

# Objektorientierte Programmierung (Java)

## Beispiel WeatherApp

Erstellen Sie eine Java Applikation „WeatherApp“, die folgende Anforderungen erfüllt:

Eine Wetterstation (WeatherStation) hat mehrere Messungen für einen Standort. Für jede Messung sollen Informationen über Datum, Temperatur und Luftfeuchtigkeit gespeichert werden.

- **Klasse Measurement**
  - Time: Date (Zeitpunkt der Messung)
  - Temperature: float (Temperatur in ° Celsius)
  - Humidity: float (Luftfeuchtigkeit)
- **Klasse WeatherStation**
  - Location: String (Standort an dem die Messungen vorgenommen werden)
  - Measurements: Liste mit allen Messungen

Legen Sie beide Klassen mitsamt ihrer Attribute, Getter- und Setter Methoden, geeigneten Konstruktor(en) und einer toString()-Methode an.

Zusätzlich dazu sollen noch weitere Methoden in der Klasse **WeatherStation** implementiert werden, um die Verwaltung der Messungen zu vereinfachen bzw. einfache Auswertungen machen zu können:

- **addMeasurement**  
legt ein neues Messungs- Element (Measurement) an und fügt es zu den bestehenden Messungen hinzu.

Bevor ein Element angelegt wird, sollte darauf geachtet werden, dass Temperatur und Luftfeuchtigkeit in folgenden Bereichen liegen sollten:

Temperatur: zwischen -80 und 80 °Celsius

Luftfeuchtigkeit: zwischen 0 und 100

Liegt einer der beiden Werte nicht in diesem Bereich, soll (direkt in dieser Methode) eine Fehlermeldung ausgegeben werden und die Messung verworfen werden (nicht zu den Messungen hinzufügen).

Übergabeparameter: Time (als String im Format „2021-07-22 14:35:22“ -> muss in Datum umgewandelt werden), Temperature, Humidity

Rückgabewert: keine

### Hinweis:

Das Umwandeln eines Strings in ein Date Object kann mit Hilfe eines SimpleDateFormat gemacht werden:

```
Date measurementTime = null;  
try {
```

```

        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd
HH:mm");
        measurementTime = formatter.parse(time);
    } catch (ParseException ex) {

    }
}

```

- **maxTemperatureMeasurement**  
durchsucht alle Messungen und gibt die Messung mit der höchsten Temperatur zurück  
Übergabeparameter: keine  
Rückgabewert: Measurement (Messung mit der höchsten Temperatur)
- **minTemperatureMeasurement**  
durchsucht alle Messungen und gibt die Messung mit der niedrigsten Temperatur zurück  
Übergabeparameter: keine  
Rückgabewert: Measurement (Messung mit der niedrigsten Temperatur)
- **avgTemperature**  
berechnet die Durchschnittstemperatur aller Messungen  
Übergabeparameter: keine  
Rückgabewert: durchschnittliche Temperatur als float
- **countMeasurements**  
gibt die Anzahl der Measurements in der WeatherStation zurück  
Übergabeparameter: keine  
Rückgabewert: int

## Hauptprogramm

```

public class WeatherApp {

    /**
     * @param args the command line arguments
     */

    public static void main(String[] args) {

        // WeatherStation anlegen

        WeatherStation station = new WeatherStation("WIFI Linz");

        // Messungen hinzufügen

        station.addMeasurement("2021-07-22 08:00:00", 14.0f, 30);
        station.addMeasurement("2021-07-22 09:00:00", 15.0f, 30);
        station.addMeasurement("2021-07-22 10:00:00", 16.0f, 32);
        station.addMeasurement("2021-07-22 11:00:00", 17.0f, 32);
        station.addMeasurement("2021-07-22 12:00:00", 19.0f, 34);
    }
}

