Logo, company name

Description automatically generated

**COMSATS University Islamabad**

**Abbottabad Campus**

**Department of Computer Science**

**Lab mid term**

**Submitted by : Sadat Mumtaz khan**

**Registration no : FA20-BSE-011**

**Submitted to : Mukhtiar Zamin**

package observer;

public interface MatchObserver {

void update(String matchStatus);

}

public interface MatchSubject {

void registerObserver(MatchObserver observer);

void removeObserver(MatchObserver observer);

void notifyObservers();

String getMatchDetails();

}

­­

import java.util.Random;

public class MatchLiveUpdatesApp {

public static void main(String[] args) {

// Creating instances

CricketFan fan1 = new CricketFan("John");

CricketFan fan2 = new CricketFan("Alice");

LiveMatchScreen mainScreen = new LiveMatchScreen();

//random matches

for (int i = 0; i < 5; i++) {

CricketMatch randomMatch = createRandomMatch("Match " + (i + 1));

mainScreen.addLiveMatch(randomMatch);

}

mainScreen.selectMatchFromMenu();

//match updates

for (int i = 0; i < 3; i++) {

CricketMatch randomMatch = mainScreen.getRandomMatch();

randomMatch.setMatchStatus("In Progress - Score: " + getRandomScore());

}

mainScreen.selectMatchFromMenu();

}

private static CricketMatch createRandomMatch(String name) {

CricketMatch match = new CricketMatch();

match.setMatchStatus(name + ": Yet to Start");

return match;

}

private static String getRandomScore() {

Random random = new Random();

int runs = random.nextInt(200);

int wickets = random.nextInt(10);

return runs + "/" + wickets;

}

}

import java.util.ArrayList;

import java.util.List;

import java.util.Random;

import java.util.Scanner;

public class LiveMatchScreen {

private List<CricketMatch> liveMatches = new ArrayList<>();

public void displayLiveMatches() {

System.out.println("Live Matches:");

for (int i = 0; i < liveMatches.size(); i++) {

System.out.println((i + 1) + ". " + liveMatches.get(i).getMatchDetails());

}

}

public void addLiveMatch(CricketMatch match) {

liveMatches.add(match);

}

public void selectMatchFromMenu() {

Scanner scanner = new Scanner(System.in);

displayLiveMatches();

System.out.print("Select a match (enter the corresponding number): ");

int selection = scanner.nextInt();

if (selection > 0 && selection <= liveMatches.size()) {

CricketMatch selectedMatch = liveMatches.get(selection - 1);

System.out.println("User selected match: " + selectedMatch.getMatchDetails());

// Simulate navigating to ball-by-ball coverage screen

BallByBallCoverageScreen coverageScreen = new BallByBallCoverageScreen(selectedMatch);

coverageScreen.displayBallByBallCoverage();

} else {

System.out.println("Invalid selection. Please try again.");

}

}

public CricketMatch getRandomMatch() {

Random random = new Random();

int randomIndex = random.nextInt(liveMatches.size());

return liveMatches.get(randomIndex);

}

}

import java.util.ArrayList;

import java.util.List;

public class CricketMatch implements MatchSubject {

private String matchStatus;

private List<MatchObserver> observers = new ArrayList<>();

@Override

public void registerObserver(MatchObserver observer) {

observers.add(observer);

}

@Override

public void removeObserver(MatchObserver observer) {

observers.remove(observer);

}

@Override

public void notifyObservers() {

for (MatchObserver observer : observers) {

observer.update(matchStatus);

}

}

public void setMatchStatus(String matchStatus) {

this.matchStatus = matchStatus;

notifyObservers();

}

@Override

public String getMatchDetails() {

return "Match Details: " + matchStatus;

}

}

public class CricketFan implements MatchObserver {

private String name;

public CricketFan(String name) {

this.name = name;

}

@Override

public void update(String matchStatus) {

System.out.println(name + " received match update: " + matchStatus);

}

}

public class BallByBallCoverageScreen {

private CricketMatch selectedMatch;

public BallByBallCoverageScreen(CricketMatch selectedMatch) {

this.selectedMatch = selectedMatch;

}

public void displayBallByBallCoverage() {

System.out.println("Ball-by-Ball Coverage for match: " + selectedMatch.getMatchDetails());

for (int ballNumber = 1; ballNumber <= 10; ballNumber++) {

String ballUpdate = simulateBallUpdate(ballNumber);

System.out.println(ballUpdate);

selectedMatch.setMatchStatus("In Progress - " + ballUpdate);

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

//match completion

selectedMatch.setMatchStatus("Match Completed - Final Score: " + getRandomScore());

}

private String simulateBallUpdate(int ballNumber) {

Random random = new Random();

int runs = random.nextInt(7);

int wickets = random.nextInt(2);

return "Ball " + ballNumber + ": " + runs + " runs, " + wickets + " wickets";

}

private String getRandomScore() {

Random random = new Random();

int runs = random.nextInt(200);

int wickets = random.nextInt(10);

return runs + "/" + wickets;

}

}