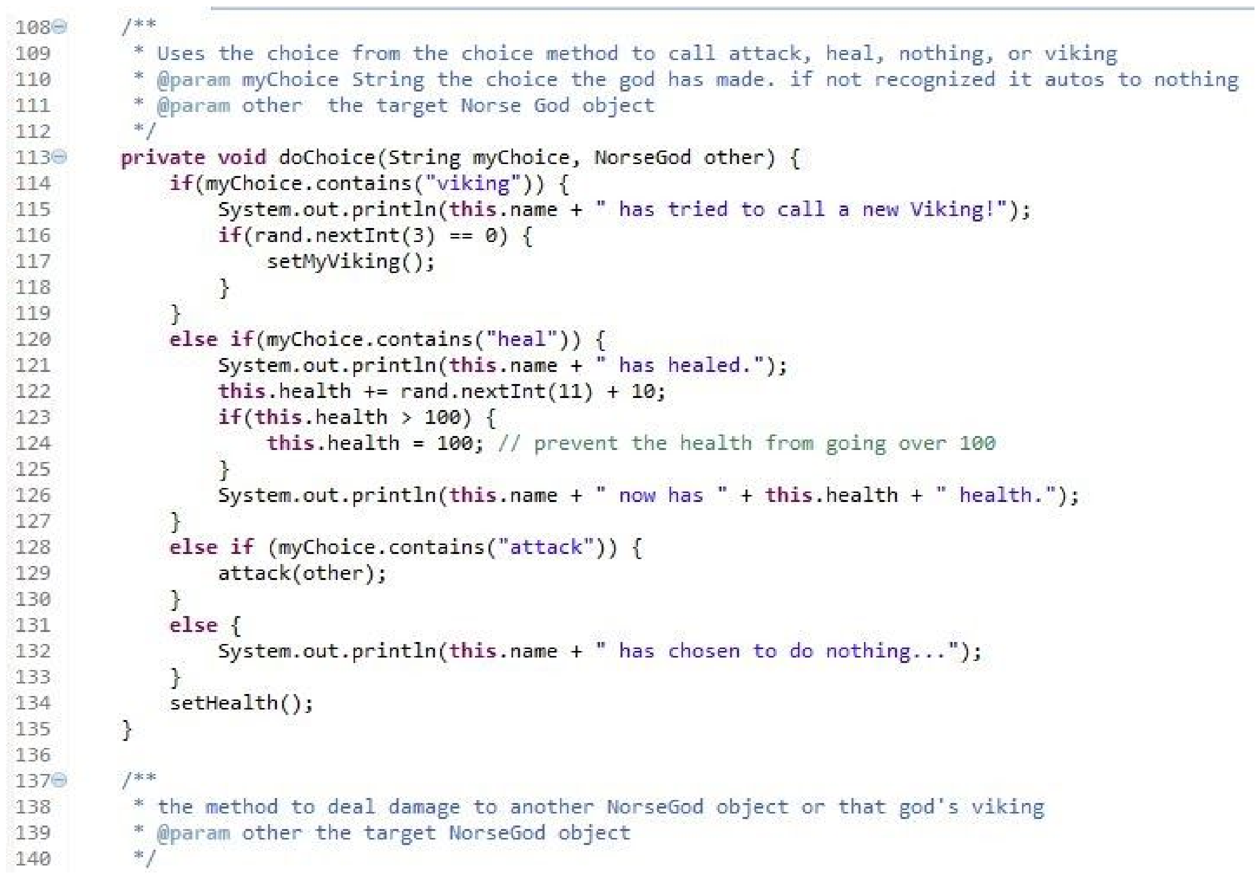
```java
package club.westcs.OOPNotes;

import java.util.Random;

public class NorseGod {
    //Attributes
    private boolean alive;
    private Random rand;
    private String name;
    private Scanner scan;
    private Viking myViking;
    private int health;
    private boolean auto;

    //Constructor
    public NorseGod(boolean auto) {
        rand = new Random();
        scan = new Scanner(System.in);
        alive = true;
        setName();
        myViking = null;
        health = rand.nextInt(51) + 50;
        this.auto = auto;
    }

    //Methods
    /**
     * Is this god alive
     * @return boolean alive
     */
    public boolean isAlive() {
        return this.alive;
    }

    /**
     * Changes the value of the life boolean if the god is out of health.
