**PROGRAM 4 POLLS – AMAAN ASLAM**

**Main Class**

import java.io.File;

import java.io.FileNotFoundException;

import java.util.HashMap;

import java.util.Scanner;

public class ObserverPattern {

public static void main(String[] args) throws FileNotFoundException {

File myfile = new File("teams.txt");

Scanner sc = new Scanner(myfile);

String line;

HashMap<String, Integer> teams = new HashMap<String, Integer>();

while(sc.hasNextLine()) {

line = sc.nextLine();

teams.put(line, 0);

}

sc.close();

AssociatedPress ap = new AssociatedPress();

USAToday ut = new USAToday();

Espn espn = new Espn(ap);

Espn espn1 = new Espn(ut);

LATimes la1 = new LATimes(ap);

LATimes la2 = new LATimes(ut);

SportsWeekly sw1 = new SportsWeekly(ap);

SportsWeekly sw2 = new SportsWeekly(ut);

WashingtonPost wp1 = new WashingtonPost(ap);

WashingtonPost wp2 = new WashingtonPost(ut);

for(int i=0; i<=99; i++) {

Voters voter = new Voters();

if(i<=40) {

voter.addObserver(ap);

}

else if(i>40 && i<60) {

voter.addObserver(ap);

voter.addObserver(ut);

}

else if(i>=60) {

voter.addObserver(ut);

}

voter.getTeams(teams);

voter.setVote(i);

}

ap.countVote();

ut.countVote();

}

}

**Voter Subject Interface**

public interface VoterSubject {

public void addObserver(PollObserver o);

public void notifyObserver(int i);

}

**Voter Concrete Class**

import java.util.ArrayList;

import java.util.HashMap;

public class Voters implements VoterSubject {

public ArrayList<PollObserver> pollobservers;

public int vote;

//private String team;

public HashMap<String, Integer> onevote = new HashMap<>();

public Voters() {

pollobservers = new ArrayList<PollObserver>();

}

@Override

public void addObserver(PollObserver o) {

pollobservers.add(o);

}

@Override

public void notifyObserver(int i) {

for(PollObserver o : pollobservers) {

o.update(onevote, i);

}

}

public void getTeams(HashMap<String, Integer> onevote) {

this.onevote = onevote;

}

public void setVote(int voterCount) {

int min = 1;

int max = 25;

int range = max - min + 1;

for(String k : onevote.keySet()) {

vote = (int) (Math.random() \* range) + min;

onevote.put(k, vote);

}

notifyObserver(voterCount);

}

}

**Poll Observer Interface**

import java.util.ArrayList;

import java.util.HashMap;

public interface PollObserver {

public ArrayList<HashMap<String, Integer>> allvoters= new ArrayList<HashMap<String, Integer>>();

public void update(HashMap<String , Integer> onevote, int i);

}

**Poll Subject Interface**

import java.util.HashMap;

public interface PollSubject {

public void addMediaObserver(MediaObserver o);

public void notifyMediaObserver();

}

**Associated Press Concrete Poll Class**

import java.util.ArrayList;

import java.util.HashMap;

public class AssociatedPress implements PollObserver, PollSubject {

public HashMap<String, Integer> finalVote = new HashMap<>();

public ArrayList<MediaObserver> mediaobservers;

public AssociatedPress() {

mediaobservers = new ArrayList<MediaObserver>();

}

@Override

public void update(HashMap<String , Integer> onevote, int i) {

allvoters.add(i, onevote);

}

public void countVote() {

finalVote = allvoters.get(0);

for(int i=1; i<allvoters.size(); i++) {

HashMap<String, Integer> temp = allvoters.get(i);

for(String k : finalVote.keySet()) {

int x = finalVote.get(k) + temp.get(k);

finalVote.put(k, x);

}

}

int sum = 26;

for(String k : finalVote.keySet()) {

finalVote.put(k, sum - (int)(Math.random() \* ((25 - 1) + 1)) + 1);

}

notifyMediaObserver();

}

@Override

public void addMediaObserver(MediaObserver o) {

mediaobservers.add(o);

}

@Override

public void notifyMediaObserver() {

for(MediaObserver o : mediaobservers) {

String pollname = "Associated Press";

o.updateFinal(finalVote, pollname);

}

}

}

**USAToday Concrete Poll Class**

import java.util.ArrayList;

import java.util.HashMap;

public class USAToday implements PollObserver, PollSubject {

public HashMap<String, Integer> finalVote = new HashMap<>();

public ArrayList<MediaObserver> mediaobservers;

public USAToday() {

mediaobservers = new ArrayList<MediaObserver>();

}

@Override

public void update(HashMap<String , Integer> onevote, int i) {

allvoters.add(i, onevote);

}

public void countVote() {

finalVote = allvoters.get(0);

for(int i=1; i<allvoters.size(); i++) {

HashMap<String, Integer> temp = allvoters.get(i);

for(String k : finalVote.keySet()) {

int x = finalVote.get(k) + temp.get(k);

finalVote.put(k, x);

}

}

int sum = 26;

for(String k : finalVote.keySet()) {

finalVote.put(k, sum - (int)(Math.random() \* ((25 - 1) + 1)) + 1);

}

notifyMediaObserver();

}

@Override

public void addMediaObserver(MediaObserver o) {

mediaobservers.add(o);

}

@Override

public void notifyMediaObserver() {

for(MediaObserver o : mediaobservers) {

String pollname = "USAToday";

o.updateFinal(finalVote, pollname);

}

}

}

**Subscriber Observer Interface**

import java.util.HashMap;

public interface MediaObserver {

public void updateFinal(HashMap<String, Integer> finalVote, String pollname);

}

**SUBSCRIBERS Concrete Classes -**

**ESPN**

import java.util.HashMap;

public class Espn implements MediaObserver {

public Espn(PollSubject poll) {

poll.addMediaObserver(this);

}

@Override

public void updateFinal(HashMap<String, Integer> finalVote, String pollname) {

System.out.println("-----------ESPN-------------\n "+pollname);

for(String keys : finalVote.keySet()) {

System.out.println(keys + ":" + finalVote.get(keys));

}

}

}

**LATimes**

import java.util.HashMap;

public class LATimes implements MediaObserver {

public LATimes(PollSubject poll) {

poll.addMediaObserver(this);

}

@Override

public void updateFinal(HashMap<String, Integer> finalVote, String pollname) {

System.out.println("-----------LA Times-------------\n "+pollname);

for(String keys : finalVote.keySet()) {

System.out.println(keys + ":" + finalVote.get(keys));

}

}

}

**SportsWeekly**

import java.util.HashMap;

public class SportsWeekly implements MediaObserver {

public SportsWeekly(PollSubject poll) {

poll.addMediaObserver(this);

}

@Override

public void updateFinal(HashMap<String, Integer> finalVote, String pollname) {

System.out.println("-----------Sports Weekly-------------\n "+pollname);

for(String keys : finalVote.keySet()) {

System.out.println(keys + ":" + finalVote.get(keys));

}

}

}

**WashingtonPost**

import java.util.HashMap;

public class WashingtonPost implements MediaObserver {

public WashingtonPost(PollSubject poll) {

poll.addMediaObserver(this);

}

@Override

public void updateFinal(HashMap<String, Integer> finalVote, String pollname) {

System.out.println("-----------Washington Post-------------\n "+pollname);

for(String keys : finalVote.keySet()) {

System.out.println(keys + ":" + finalVote.get(keys));

}

}

}