import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.Random;  
  
abstract class Fighter {  
 protected String name;  
 protected int health;  
 protected int attackPower;  
 protected int maxHealth;  
  
  
 public Fighter(String name, int health, int attackPower) {  
 this.name = name;  
 this.health = health;  
 this.maxHealth = health;  
 this.attackPower = attackPower;  
 }  
  
  
  
 public void takeDamage(int damage) {  
 this.health -= damage;  
 if (this.health < 0) {  
 this.health = 0;  
 }  
 }  
  
  
 public boolean isAlive() {  
 return health > 0;  
 }  
  
  
 public void attack(Fighter opponent) {  
 int damage = attackPower;  
 opponent.takeDamage(damage);  
 }  
  
 public abstract void specialMove(Fighter opponent);  
  
 public abstract String getSpecialMoveName();  
  
  
 public String getName() {  
 return name;  
 }  
}  
  
  
class ShakaZulu extends Fighter {  
 public ShakaZulu() {  
 super("Shaka Zulu", 120, 15);  
 }  
  
 @Override  
 public void specialMove(Fighter opponent) {  
 int damage = 30;  
 opponent.takeDamage(damage);  
 CombatGame.*logEvent*(this.name + " unleashes his Iklwa Strike, dealing " + damage + " massive damage!");  
 }  
  
 @Override  
 public String getSpecialMoveName() {  
 return "Iklwa Strike";  
 }  
}  
  
  
class QueenNzinga extends Fighter {  
 public QueenNzinga() {  
 super("Queen Nzinga", 100, 20);  
 }  
  
 @Override  
 public void specialMove(Fighter opponent) {  
 int damage = 10;  
 int healthDrain = 5;  
 opponent.takeDamage(damage);  
 this.health += healthDrain;  
 CombatGame.*logEvent*(this.name + " performs a Strategic Retreat, dealing " + damage + " damage and regaining " + healthDrain + " health!");  
 }  
  
 @Override  
 public String getSpecialMoveName() {  
 return "Strategic Retreat";  
 }  
}  
  
  
class MansaMusa extends Fighter {  
 public MansaMusa() {  
 super("Mansa Musa", 150, 10);  
 }  
  
 @Override  
 public void specialMove(Fighter opponent) {  
 int healthHeal = 40;  
 this.health += healthHeal;  
 if (this.health > this.maxHealth) {  
 this.health = this.maxHealth;  
 }  
 CombatGame.*logEvent*(this.name + " distributes Golden Riches, healing for " + healthHeal + " health!");  
 }  
  
 @Override  
 public String getSpecialMoveName() {  
 return "Golden Riches";  
 }  
}  
  
class Tutankhamun extends Fighter {  
 public Tutankhamun() {  
 super("Tutankhamun", 90, 25);  
 }  
  
 @Override  
 public void specialMove(Fighter opponent) {  
 Random rand = new Random();  
 if (rand.nextBoolean()) {  
 int damage = 50;  
 opponent.takeDamage(damage);  
 CombatGame.*logEvent*(this.name + " unleashes the Curse of the Pharaoh, dealing a devastating " + damage + " damage!");  
 } else {  
 CombatGame.*logEvent*(this.name + "'s Curse of the Pharaoh fails to land!");  
 }  
 }  
  
 @Override  
 public String getSpecialMoveName() {  
 return "Curse of the Pharaoh";  
 }  
}  
  
class Hannibal extends Fighter {  
 public Hannibal() {  
 super("Hannibal", 110, 18);  
 }  
  
 @Override  
 public void specialMove(Fighter opponent) {  
 int hits = 3;  
 int damagePerHit = 8;  
 int totalDamage = hits \* damagePerHit;  
 opponent.takeDamage(totalDamage);  
 CombatGame.*logEvent*(this.name + " commands an Elephant Charge, trampling " + opponent.getName() + " for " + hits + " hits, dealing a total of " + totalDamage + " damage!");  
 }  
  
