Name:- N. Harikishna

Index No:- 210206B

1) Objects – States
    - Swimming pool
        - States - Has 5 swimming lanes, Dimension
        - Behaviors - Touch pads are installed in end ,swimmers can swim in it
    - Pavillion
        - States - Number of seats
        - Behaviors - Spectators can watch the competition from here
    - Swimmers
        - States - Name, Gender ,Swimwear color according to gender
        - Behaviors – Touch the touchpad, Swim, Waiting for the results
    - Spectators
        - States – Name
        - Behaviors – watch the competition , Go after match
    - Judges
        - States – Name
        - Behaviors – Blow the whistle, Watching swimmers, Checking rules
    - Supporting staff
        - States –Name , Work
        - Behaviors – Helping Swimmers, judges and spectators with any issue
    - Touch pad
        - States – Fixed Location ,Time of touch
        - Behaviors – Receive touches ,Take the time
    - Score board
        - States – Finishing time , Order of finishing
        - Behaviors – Compare time, Display the order

2) Objects:

- Newspaper
- News Articles
- Text
- Images

Abstract object: Paragraph - a collection of one or more lines of text.

Characteristics of objects:

- Newspaper

  Internal state: name, date of publication

  Behaviors: having articles, publish articles, date of publication

- News Articles

  Internal state: headline, byline, tagline, date-stamp, text, images

  Behaviors: get headline, get byline, get tagline, get date-stamp, get text, get images

- Text

  Internal state: font type, font size, font color, background color, bold typeface, italic typeface, underlined typeface

  Behaviors: get font type, get font size, get font color, get background color, get bold typeface, get italic typeface, get underlined typeface

- Images

  Internal state: file name, format, resolution, size

  Behaviors: get file name, get format, get resolution, get size

3)

```
import java.util.ArrayList;

// creating swimmer class
class Swimmer {
    private String name;
    private int idNumber;

    public Swimmer(String name, int idNumber) {
        this.name = name;
        this.idNumber = idNumber;
    }

    public void swim() {
        System.out.println(this.name + " is swimming.");
    }
}

// creating spectator class
class Spectator {
    private String name;
    private int idNumber;

    public Spectator(String name, int idNumber) {
        this.name = name;
        this.idNumber = idNumber;
    }

    public void checkScoreboard(Scoreboard scoreboard) {
        System.out.println(this.name + " is checking the scoreboard. Current
score: " + scoreboard.getCurrentScore());
    }
}

// creating judge class
class Judge {
    private String name;
```

```
    private int idNumber;

    public Judge(String name, int idNumber) {
        this.name = name;
        this.idNumber = idNumber;
    }

    public void blowWhistle() {
        System.out.println(this.name + " is blowing the whistle.");
    }
}

// creating supportive staff class
class SupportingStaff {
    private String name;
    private int idNumber;

    public SupportingStaff(String name, int idNumber) {
        this.name = name;
        this.idNumber = idNumber;
    }

    public void checkScoreboard(Scoreboard scoreboard) {
        System.out.println(this.name + " is checking the scoreboard. Current
score: " + scoreboard.getCurrentScore());
    }
}

//creating scoreboard to determine the winner
class Scoreboard {
    private int currentScore;

    public Scoreboard() {
        this.currentScore = 0;
    }

    public int getCurrentScore() {
        return this.currentScore;
    }

    public void updateScore(int score) {
        this.currentScore += score;
    }
}
```

```java
//start competition
public class SwimmingCompetition {
    public static void main(String[] args) {
        int numSwimmers = Integer.parseInt(args[0]);
        int numSpectators = Integer.parseInt(args[1]);
        int numJudges = Integer.parseInt(args[2]);
        int numStaff = Integer.parseInt(args[3]);

        ArrayList<Swimmer> swimmers = new ArrayList<>();
        ArrayList<Spectator> spectators = new ArrayList<>();
        ArrayList<Judge> judges = new ArrayList<>();
        ArrayList<SupportingStaff> staff = new ArrayList<>();

        //add new swimmers
        for (int i = 0; i < numSwimmers; i++) {
            Swimmer swimmer = new Swimmer("Swimmer " + i, i);
            swimmers.add(swimmer);
        }

        //add new spectators
        for (int i = 0; i < numSpectators; i++) {
            Spectator spectator = new Spectator("Spectator " + i, i);
            spectators.add(spectator);
        }

        //add new judge
        for (int i = 0; i < numJudges; i++) {
            Judge judge = new Judge("Judge " + i, i);
            judges.add(judge);
        }

        //add new staff
        for (int i = 0; i < numStaff; i++) {
            SupportingStaff staffMember = new SupportingStaff("Staff " + i, i);
            staff.add(staffMember);
        }

        //creating scoreboard
        Scoreboard scoreboard = new Scoreboard();

        // check the scoreboard
        for (Swimmer swimmer : swimmers) {
            swimmer.swim();
        }
```

```
        for (Spectator spectator : spectators) {
            spectator.checkScoreboard(scoreboard);
        }

        for (SupportingStaff staffMember : staff) {
            staffMember.checkScoreboard(scoreboard);
        }

        //blow whistle
        for (Judge judge : judges) {
        Judge headJudge = judges.get(0);
        headJudge.blowWhistle();
        }

        // scoreboard update
        scoreboard.updateScore(10);

        //check the scoreboard again.
        for (Swimmer swimmer : swimmers) {
            swimmer.swim();
        }

        for (Spectator spectator : spectators) {
            spectator.checkScoreboard(scoreboard);
        }

        for (SupportingStaff staffMember : staff) {
            staffMember.checkScoreboard(scoreboard);
        }
    }
}
```