Kedves jelentkező,

Az alábbi kérdések közül pár darab konkrét megoldást var ahol a válasz vagy jó vagy sem de van közöttük több amiknél nincs jó vagy rossz válasz. Sokkal inkább arra vagyunk kíváncsiak, hogy te mit gondolsz. Ezekre a kérdésekre elege csak röviden, egy-két mondatban válaszolni. Nem Wikipedia definíciókat varunk sokkal inkább a TE gondolataidat. Ha esetleg véleményezed is a kérdéses dolgokat annak külön örülünk (például a második kérdésnél, hogy te használsz-e private tagfüggvényeket vagy ellenzed a használatukat).

1. Ha a párizsi Eiffel torony alatt állsz, Dél-Kelet fele fordulsz majd elindulsz egyenesen akkor hova jutsz (helyismeret nem szükséges a megoldáshoz)?
2. Ebben feladatban egy “tisztító” algoritmust kell megvalósítani. Ki kell szűrni koordináták egy halmazából (szelességi, hosszúsági, időbélyeg) azokat az elemeket amik feltehetőleg nem illenek a halmazba. Képzeld azt, hogy GPS segítségével rögzítünk egy utazást de mivel a GPS néha hibázik/pontatlan ezert a listába kerülhetnek teljesen oda nem illő adatok. Ezeket kell kiszűrni.  
   Mellékelve találsz egy points.php fájl-t. Megtalálod benne a **Coordinate** osztályt (és annak interface-t is) aminek példányai egy-egy koordináta pontot képviselnek (szelesség, magasság, időbélyeg). Ezen kívül találsz benne egy **Journey**\_Abstract nevű absztrakt osztályt is. A feladat, hogy implementáld a **Journey** osztályt ami kiterjeszti a **Journey\_Abstract**-ot.  
     
   Csak a **Journey** osztály megirasa a feladat, NEM kell működőkepés, futtatható PHP alkalmazást készíteni. A mellékelt point.csv fájlban találsz pár koordinátát. Ezek egy valós utazást reprezentálnak néhány hibával. A feladat kiirasa szerint nem kell sehol behívni, vagy beolvasni ezt a fájlt de a saját munkád ellenőrzésére hasznos lehet.  
     
   A munkát teszt vezérelt módon (TDD) javasolt elvégezni. Ha az egységtesztek és a TDD világa messze áll tőled akkor nem kötelező így eljárni. A teszt(ek) kerülhetnek külön fájlba is, rad van bízva, csak ne felejtsd el csatolni.
3. Mi a különbség a **public**, **protected**, **private** tagfüggvény láthatóságok között a PHP nyelvben. Milyen előnnyel bír a privát tagfüggvény használata. Mondj egy példát.
4. Adott a következő PHP script:  
     
   <?php  
   $currencies = array(  
    ‘GBP’ => 360,  
    ‘HUF’ => 1  
   );  
     
   echo $currencies[‘GBP’];  
   echo $currencies[‘EUR’];  
   ?>  
     
   Mi fog, mi jelenhet meg a kimeneten? Valtoztatnal-e valamit a scripten és ha igen, mit?
5. Mire jó a **@** operator PHP nyelvben? Hol használnád?
6. Van-e lehetőség PHP nyelvben **FATAL ERROR** eseten ‘hibakezelő funkció’ futtatására (pl email küldés, log bejegyzés, adatbázis tranzakció visszafordítás)?
7. Adott egy alkalmazás amiről annyit tudunk, hogy **php/html/js/mysql** technológiákat használ (tehát a jól bevált standard webes stack). Sajnos több panaszt kaptunk, hogy rendkívül lassan ‘tolt be’ az oldal. A feladatot te kapod, hogy nézz utána mi lehet a hiba és optimalizáld a kódot. Mit teszel? Miket nézel meg? Mikre figyelsz oda és milyen esetleges eszközöket használsz fel?
8. Linux rendszer eseten mihez fordulnál ha egy scriptet (legyen az akar php vagy barmi más) bizonyos időközönként automatikusan le kell futtatni.
9. Linux konzolban hogyan íratnád ki egy szöveges fájl tartalmat a standard kimenetre úgy, hogy ha a fájl tartalma változik akkor a kimeneten is egyből megjelenjen az. Tehát mintha egy log fájlt monitoroznál fejlesztés közben.
10. Írj Linux parancsot ami az aktuális könyvtárban fellelhető fájl-ok közül egy logfiles.txt-be írja azok fájlneveit amik nevei tartalmazzak a ‘log’ kifejezést (bármilyen formában).
11. Mi a különbség a **GET/POST/PUT/DELETE** keresek között mind **HTTP** mind **PHP** szempontból. Mondj egy-egy példát mikor használnád őket.
12. Mi a különbség a **401** és **403** **HTTP** válaszkód között. Mondj mindkettőre egy szituációt amikor használnád.
13. Miben különbözik a **https** a **http**-tol? Mikor javaslod a használatát a sima **http** helyett?
14. Hogyan védekeznél a cross-site request forgery (CSRF) támadástípus ellen. Mondj egy példát.
15. Milyen esetekben részesítenéd előnyben a **NoSQL** adatbázisokat az **SQL**-ekkel szemben?
16. Adott a két következő MySQL tábla:

|  |  |
| --- | --- |
| **Users** | **Logins** |
| user\_id (int – primary key) | login\_id (int – primary key) |
| name (varchar 32) | user\_id (int – foreign key) |
| register\_date (timestamp) | login\_date (timestamp) |

Írj egy SQL lekerest ami kilistázza azon felhasználók neveit akik 7 napnál korábban regisztráltak és több mint 5 alkalommal leptek be az oldalra. A Users table tartalmazza a felhasználókat a Logins pedig a belépéseket.