For this exercise, implement the following steps:

1. Navigate to https://www.gamesforthebrain.com/game/checkers/

2. Confirm that the site is up

3. Make five legal moves as orange:

a) Include taking a blue piece

b) Use “Make a move” as confirmation that you can take the next step

c) Restart the game after five moves

d) Confirm that the restarting had been successful

**package** com.example;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.openqa.selenium.interactions.Actions;

**import** org.openqa.selenium.support.ui.ExpectedConditions;

**import** org.openqa.selenium.support.ui.WebDriverWait;

**import** org.testng.Assert;

**import** org.testng.annotations.AfterClass;

**import** org.testng.annotations.BeforeClass;

**import** org.testng.annotations.Test;

**import** java.time.Duration;

**import** java.util.List;

**public** **class** CheckerMoveGames {

WebDriver driver;

WebDriverWait wait;

Actions actions;

@BeforeClass

**public** **void** setup() {

System.*setProperty*("webdriver.chrome.driver", "C:\\webDriver\\chromedriver.exe");

driver = **new** ChromeDriver();

driver.manage().window().maximize();

wait = **new** WebDriverWait(driver, Duration.*ofSeconds*(15));

actions = **new** Actions(driver);

}

@Test

**public** **void** testCheckersGameMoves() {

**try** {

driver.get("https://www.gamesforthebrain.com/game/checkers/");

wait.until(ExpectedConditions.*titleContains*("Checkers"));

Assert.*assertTrue*(driver.getTitle().contains("Checkers"));

WebElement gameBoard = wait.until(ExpectedConditions.*presenceOfElementLocated*(By.*id*("board")));

Assert.*assertNotNull*(gameBoard, "Game board should be present");

WebElement initialMessage = wait.until(ExpectedConditions. *visibilityOfElementLocated*(By.*id*("message")));

System.***out***.println("Initial message: " + initialMessage.getText());

makeVisualMoves();

WebElement restartBtn = wait.until(ExpectedConditions.*elementToBeClickable*(By.*linkText*("Restart...")));

restartBtn.click();

WebElement msg = wait.until(ExpectedConditions. *visibilityOfElementLocated*(By.*id*("message")));

Assert.*assertTrue*(msg.getText().contains("Make a move") || msg.getText().contains("Select"));

} **catch** (Exception e) {

System.***err***.println("Test failed with exception: " + e.getMessage());

printPageStructure();

**throw** e;

}

}

**private** **void** makeVisualMoves() {

**try** {

Thread.*sleep*(2000);

List<WebElement> orangePieces = driver.findElements(By.*cssSelector*("img[src\*='you1.gif']"));

System.***out***.println("Found " + orangePieces.size() + " orange pieces");

**if** (orangePieces.isEmpty()) {

orangePieces = driver.findElements(By.*cssSelector*("img[src\*='you']"));

System.***out***.println("Found " + orangePieces.size() + " pieces with alternative selector");

}

**int** movesMade = 0;

**int** maxMoves = Math.*min*(5, orangePieces.size());

**for** (**int** i = 0; i < maxMoves && movesMade < 5; i++) {

**try** {

WebElement piece = orangePieces.get(i);

wait.until(ExpectedConditions.*elementToBeClickable*(piece));

System.***out***.println("Clicking piece " + (i + 1));

piece.click();

Thread.*sleep*(1000);

WebElement targetSquare = findValidMove(piece);

**if** (targetSquare != **null**) {

System.***out***.println("Moving to target square");

actions.moveToElement(targetSquare)

.pause(500)

.click()

.perform();

Thread.*sleep*(2000);

WebElement message = driver.findElement(By.*id*("message"));

System.***out***.println("Move " + (movesMade + 1) + " message: " + message.getText());

**if** (message.getText().contains("Make a move") ||

message.getText().contains("Select") ||

message.getText().contains("Please wait")) {

movesMade++;

}

} **else** {

System.***out***.println("No valid move found for piece " + (i + 1));

}

} **catch** (Exception e) {

System.***out***.println("Move " + (i + 1) + " failed: " + e.getMessage());

}

}

System.***out***.println("Successfully made " + movesMade + " visual moves");

} **catch** (Exception e) {

System.***err***.println("Error making visual moves: " + e.getMessage());

}

}

**private** WebElement findValidMove(WebElement piece) {

**try** {

String pieceSrc = piece.getAttribute("src");

List<WebElement> emptySquares = driver.findElements(By.*cssSelector*("img[src\*='gray.gif']"));

**for** (WebElement square : emptySquares) {

**try** {

**if** (square.isDisplayed() && square.isEnabled()) {

**return** square;

}

} **catch** (Exception e) {

}

}

List<WebElement> allSquares = driver.findElements(By.*cssSelector*("td img"));

**for** (WebElement square : allSquares) {

**try** {

String src = square.getAttribute("src");

**if** (src != **null** && (src.contains("gray.gif") || src.contains("space"))) {

**return** square;

}

} **catch** (Exception e) {

}

}

} **catch** (Exception e) {

System.***err***.println("Error finding valid move: " + e.getMessage());

}

**return** **null**;

}

**private** **void** printPageStructure() {

**try** {

System.***out***.println("=== PAGE DEBUG INFO ===");

System.***out***.println("Page title: " + driver.getTitle());

System.***out***.println("Current URL: " + driver.getCurrentUrl());

List<WebElement> boardElements = driver.findElements(By.*id*("board"));

System.***out***.println("Board elements found: " + boardElements.size());

List<WebElement> allImages = driver.findElements(By.*tagName*("img"));

System.***out***.println("Total images found: " + allImages.size());

**for** (**int** i = 0; i < Math.*min*(10, allImages.size()); i++) {

WebElement img = allImages.get(i);

System.***out***.println("Image " + i + " src: " + img.getAttribute("src"));

}

List<WebElement> messageElements = driver.findElements(By.*id*("message"));

**if** (!messageElements.isEmpty()) {

System.***out***.println("Message: " + messageElements.get(0).getText());

}

} **catch** (Exception e) {

System.***err***.println("Error printing page structure: " + e.getMessage());

}

}

@AfterClass

**public** **void** teardown() {

**if** (driver != **null**) {

driver.quit();

}

}

}