 @Override  
 public String getSpecialMoveName() {  
 return "Elephant Charge";  
 }  
}  
  
  
public class CombatGame {  
  
 private static JFrame *frame*;  
 private static JTextArea *combatLog*;  
 private static JProgressBar *player1HealthBar*;  
 private static JProgressBar *player2HealthBar*;  
 private static JLabel *player1NameLabel*;  
 private static JLabel *player2NameLabel*;  
  
 private static Fighter *player1*;  
 private static Fighter *player2*;  
 private static Fighter *currentPlayer*;  
 private static Fighter *opponent*;  
  
 */\*\*  
 \* The main method to start the GUI.  
 \*/* public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(() -> {  
 new CombatGame().initializeGUI();  
 });  
 }  
  
 private void initializeGUI() {  
 *frame* = new JFrame("African Legends Combat");  
 *frame*.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 *frame*.setSize(800, 600);  
 *frame*.setLayout(new BorderLayout(10, 10));  
  
 // Create and set up the main game panel  
 JPanel gamePanel = new JPanel();  
 gamePanel.setLayout(new GridBagLayout());  
 gamePanel.setBackground(new Color(24, 24, 24)); // Dark background  
 GridBagConstraints gbc = new GridBagConstraints();  
 gbc.insets = new Insets(10, 10, 10, 10);  
  
 // Fighter Selection Screen  
 JPanel selectionPanel = createSelectionPanel();  
 *frame*.add(selectionPanel, BorderLayout.*CENTER*);  
 *frame*.setVisible(true);  
 }  
  
  
 private JPanel createSelectionPanel() {  
 JPanel panel = new JPanel(new GridLayout(2, 3, 20, 20));  
 panel.setBackground(new Color(24, 24, 24));  
 panel.setBorder(BorderFactory.*createEmptyBorder*(20, 20, 20, 20));  
  
 Fighter[] fighters = {  
 new ShakaZulu(), new QueenNzinga(), new MansaMusa(), new Tutankhamun(), new Hannibal()  
 };  
  
 for (Fighter f : fighters) {  
 JButton button = new JButton(f.getName());  
 button.setFont(new Font("Arial", Font.*BOLD*, 16));  
 button.setBackground(new Color(50, 50, 50));  
 button.setForeground(Color.*WHITE*);  
 button.setFocusPainted(false);  
 button.addActionListener(e -> selectFighter(f));  
 panel.add(button);  
 }  
  
 return panel;  
 }  
  
  
 private void selectFighter(Fighter fighter) {  
 if (*player1* == null) {  
 *player1* = fighter;  
 JOptionPane.*showMessageDialog*(*frame*, *player1*.getName() + " selected! Player 2, choose your fighter.");  
 } else if (*player2* == null) {  
 *player2* = fighter;  
 JOptionPane.*showMessageDialog*(*frame*, "Player 2 selected! The combat begins!");  
 startGame();  
 }  
 }  
  
 private void startGame() {  
 // Remove selection panel and add combat panel  
 *frame*.getContentPane().removeAll();  
  
 JPanel combatPanel = new JPanel(new BorderLayout(10, 10));  
 combatPanel.setBackground(new Color(24, 24, 24));  
  
 // Fighter display and health bars  
 JPanel fighterDisplayPanel = new JPanel(new GridLayout(1, 2, 20, 20));  
 fighterDisplayPanel.setBackground(new Color(24, 24, 24));  
 fighterDisplayPanel.add(createFighterPanel(*player1*));  
 fighterDisplayPanel.add(createFighterPanel(*player2*));  
 combatPanel.add(fighterDisplayPanel, BorderLayout.*NORTH*);  
  
 // Combat Log  
 *combatLog* = new JTextArea(10, 30);  
 *combatLog*.setEditable(false);  
 *combatLog*.setBackground(new Color(35, 35, 35));  
 *combatLog*.setForeground(Color.*WHITE*);  
 *combatLog*.setBorder(BorderFactory.*createLineBorder*(new Color(70, 70, 70)));  
 JScrollPane logScrollPane = new JScrollPane(*combatLog*);  
 combatPanel.add(logScrollPane, BorderLayout.*CENTER*);  
  