     */
```



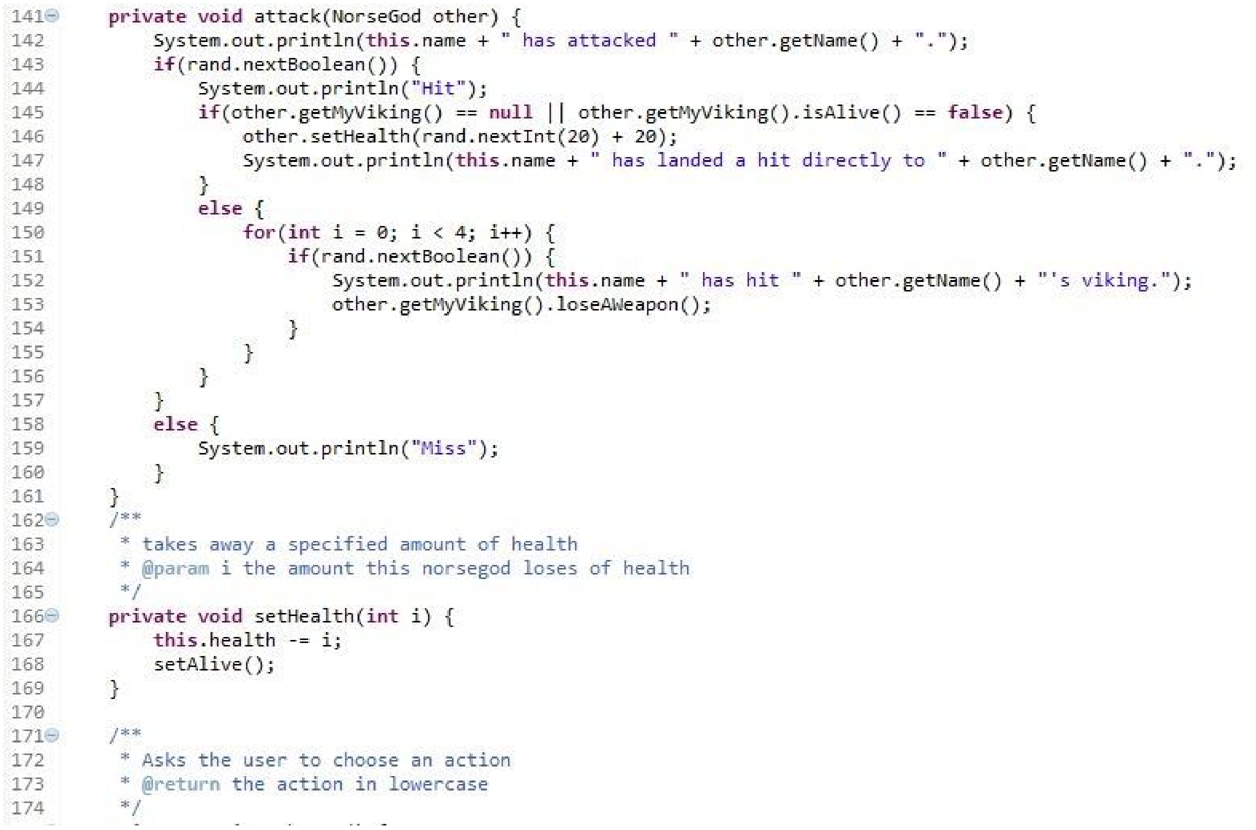
```java
    public void setAlive() {
        if(this.alive && this.health <= 0) {
            this.alive = false;
            System.out.println(this.name + " has fallen!");
        }
    }

    /**
     * @return The String name for this god
     */
    public String getName() {
        return this.name;
    }

    /**
     * Assigns a name to the Norse god.
     */
    public void setName() {
        System.out.println("What new deity has emerged from the mists?");
        this.name = scan.nextLine();
    }

    /**
     * @return The Viking this Norse god currently has.
     */
    public Viking getMyViking() {
        return this.myViking;
    }

    /**
     * If the viking does not exist or has died it will make a new viking.