```

```

station.addMeasurement("2021-07-22 13:00:00", 22.0f, 34);
station.addMeasurement("2021-07-22 14:00:00", 24.0f, 34);
station.addMeasurement("2021-07-22 15:00:00", 25.0f, 34);
station.addMeasurement("2021-07-22 16:00:00", 24.5f, 34);
station.addMeasurement("2021-07-22 17:00:00", 23.0f, 34);
station.addMeasurement("2021-07-22 18:00:00", 22.5f, 38);
station.addMeasurement("2021-07-22 19:00:00", 21.0f, 38);
station.addMeasurement("2021-07-22 20:00:00", 20.5f, 38);

// Messungen mit Fehlern hinzufügen
station.addMeasurement("2021-07-22 21:00:00", 20.5f, 380);
station.addMeasurement("2021-07-22 22:00:00", 120.5f, 380);

// Ausgabe der WeatherStation
System.out.println(station);

// Messung mit niedrigster Temperatur ermitteln und ausgeben
System.out.println("\nMessung mit niedrigster Temperatur:");
Measurement min = station.minTemperatureMeasurement();
System.out.println(min);

// Messung mit höchster Temperatur ermitteln und ausgeben
System.out.println("\nMessung mit höchster Temperatur:");
Measurement max = station.maxTemperatureMeasurement();
System.out.println(max);

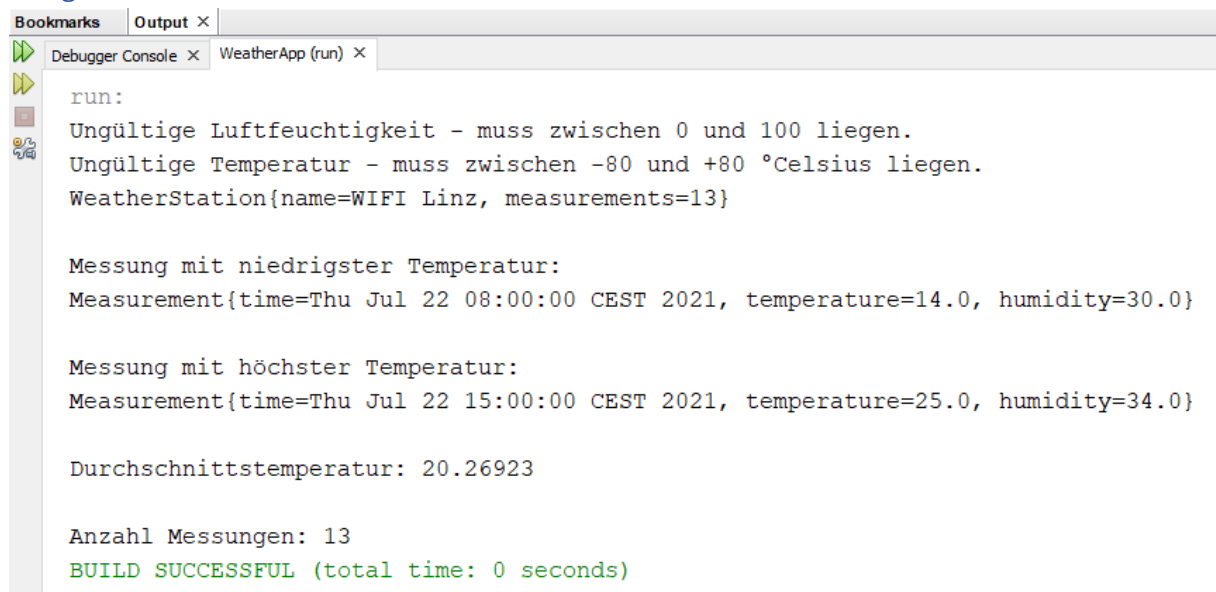
// Durchschnittstemperatur ermitteln und ausgeben
float avgTemp = station.avgTemperature();
System.out.println("\nDurchschnittstemperatur: " + avgTemp);

// Anzahl an Messungen in der WeatherStation ermitteln und ausgeben
int countMeasurements = station.countMeasurements();
System.out.println("\nAnzahl Messungen: " + countMeasurements);
}

}

```

## Ausgabe



```
run:
Ungültige Luftfeuchtigkeit - muss zwischen 0 und 100 liegen.
Ungültige Temperatur - muss zwischen -80 und +80 °Celsius liegen.
WeatherStation{name=WIFI Linz, measurements=13}

Messung mit niedrigster Temperatur:
Measurement{time=Thu Jul 22 08:00:00 CEST 2021, temperature=14.0, humidity=30.0}

Messung mit höchster Temperatur:
Measurement{time=Thu Jul 22 15:00:00 CEST 2021, temperature=25.0, humidity=34.0}

Durchschnittstemperatur: 20.26923

Anzahl Messungen: 13
BUILD SUCCESSFUL (total time: 0 seconds)
```