**Recapitulare**

1. Selecteaza afirmatiile adevarate referitoare la Design Patterns in Python:
2. Design pattern-urile reprezinta cod definit care ne ajuta sa rezolvam o problema comuna, putem sa il copiem si sa rezolvam problemele similare.
3. Design Patterns reprezinta un set de “good practices”, dupa care se ghideaza scrierea de cod in OOP.
4. Folosin un design pattern, putem sa scriem cod mai curat, mai flexibil, extensibil si decuplat.
5. Exista 3 categorii de design patterns: creational, behavioural, structural.
6. Decorator este un design pattern structural.

**Raspunsuri corecte:**

1. Selecteaza afirmatiile adevarate:
2. Un obiect este iterator daca are implementata metoda \_\_iter\_\_().
3. Un obiect este iterabil daca are implementate metodele \_\_iter\_\_() si \_\_next\_\_().
4. Un obiect este iterator daca are implementate metodele \_\_enter\_\_() si \_\_exit\_\_().
5. Un obiect este iterabil daca are implementata metoda \_\_iter\_\_().
6. Un obiect este iterator daca are implementata metoda \_\_iter\_\_() si \_\_next\_\_().

**Raspunsuri corecte:**

1. Selecteaza afirmatiile adevarate:
2. Un generator este un nou tip de date din Python.
3. Un generator este o functie care are cel putin o instructiune yield.
4. Spre deosebire de instructiunea return, instructiunea yield incheie o functie in intregime.
5. Spre deosebire de instructiunea return, instructiunea yield opreste functia salvand toate starile sale si mai tarziu continua de acolo la apeluri succesive.
6. Generatorii sunt tot iteratori, si metodele \_\_iter\_\_() si \_\_next\_\_() sunt create in mod automat.

**Raspunsuri corecte:**

1. Selecteaza afirmatiile adevarate:
2. Un context manager este o functie noua in Python.
3. Un context manager este un obiect care implementeaza metodele \_\_iter\_\_() si \_\_next\_\_().
4. Un context manager este un obiect care implementeaza metodele \_\_enter\_\_() si \_\_next\_\_().
5. Un context manager este un obiect care implementeaza metodele \_\_enter\_\_() si \_\_exit\_\_().
6. Metodele \_\_enter\_\_() si \_\_exit\_\_() sunt metode apelate automat cand folosim blocul with.

**Raspunsuri corecte:**

1. Selecteaza afirmatiile adevarate:
2. Decoratorii sunt functii cu utilizare rara in Python
3. Decoratorii sunt functii foarte importante in Python si reprezinta un tool puternic pentru a extinde comportamentul altor functii.
4. Cand folosim decoratori, implementarea functiilor care sunt decorate trebuie schimbata.
5. Avantajul decoratoriilor este ca extind functionalitatea functiilor fara sa le modifice implementarea.
6. Pentru a invoca un decorator, folosim @.

**Raspunsuri corecte:**

1. Care este diferenta principala intre un iterator si un generator in Python?
2. Un iterator este o functie, in timp ce un generator este un obiect.
3. Un iterator trebuie sa implementeze metoda \_\_iter\_\_(), in timp ce un generator trebuie sa implementeze metoda \_\_next\_\_().
4. Un iterator este un obiect, in timp ce un generator este o functie.
5. Un iterator trebuie sa implementeze metodele \_\_iter\_\_() si \_\_next\_\_(), in timp ce in cazul generatorilor, aceste metode nu sunt folosite.
6. Nu exista diferente semnificative intre un iterator si un generator.

**Raspunsuri corecte:**