# CA Exam Gradesheet MAR-04

## Table of Contents



For discussion of CA Exam in lectures/labs.

## Q1a (10 marks)

Write an interface called `Weather` that has methods for the following:

- +2 marks for interface definition
- `getName` to get the name of the weather, e.g. "Rainy", "Sunny"
- `getDuration` to get the minutes the weather was in effect during the day, e.g. "60", "600"
- `wasItRainy` to get whether it rained when the weather was in effect - this will be either true/yes or false/no
- `wasItSunny` to get whether it was sunny when the weather was in effect - this will be either true/yes or false/no
- `wasItWindy` to get whether it was windy when the weather was in effect - this will be either true/yes or false/no
- `wasItSnowy` to get whether it snowed when the weather was in effect - this will be either true/yes or false/no

- Write a class called `TypicalDay` which implements `Weather` (+2 marks)
- with the default values for name as "Typical Day", duration as 1440 minutes, (+1 marks)
- and where it rains, is sunny, windy, and snowy within the same day. (+1 marks)
- For the name and duration, write appropriate getters and setters such that the name cannot be less than 5 letters (+2 marks)
- and the duration cannot be less than 60mins. (+2 marks)

JAVA

```java
// Sample Solution
interface Weather {
    // to get the name of the weather, e.g. "Rainy", "Sunny"
    String getName();
    // to get the minutes the weather was in effect during the day,
    // e.g. "60", "600"
    int getDuration();
    // to get whether it rained when the weather was in effect -
    // this will be either true/yes or false/no
    boolean wasItRainy();
    // to get whether it was sunny when the weather was in effect -
    // this will be either true/yes or false/no
    boolean wasItSunny();
    // to get whether it was windy when the weather was in effect -
    // this will be either true/yes or false/no
    boolean wasItWindy();
    // to get whether it snowed when the weather was in effect -
    // this will be either true/yes or false/no
    boolean wasItSnowy();
}

class TypicalDay implements Weather {
    String name = null;
    int duration = 0;
    boolean isSunny = false;
    boolean isRainy = false;
    boolean isWindy = false;
    boolean isSnowy = false;

    public TypicalDay() {
        // default values for name as "Typical Day", duration as 1440 minutes,
        // and where it rains, is sunny, windy, and snowy within the same day
        this.name = "Typical Day";
        this.duration = 1440;
        this.isSunny = true;
        this.isRainy = true;
        this.isWindy = true;
        this.isSnowy = true;
    }

    public String getName() { return this.name; }
    public void setName(String name) {
        // name cannot be less than 5 letters
        if (name.length() < 5) { return; }
        this.name = name;
    }

    public int getDuration() { return this.duration; }
    public void setDuration(int duration) {
        // duration cannot be less than 60mins
        if (duration < 60) { return; }
        this.duration = duration;
    }

    public boolean wasItRainy() { return this.isRainy; }
    public boolean wasItSunny() { return this.isSunny; }
    public boolean wasItWindy() { return this.isWindy; }
    public boolean wasItSnowy() { return this.isSnowy; }
}
```

# Q1b (10 marks)

Write a class called `WeatherStation` which keeps records of the weather in terms of how many hours of sunshine, rain, winds, and snow have occurred. The `WeatherStation` contains the following features:

- a list of `Weather` instances (+1 mark for list definition, +1 mark for initialising)
- a map or dictionary which has string keys as weather type (e.g. 'rainy') and integer values as the count of duration for that weather in hours e.g. "Sunny: 2" would represent sunny weather for a total of 2 hours (+1 mark for definition, +1 mark for initialising)
- +1 marks for initialising the data structures i.e. set counters to 0
- method `addWeather` which takes in an instance of `Weather` and adds it to the list, and increments appropriate counters for sunshine, rain, etc. (+5 marks - definition +1, add to list +1, iterate +1, increment counter +2)

JAVA

```java
/**
 * keeps records of the weather in terms of how many hours of sunshine, rain, winds, and snow have occurred
 */
class WeatherStation {
    List<Weather> record = null;
    // map or dictionary which has keys as weather type (e.g. 'rainy')
    // and values as the count of duration for that weather in hours
    // e.g. "Sunny: 2" would represent sunny weather for a total of 2 hours
    Map<String,Integer> weatherCounter = null;
    final String RAINY = "Rainy";
    final String SUNNY = "Sunny";
    final String WINDY = "Windy";
    final String SNOWY = "Snowy";

    public WeatherStation() {
        this.record = new ArrayList<Weather>();
        this.weatherCounter = new HashMap<String,Integer>();
        this.weatherCounter.put(RAINY, 0);
        this.weatherCounter.put(SUNNY, 0);
        this.weatherCounter.put(WINDY, 0);
        this.weatherCounter.put(SNOWY, 0);
    }

    public void addWeather(Weather record) {
        if (record == null) { return; }
        this.record.add(record);

        int duration = record.getDuration() / 60; // convert duration to hours

        if (record.wasItRainy()) {
            this.weatherCounter.compute(RAINY, (k, v) -> v + duration);
            // this.weatherCounter.put(RAINY, this.weatherCounter.get(RAINY) + duration);
        }
        if (record.wasItSunny()) {
            this.weatherCounter.compute(SUNNY, (k, v) -> v + duration);
            // this.weatherCounter.put(RAINY, this.weatherCounter.get(RAINY) + duration);
        }
        if (record.wasItWindy()) {
            this.weatherCounter.compute(WINDY, (k, v) -> v + duration);
            // this.weatherCounter.put(RAINY, this.weatherCounter.get(RAINY) + duration);
        }
        if (record.wasItSnowy()) {
            this.weatherCounter.compute(SNOWY, (k, v) -> v + duration);
            // this.weatherCounter.put(RAINY, this.weatherCounter.get(RAINY) + duration);
        }
    }
}
```

# Q1c (5 marks)

- the singleton pattern definition in a classed called `SingleWeather`
- using generics in definition +1 mark
- that only classes implementing `Weather` can be used with it +1 mark
- singleton instance saved as private +1 mark
- Write methods to retrieve and set the singleton - +2 marks

JAVA

```java
class SingleWeather<T extends Weather> {
    private T instance;
    public T getInstance() {
        return this.instance;
    }
    public void setInstance(T instance) {
        this.instance = instance;
    }
}
```

Last updated 2024-03-07 13:31:19 UTC