     */
```



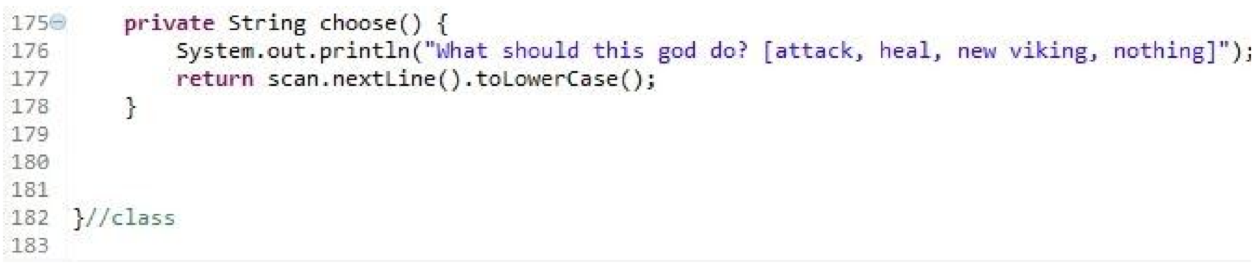
```java
    public void setMyViking() {
        if(this.myViking == null || this.myViking.isAlive() == false) {
            String [] names = {"Hrothgar", "Beowulf", "Sven", "Erik", "Bjorn"};
            this.myViking = new Viking(names[rand.nextInt(names.length)]);
        }
    }
    /**
     * The current numeric health of the norseGod object
     * @return int health
     */
    public int getHealth() {
        return this.health;
    }

    /**
     * lose 1-5 health
     */
    public void setHealth() {
        this.health -= rand.nextInt(5) + 1;
    }

    /**
     * chooses what the Norse god will do
     * @param the other god
     */
    public void choice(NorseGod other) {
        String myChoice = "";  // blank string for the choice
        if(this.auto) {
            String [] choices = {"attack", "viking", "heal", "nothing"}; // array of possible choices
            myChoice = choices[rand.nextInt(4)];  // randomly assign myChoice to one of the choices
        }
        else {
            myChoice = choose();  // call a method so the user can choose
        }
        doChoice(myChoice, other); // do the choice the user has chosen
    }

```



```java
    /**
     * Uses the choice from the choice method to call attack, heal, nothing, or viking
     * @param myChoice String the choice the god has made. if not recognized it autos to nothing
     * @param other  the target Norse God object
     */
    private void doChoice(String myChoice, NorseGod other) {
        if(myChoice.contains("viking")) {
            System.out.println(this.name + " has tried to call a new Viking!");
            if(rand.nextInt(3) == 0) {
                setMyViking();
            }
        }
        else if(myChoice.contains("heal")) {
            System.out.println(this.name + " has healed.");
            this.health += rand.nextInt(11) + 10;
            if(this.health > 100) {
                this.health = 100; // prevent the health from going over 100
            }
            System.out.println(this.name + " now has " + this.health + " health.");
        }
        else if (myChoice.contains("attack")) {
            attack(other);
        }
        else {
            System.out.println(this.name + " has chosen to do nothing...");
        }
        setHealth();
    }

    /**
     * the method to deal damage to another NorseGod object or that god's viking
     * @param other the target NorseGod object
     */
```



```java
    private void attack(NorseGod other) {
        System.out.println(this.name + " has attacked " + other.getName() + ".");
        if(rand.nextBoolean()) {
            System.out.println("Hit");
            if(other.getMyViking() == null || other.getMyViking().isAlive() == false) {
                other.setHealth(rand.nextInt(20) + 20);
                System.out.println(this.name + " has landed a hit directly to " + other.getName() + ".");
            }
            else {
                for(int i = 0; i < 4; i++) {
                    if(rand.nextBoolean()) {
                        System.out.println(this.name + " has hit " + other.getName() + "'s viking.");
                        other.getMyViking().loseAWeapon();
                    }
                }
            }
        }
        else {
            System.out.println("Miss");
        }
    }
    /**
     * takes away a specified amount of health
     * @param i the amount this norsegod loses of health
     */
    private void setHealth(int i) {
        this.health -= i;
        setAlive();
    }

    /**
     * Asks the user to choose an action
     * @return the action in lowercase
     */
```



```java
    private String choose() {
        System.out.println("What should this god do? [attack, heal, new viking, nothing]");
        return scan.nextLine().toLowerCase();
    }



}//class
```