Test Project

Session 7 - AirWatch

IT Software Solutions for Business

Submitted by:
Faye FEI - Neusoft Education and Technology Group, Independent Test Project Designer
Benjamin Frost, Independent Test Project Validator

Contents

Introduction **3**
Description of project and tasks **3**
The system under test 3
SubIndex *3*
Category *3*
OverallAqi *4*
DominantPollutant *4*
NowCast *4*
Instructions to the Competitor **4**
Other **4**
Appendix — Domain model **5**
Appendix — AQI breakpoint tables **6**
Appendix — AQI categories **7**

© WorldSkills International

Introduction

AirWatch, a non-profit, runs a city's low-cost air-quality sensor network and publishes an **Air Quality Index (AQI)** so residents know when the air is unhealthy. The AQI library is already written and in production. Before the next rollout, the city requires an **independent, high-quality automated test suite** that proves the library matches its specification and will catch future regressions. Writing that suite is your job.

Description of project and tasks

Write a thorough xUnit test suite for the five methods of **IAirQualityIndex**, in **AirWatch.Aqi.UnitTests** — the only project you submit. You are testing a **provided, compiled reference implementation**; you do **not** change it. Resolve it with `ServiceResolver.Resolve<IAirQualityIndex>()` (the sample tests show the pattern). Your suite is graded on **how much of the library it exercises** and **how many injected faults it catches** — so test the boundaries, not just the happy path. No air-quality knowledge is required: every rule is in the specification below.

The system under test

The library is **pure, deterministic, and never throws**: each method returns a **documented sentinel** on bad input rather than throwing. Concentrations are doubles (PM in µg/m³, ozone in ppb); sub-indices and the overall AQI are **whole numbers**; rounding is **half away from zero**. The `Pollutant` enum and the `Reading` record are defined in the appendix.

SubIndex

```
int SubIndex(Pollutant pollutant, double concentration);
```

Convert a concentration to its integer AQI sub-index using the pollutant's **breakpoint table** (given in the appendix). First **truncate** the concentration to the pollutant's reporting precision — 1 decimal place for PM2.5, whole units for PM10 and O_3. Then find the band whose range contains the truncated concentration (a value on a **shared boundary belongs to the lower band**) and interpolate linearly between its bounds, where:

- C **— the input concentration (after truncation).**
- C_{lo}**,** C_{hi} **— the band's lower and upper concentration bounds.**
- I_{lo}**,** I_{hi} **— the band's lower and upper sub-index bounds.**
- I **— the resulting AQI sub-index.**
- $I = round\left(\frac{I_{hi}-I_{lo}}{C_{hi}-C_{lo}} \cdot (C - C_{lo}) + I_{lo}\right)$

Sentinel: return `-1` when the concentration is negative or above the table's top breakpoint.

Example. PM2.5 is truncated to 1 dp first. `9.04` µg/m³ → `9.0`, in band 0.0–12.0, so `I = round(50 / 12 · 9.0) = round(37.5) = 38`. `12.34` → `12.3`, in band 12.0–35.4, so `I = round(51 + (100 – 51) / (35.4 – 12.0) · (12.3 – 12.0)) = round(51.63) = 52`. Exactly `12.0` sits on the shared boundary and takes the lower band, so `I = 50`.

Category

```
string Category(int aqi);
```

Map an AQI value to its category label (the exact string the method returns); the bands are listed in the appendix.

© WorldSkills International

OverallAqi

```
int OverallAqi(IReadOnlyList<Reading> readings);
```

The overall AQI is the **greatest** valid sub-index across the readings; invalid readings (sub-index `-1`) are ignored. **Sentinel:** return `-1` when `readings` is null or empty, or when no reading is valid.

DominantPollutant

```
Pollutant? DominantPollutant(IReadOnlyList<Reading> readings);
```

The pollutant with the **highest** valid sub-index — the one that sets the overall AQI. Ties are broken by **enum order** (`Pm25` before `Pm10` before `O3`). **Sentinel:** return `null` when no reading is valid.

