**9. Objektum-Orientált Programozás (OOP) – Tesztkérdések**

1. Melyik NEM az OOP alapelve?
   * **Functional Programming** ✅
   * Encapsulation
   * Inheritance
2. Mi az Encapsulation célja?
   * Az öröklődés biztosítása
   * **Az adatok és a funkcionalitás elrejtése** ✅
   * A kód sebességének növelése
3. Mit jelent az Inheritance?
   * **Egy osztály örökölheti egy másik osztály tulajdonságait és metódusait** ✅
   * Egy osztály belsejében más osztályokat hozhatunk létre
   * Egy változó típusa nem változhat meg
4. Melyik NEM a Polymorphism egy példája?
   * **Egy függvény csak egyféle paramétertípust fogad el** ✅
   * Egy metódus különböző paraméterlistákkal meghívható
   * Egy osztály felüldefiniálhat egy örökölt metódust
5. Mi az Abstraction előnye?
   * **Csökkenti a kód bonyolultságát azáltal, hogy elrejti a belső részleteket** ✅
   * Segít globális változókat létrehozni
   * Felgyorsítja a program végrehajtását
6. Mit csinál egy interfész?
   * **Csak metódusok deklarációit tartalmazza, de nem tartalmaz implementációt** ✅
   * Megvalósít egy teljes osztályt
   * A futtatás közbeni hibák elkerülésére szolgál
7. Melyik NEM egy OOP tervezési minta?
   * Singleton
   * Factory
   * **Bootstrap** ✅
8. Mit biztosít a Singleton minta?
   * **Egy osztályból csak egyetlen példány létezhet** ✅
   * Egy osztály több példányát egyszerre
   * Több interfész megvalósítását
9. Mi a Factory Pattern célja?
   * **Objektumok létrehozása anélkül, hogy a konkrét osztályt megadnánk** ✅
   * Több osztály egyesítése egy interfészen keresztül
   * Objektumok tömbökké alakítása
10. Mit csinál az Observer minta?

* **Egy esemény bekövetkezésekor értesíti a feliratkozott objektumokat** ✅
* Egy osztály egy másik osztálytól örököl
* Több interfész egyidejű használatát teszi lehetővé

1. Mit jelent az Dependency Injection (DI)?

* **Egy osztály kívülről kapja meg a szükséges függőségeket** ✅
* Egy osztály belsőleg inicializálja az összes változóját
* Az osztályokat nem lehet módosítani futási időben

1. Mi az Inversion of Control (IoC) célja?

* **A vezérlés átvétele az osztályoktól a keretrendszer által** ✅
* A függvények inline módon való deklarálása
* Egy osztályt közvetlenül inicializálunk

1. Melyik nyelv támogatja az OOP-t?

* Java
* Python
* **Mindkettő** ✅

1. Hogyan lehet elérni a polimorfizmust?

* **Metódus túlterheléssel és metódus felüldefiniálással** ✅
* Csak statikus változók létrehozásával
* Csak interfészek használatával

1. Melyik igaz az absztrakt osztályokra?

* **Tartalmazhat absztrakt és konkrét metódusokat is** ✅
* Csak absztrakt metódusokat tartalmazhat
* Nem lehet belőlük leszármaztatni más osztályokat

1. Melyik nyelvben NINCS natív OOP támogatás?

* **C** ✅
* Java
* C#

1. Melyik metódus NEM öröklődik alapértelmezés szerint Java-ban?

* toString()
* equals()
* **main()** ✅

1. Hogyan hívható meg a szülő osztály konstruktora egy alosztályban Java-ban?

* parent.constructor()
* super() ✅
* base()

1. Mi a különbség az absztrakt osztály és az interfész között?

* Az absztrakt osztály csak absztrakt metódusokat tartalmazhat
* **Az interfészben minden metódus absztrakt, az absztrakt osztály tartalmazhat implementációt** ✅
* Az interfésznek nincs szerepe az OOP-ben

1. Mi történik, ha egy osztály nem definiál egy interfész összes metódusát?

* **Hibaüzenetet kapunk, mert minden metódust meg kell valósítani** ✅
* Az interfész automatikusan implementálja a hiányzó metódusokat
* Semmi, a program továbbra is működik