

# Laboratorium 9 – Adnotacja typami, testowanie

Języki skryptowe

## Cele dydaktyczne

1. Zapoznanie się z adnotacją typami w języku Python.
2. Ćwiczenie statycznej weryfikacji typów z użyciem narzędzia mypy.
3. Praktyczne wykorzystanie biblioteki pytest do testowania klas i metod związanych z przetwarzaniem danych pomiarowych.

## Zadania

1. Dokonaj refaktoryzacji klas utworzonych w ramach Laboratorium 6: Station, TimeSeries, SeriesValidator oraz klas dziedziczących, Measurements.  
Refaktoryzacja powinna obejmować **adnotację typami**:
  - a. wszystkich parametrów metod (poza self i cls)
  - b. wartości zwracanych przez metody,
  - c. typów zmiennych wewnętrznych (jeśli są jawnie deklarowane).
2. Dokonaj statycznej weryfikacji typów z wykorzystaniem narzędzia [mypy](#). Dokonaj poprawek w kodzie lub adnotacjach w taki sposób, by mypy nie raportował informacji o błędach.
- 3.
4. Z wykorzystaniem biblioteki [pytest](#), skonstruuj następujące testy jednostkowe:
  - a. test weryfikujący poprawność działania metody `__eq__` klasy Station dla obiektów o tym samym i różnych kodach stacji,
  - b. testy do weryfikacji poprawnego działania metody `__getitem__` klasy TimeSeries przy podaniu:
    - i. indeksu całkowitego,
    - ii. obiektu slice,
    - iii. obiektu datetime.date odpowiadającego istniejącej dacie pomiaru,
    - iv. obiektu datetime.date nieistniejącego w danych (powinien rzucać

KeyError),

- c. test poprawności obliczenia średniej (mean) i odchylenia standardowego (stddev) w obiekcie TimeSeries – przetestuj przypadek:
  - i. z kompletnym zestawem danych (bez None),
  - ii. z danymi zawierającymi wartości None (powinny być ignorowane przy obliczeniach),
- d. test walidatora OutlierDetector dla danych zawierających wartość odstającą o więcej niż k odchyłeń standardowych,
- e. test walidatora ZeroSpikeDetector dla danych z trzema kolejnymi wartościami 0 lub None,
- f. test walidatora ThresholdDetector dla wartości przekraczających zdefiniowany próg,
- g. test działania detect\_all\_anomalies():
  - i. z wykorzystaniem dekoratora [@pytest.mark.parametrize](#),
  - ii. z różnymi obiektami walidatorów (np. OutlierDetector, ZeroSpikeDetector, SimpleReporter),
  - iii. weryfikuj, że każdy z obiektów typu „analyzer” zwraca listę komunikatów dla przykładowego TimeSeries,
  - iv. nie stosuj sprawdzania typu (isinstance, issubclass) – testuj zgodność przez zachowanie, zgodnie z zasadą polimorfizmu strukturalnego (kaczego typowania).