Zadanie Python: Magazyn danych – JSON, XML i dane syntetyczne

Cel

Zaimplementuj mini-bibliotekę w Pythonie, która:
1) generuje dane syntetyczne,
2) zapisuje je i odczytuje w formatach JSON oraz XML,
3) zapewnia prosty interfejs obiektowy z czytelnymi klasami i testami.

Wymagania funkcjonalne

- Dane domenowe: rekord osoby/klienta (ID, imię, nazwisko, e-mail, wiek, data_rejestracji, tagi).
- Generator danych syntetycznych:
  • możliwość wygenerowania n rekordów z kontrolą zakresu wieku (np. 18–80),
  • deterministyczność przy ustawieniu seed.
- Zapis/Odczyt:
  • JSON: zapis listy rekordów do pliku, odczyt do listy obiektów.
  • XML: zapis listy rekordów do pliku (węzeł root <people>; element <person> z atrybutem id), odczyt do listy obiektów.
  • Obsługa katalogów (tworzenie, jeśli nie istnieją).
  • Bez duplikatów (ten sam id nie może pojawić się 2× przy dopisywaniu).
- Interfejs CLI z argumentami: --format, --out PATH, --n N, --seed SEED, --min-age, --max-age.

Wymagania techniczne

- Użyj klas i dataclasses.
- Stwórz interfejs DataStore (ABC) z metodami save, load, append_unique.
- Implementacje: JsonStore, XmlStore.
- Biblioteki standardowe: json, xml.etree.ElementTree, xml.dom.minidom, pathlib.Path, argparse, random, datetime.
- Testy: min. 5 asercji.

Struktura projektu (propozycja)

```
project/
  data/
  src/
    models.py
    generator.py
    store_base.py
    store_json.py
    store_xml.py
    cli.py
```

tests/
  test_basic.py

Specyfikacja klas

Person: dataclass z polami (id, first_name, last_name, email, age, registered_at, tags).
DataStore (ABC): save, load, append_unique.
JsonStore/XmlStore: implementacje metod.
SyntheticDataGenerator: generate(n, seed, min_age, max_age).

Interfejs CLI

Przykład użycia:
python -m src.cli --format json --out data/people.json --n 100 --seed 42
python -m src.cli --format xml --out data/people.xml --n 20 --seed 123 --min-age 21 --max-age 65

Podpowiedzi (hinty)

- Tworzenie katalogu: path.parent.mkdir(parents=True, exist_ok=True)
- JSON zapis/odczyt: json.dump, json.load
- XML pretty print: xml.etree.ElementTree + xml.dom.minidom
- Unikalność: sprawdzaj istniejące id przed dopisaniem.

Zadania rozszerzone (bonus)

1) Walidacja email.
2) Eksport do CSV.
3) Filtrowanie przy odczycie.
4) Benchmark JSON vs XML.
5) Walidacja struktury XML (np. XSD).