Vikings vs Saxons Battle Simulation

by: Ginger Spice

Group members: Marc, Mirko, Zhoubin

Index:

- Project Description
- Libraries & Tools Overview
- Code Focus
- Demonstration & Results
- Q/A

Project Description

This project utilizes **inheritance** in Python to simulate a battle between **Vikings** and **Saxons**. The game implements a **war simulation** where Vikings and Saxons engage in combat, with each side attacking and receiving damage based on their attributes. The game continues until one side emerges victorious.

Libraries & Tools Overview

Colorama Library

- **Purpose**: Enables color and style customization for text output in the terminal.

Fore Module (Colorama)

- **Function**: Controls the foreground (text) colors within the Colorama library.

Pygame Library

- **Purpose**: A suite of Python modules designed for creating video games, featuring tools for computer graphics, sound, and music.

pygame.mixer

- **Function**: Manages sound and music in Pygame, allowing you to load and play audio files.

Time Library

- **Purpose**: Provides functions for working with time-related tasks, such as introducing delays.

Freesound.org

- **Purpose**: An online platform for creating and sharing audio files, useful for generating sounds for games and applications.
  Freesound.org

Code Focus

```python
# Soldier
class Soldier:
    def __init__(self, health, strength):
        self.health = health
        self.strength = strength

    def attack(self):
        return self.strength

    def receiveDamage(self, damage):
        self.health -= damage
        if self.health <= 0:
            return None
        return None
```

```python
# Viking
class Viking(Soldier):
    def __init__(self, name, health, strength):
        super().__init__(health, strength)
        self.name = name

    def battleCry(self):
        return "Odin Owns You All!"

    def receiveDamage(self, damage):
        self.health -= damage
        if self.health <= 0:
            return f"{self.name} has died in act of combat"
        return f"{self.name} has received {damage} points of damage"
```

Code Focus

```
# Saxon
class Saxon(Soldier):
    def __init__(self, health, strength):
        super().__init__(health, strength)

    def receiveDamage(self, damage):
        self.health -= damage
        if self.health <= 0:
            return "A Saxon has died in combat"
        return f"A Saxon has received {damage} points of damage"
```

Code Focus

```python
# War Class
class War:
    def __init__(self):
        self.vikingArmy = []
        self.saxonArmy = []

    def addViking(self, viking):
        self.vikingArmy.append(viking)

    def addSaxon(self, saxon):
        self.saxonArmy.append(saxon)

    def vikingAttack(self):
        if len(self.saxonArmy) == 0:
            return "No Saxons left to attack."
        saxon = random.choice(self.saxonArmy)
        viking = random.choice(self.vikingArmy)
        damage = viking.attack()
        result = saxon.receiveDamage(damage)
        if saxon.health <= 0:
            self.saxonArmy.remove(saxon)
        return result

    def saxonAttack(self):
        if len(self.vikingArmy) == 0:
            return "No Vikings left to attack."
        if len(self.saxonArmy) == 0:
            return "No Saxons left to attack."
        saxon = random.choice(self.saxonArmy)
        viking = random.choice(self.vikingArmy)
        damage = saxon.attack()
        result = viking.receiveDamage(damage)
        if viking.health <= 0:
            self.vikingArmy.remove(viking)
        return result

    def showStatus(self):
        if len(self.saxonArmy) == 0:
            return "Vikings have won the war of the century!"
        elif len(self.vikingArmy) == 0:
            return "Saxons have fought for their lives and survive another day..."
        return "Vikings and Saxons are still in the thick of battle."
```

DEMO

Q/A and conclusion

Questions are welcome!

Ginger Spice

Marc, Mirko, Zhoubin

Thank you for your attention!