**Web programozás**

**3.Laborgyakorlat**

**Feladatok:**

1. Konstruktor, destruktor műveletek tanulmányozása, példányosítás, adattagok hozzáférése (User osztály, *Adattagok: vezNev, kerNev, szulEv, minKor - statikus*). Náhány user-t tegyünk be egy tömbbe, majd listázzuk ki egy bizonyos évszám előtt születetteket (használjunk closurt, array\_filter-t)
2. Készítsünk egy egyszerű *Calculator* osztályt, amely képes a 4 alapművelet elvégzésére, majd teszteljük ezeket a műveleteket. Használjuk a \_\_get, \_\_set metódusokat a private adattagok elérésére. Használjuk a *\_\_toString()* metódust objektum kiíratásra.

class Calculator {

private $val1, $val2;

public function add() {

}

public function subtract() {

}

public function multiply() {

}

public function divide() {

}

}

1. Az előző osztályból (*Calculator*) származtassunk egy *AdvCalculator* osztályt, amely képes tetszőleges hatványra emelni, négyzetgyököt vonni.
2. Készítsünk egy *BankAccount* nevű osztályt *accountNumber* és *accountName* adattagokkal. Az osztálynak legyen konstruktora és destruktora. Az adattagoknak megfelelően készítsük el a getter és setter metódusokat. Származtassunk egy *SavingsAccount* osztályt, amelynek lesz *interestRate* mezője. Itt is van konstruktor és lehetőség van az interestRate beállítására illetve lekérdezésére valamint a kamat kiszámítására. Teszteljük az implementált osztályokat.
3. A fenti *bankAccount* osztályt egészítsük ki egy *balance* mezővel, amely az egyenleget tárolja. Továbbá lehetőség van betenni illetve kivenni (Withdraw, Deposit). Az egyenleget le kell tudni kérdezni. Teszteljed a kódot.

**Házi feladatok:**

1. Egy könyvtár nyilvántartó szimulálása (AbstractLibrary, Library, Author, Book, index). A program váza (és az utasítások) elérhető a Moodle-ben.

**vagy**

1. Egy bevásárló kosár szimulálása (CartItem, Cart, Product, index). A program váza (és az utasítások) elérhető a Moodle-ben.