 // Action Buttons  
 JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.*CENTER*, 20, 10));  
 buttonPanel.setBackground(new Color(24, 24, 24));  
 JButton attackButton = new JButton("Basic Attack");  
 JButton specialButton = new JButton("Special Move");  
  
 attackButton.addActionListener(e -> performMove("attack"));  
 specialButton.addActionListener(e -> performMove("special"));  
  
 buttonPanel.add(attackButton);  
 buttonPanel.add(specialButton);  
 combatPanel.add(buttonPanel, BorderLayout.*SOUTH*);  
  
 *frame*.add(combatPanel, BorderLayout.*CENTER*);  
 *frame*.revalidate();  
 *frame*.repaint();  
  
 *currentPlayer* = *player1*;  
 *opponent* = *player2*;  
 *logEvent*("Welcome, " + *player1*.getName() + " and " + *player2*.getName() + "!");  
 *logEvent*("It's " + *currentPlayer*.getName() + "'s turn!");  
 updateHealthBars();  
 }  
  
 private JPanel createFighterPanel(Fighter fighter) {  
 JPanel panel = new JPanel(new BorderLayout(5, 5));  
 panel.setBackground(new Color(40, 40, 40));  
 panel.setBorder(BorderFactory.*createLineBorder*(new Color(100, 100, 100)));  
  
 JLabel nameLabel = new JLabel(fighter.getName(), SwingConstants.*CENTER*);  
 nameLabel.setFont(new Font("Arial", Font.*BOLD*, 18));  
 nameLabel.setForeground(Color.*YELLOW*);  
  
 JProgressBar healthBar = new JProgressBar(0, fighter.maxHealth);  
 healthBar.setValue(fighter.health);  
 healthBar.setStringPainted(true);  
 healthBar.setBackground(new Color(70, 70, 70));  
 healthBar.setForeground(Color.*RED*);  
  
 if (fighter == *player1*) {  
 *player1NameLabel* = nameLabel;  
 *player1HealthBar* = healthBar;  
 } else {  
 *player2NameLabel* = nameLabel;  
 *player2HealthBar* = healthBar;  
 }  
  
 panel.add(nameLabel, BorderLayout.*NORTH*);  
 panel.add(healthBar, BorderLayout.*CENTER*);  
 return panel;  
 }  
  
 private void performMove(String moveType) {  
 if (!*player1*.isAlive() || !*player2*.isAlive()) {  
 return; // Game is over  
 }  
  
 if (moveType.equals("attack")) {  
 *currentPlayer*.attack(*opponent*);  
 *logEvent*(*currentPlayer*.getName() + " performs a basic attack.");  
 } else if (moveType.equals("special")) {  
 *currentPlayer*.specialMove(*opponent*);  
 }  
  
 updateHealthBars();  
  
 // Check for winner  
 if (!*player1*.isAlive() || !*player2*.isAlive()) {  
 endGame();  
 } else {  
 // Swap players for the next turn  
 Fighter temp = *currentPlayer*;  
 *currentPlayer* = *opponent*;  
 *opponent* = temp;  
 *logEvent*("It's " + *currentPlayer*.getName() + "'s turn!");  
 }  
 }  
  
  
 private void updateHealthBars() {  
 *player1HealthBar*.setValue(*player1*.health);  
 *player1HealthBar*.setString(*player1*.health + "/" + *player1*.maxHealth);  
 *player2HealthBar*.setValue(*player2*.health);  
 *player2HealthBar*.setString(*player2*.health + "/" + *player2*.maxHealth);  
 }  
  
  
 public static void logEvent(String message) {  
 *combatLog*.append(message + "\n");  
 *combatLog*.setCaretPosition(*combatLog*.getDocument().getLength());  
 }  
  
   
 private void endGame() {  
 String winnerName = *player1*.isAlive() ? *player1*.getName() : *player2*.getName();  
 *logEvent*("\n----------------------------");  
 *logEvent*("Game Over! " + winnerName + " wins!");  
 JOptionPane.*showMessageDialog*(*frame*, "Game Over! " + winnerName + " wins!");  
 // Disable buttons to prevent further moves  
 for (Component comp : ((JPanel)((BorderLayout)*frame*.getContentPane().getLayout()).getLayoutComponent(BorderLayout.*SOUTH*)).getComponents()) {  
 if (comp instanceof JButton) {  
 ((JButton) comp).setEnabled(false);  
 }  
 }  
 }  
}