NowCast

```
int NowCast(Pollutant pollutant, IReadOnlyList<double?> recentHours);
```

Compute the **real-time** sub-index from up to the last 12 hourly concentrations, weighting more recent hours more heavily. `recentHours[0]` is the most recent hour; `null` marks a missing reading; entries beyond index 11 are ignored. The NowCast concentration is the recency-weighted mean of the present hours, where:

- c_i — the concentration at hour i (only present hours are included).
- i — the hour index, with $i = 0$ the most recent; a reading is weighted by w raised to its own age.
- c_{min}, c_{max} — the smallest and largest present values among those hours.
- w — the weight, $w = clamp(c_{min}/c_{max},\ 0.5,\ 1)$, taken as 1 when $c_{max} = 0$.
- $nowcast$ — the resulting concentration, fed to `SubIndex`.

$$nowcast = \frac{\sum_i \quad w^i \cdot c_i}{\sum_i \quad w^i}$$

- **Data sufficiency.** If fewer than two of the three most recent hours (`recentHours[0]`, `[1]`, `[2]`) have a value, return `-1`.
- **Result.** Return `SubIndex(pollutant, nowcast)` — the truncation rule above applies to the NowCast concentration.

Sentinel: return `-1` when the data-sufficiency rule fails, or when `recentHours` is null or empty. Never throws.

Example. For PM2.5 with `recentHours` = [20.0, 10.0, 30.0] (most recent first), the range gives `w` = clamp(10 / 30, 0.5, 1) = 0.5, so `nowcast` = (20 + 0.5 · 10 + 0.25 · 30) / (1 + 0.5 + 0.25) = 18.571…, truncated to 18.5, giving `SubIndex(Pm25, 18.5)` = 65.

Instructions to the Competitor

This section contains the instructions given to the Competitor for this session. The instructions can be increased and improved at the Competition during the preparation time.

Other

Use this section for information that is important but is not covered in the previous sections.

© WorldSkills International

Appendix — Domain model

The system under test exposes two types from `AirWatch.Aqi.Contracts`.

`Pollutant` — an enum naming the pollutants the index covers.

VALUE	POLLUTANT	MEANING
`Pm25`	PM2.5	Fine particulate matter up to 2.5 µm across (e.g. soot, smoke); measured in µg/m³.
`Pm10`	PM10	Coarse particulate matter up to 10 µm across (e.g. dust, pollen); measured in µg/m³.
`O3`	Ozone (O_3)	Ground-level ozone; measured in parts per billion (ppb) as an 8-hour average.

`Reading` — one pollutant measurement taken at a sensor.

FIELD	TYPE	DESCRIPTION
`Pollutant`	`Pollutant`	Which pollutant this reading is for.
`Concentration`	`double`	The measured concentration — µg/m³ for PM2.5 and PM10, ppb for O_3.

© WorldSkills International

Appendix — AQI breakpoint tables

Each pollutant's concentration bands and their sub-index ranges, used by `SubIndex`. A concentration on a shared boundary belongs to the lower band.

POLLUTANT	CONCENTRATION RANGE (C_{LO}–C_{HI})	SUB-INDEX (I_{LO}–I_{HI})
PM2.5 (µg/m³)	0.0 – 12.0	0 – 50
	12.0 – 35.4	51 – 100
	35.4 – 55.4	101 – 150
	55.4 – 150.4	151 – 200
	150.4 – 250.4	201 – 300
	250.4 – 500.4	301 – 500
PM10 (µg/m³)	0 – 54	0 – 50
	54 – 154	51 – 100
	154 – 254	101 – 150
	254 – 354	151 – 200
	354 – 424	201 – 300
	424 – 604	301 – 500
O_3 (ppb, 8-h)	0 – 54	0 – 50
	54 – 70	51 – 100
	70 – 85	101 – 150
	85 – 105	151 – 200
	105 – 200	201 – 300

© WorldSkills International

Appendix — AQI categories

The AQI ranges and the category string that `Category` returns for each.

AQI	RETURNED STRING
0 – 50	`"Good"`
51 – 100	`"Moderate"`
101 – 150	`"USG"` (Unhealthy for Sensitive Groups)
151 – 200	`"Unhealthy"`
201 – 300	`"Very Unhealthy"`
301 and above	`"Hazardous"`
below 0	`"unknown"`

© WorldSkills International