Óraifeladat megoldás

Időkorlát: 80 perc

1. Egyszerű osztály létrehozása

Feladat: Készíts egy `Dog` nevű osztályt. A kutyának legyen neve és fajtája. Legyen egy metódus, ami kiírja, hogy „XY, a Z fajta kutya ugat: Vau vau!”.
Példa: `"Bodri a Puli ugat: Vau vau!"`

---

2. Konstruktor gyakorlása

Feladat: Készíts egy `Book` osztályt, amiben megadod a könyv címét, szerzőjét és oldalszámát. Hozz létre 2 könyvet, és írasd ki az adataikat szépen sorban.

---

3. Öröklés – alap

Feladat: Hozz létre egy `Vehicle` (jármű) nevű alaposztályt. Ezután csinálj egy `Car` (autó) osztályt, ami ebből örököl. A `Car` osztályban legyen egy metódus, ami kiírja: „Az autó elindult.”

---

4. Metódus felülírás

Feladat: Készíts egy `Animal` (állat) osztályt egy `speak()` nevű metódussal. Ezután hozz létre két osztályt: `Dog` és `Cat`, amik felülírják a `speak()` metódust úgy, hogy „Vau” és „Miau” jelenjen meg.

---

5. Polimorfizmus – egyszerű példa

Feladat: Írj egy `make_animal_speak(animal)` függvényt, ami meghívja az adott állat `speak()` metódusát. Próbáld ki kutyával és macskával is.

---

6. Egy osztály csak egy dolgot csináljon (SRP – Single Responsibility Principle)

Feladat: Készíts egy `User` osztályt, ami csak tárolja a felhasználó adatait (pl. név, email). Ezután írj egy külön `UserPrinter` osztályt, ami szép formában ki tudja írni ezeket az adatokat. Ne keverjük az adattárolást és a kiírást.

---

7. OCP – nyitott legyen bővítésre, de zárt módosításra

Feladat: Készíts egy `Shape` nevű alaposztályt. Ezután hozz létre külön `Circle` (kör) és `Square` (négyzet) osztályokat, amik mindegyikében külön-külön van egy `area()` metódus. Ne if-else-sel döntsd el, melyik micsoda – minden osztály tegye a saját dolgát.

---

8. LSP – helyettesíthetőség

Feladat: Legyen egy `Bird` osztály. Abból származzon egy `Duck` és egy `Penguin` osztály. Figyelj arra, hogy a pingvin ne örököljön pl. repülés funkciót, mert nem tud repülni. Ne erőltess rá olyan viselkedést, amit nem tud.

---

9. ISP – csak azt kelljen megvalósítani, amire szükség van

Feladat: Legyen egy `Printer` interfész, amiben van `print()`, `scan()` és `fax()` metódus. Csinálj két osztályt: egy egyszerű nyomtatót (`SimplePrinter`), ami csak nyomtat, és egy haladó nyomtatót (`AdvancedPrinter`), ami mindent tud. Ne kelljen az egyszerű nyomtatónak implementálni a faxolást, ha nem tudja.

---

10. DIP – ne legyenek konkrét függőségek

Feladat: Írj egy `Logger` osztályt, ami naplóz. Ezután írj egy `UserService` osztályt, ami nem maga hozza létre a `Logger`-t, hanem kívülről kapja meg. Így könnyebb majd tesztelni vagy cserélni a naplózást.

---

11. Kompozíció – objektumok egymásban

Feladat: Legyen egy `Engine` (motor) osztály. Készíts egy `Car` (autó) osztályt, ami tartalmaz egy `Engine` példányt. Ha elindítod az autót (`car.start()`), az indítsa el a motort is.

---

12. Getter / Setter és validálás property-vel

Feladat: Írj egy `Person` osztályt, benne egy `age` (életkor) adattaggal. Ne lehessen negatív számot beállítani! Használj `@property` dekorátort, így lehet ellenőrizni a setter metódusban, hogy mit kap.