Szoftverfejlesztés gyakorlat – 08. óra

October 25

2018

OOP – Absztrakt osztály, Összetett OOP tervezéses feladat

Feladatlap

Objektumorientált tervezés

A tervezés során a valós világ elemeire vonatkozó feladat objektummodelljét állítjuk elő. Az egyes elemeket objektumokkal modellezzük. Ezt a folyamatot gyakran megszemélyesítésnek vagy antropomorf tervezésnek nevezzük.

Általánosságban az Objektum Orientált Programozás tervezés lépései a következőek:

- A feladat pontos specifikálása
- A feladat elvégzéséhez szükséges objektumok meghatározása
- Az objektumok tulajdonságainak (adattagok), tevékenységeinek (metódusok) felmérése.
- A közös tulajdonságok és tevékenységek kiemelése. (Öröklődés, Polimorfizmus)
- Az osztályhierarchia kialakítása általánosítással és specializációval
- Az osztályok/objektumok kapcsolatának, együttműködésének kiépítése.

A feladat megoldásához szükséges objektumok azonosításában segítenek a specifikációban szereplő főnevek, illetve a tevékenységeket pedig igék segítségével a legegyszerűbb meghatározni.

Absztrakt osztály

- Az Objektum Orientált fejlesztés során olyan osztályokat is kialakíthatunk, melyeket csak továbbfejlesztésre, származtatásra lehet használni, vele objektumpéldány nem készíthető, azonban objektumreferencia igen.
- Az osztály fejlécében az abstract kulcsszóval hozunk létre abstract osztályt.
- Az abstract osztályok további jellegzetessége, hogy bizonyos műveletek, amelyek szükségesek az osztály működéséhez, általában nincsenek kidolgozva – a függvény dekralációt pontosvessző zárja és nincs törzsük. Az ilyen metódusoknál is alkalmazni kell az abstract kulcsszót.
- Ilyenkor az abstract metódusok implementációját a származtatott osztályban kell megtenni. Amennyiben ezt nem tesszük meg, akkor a származtatott osztálynak is abstract-nak kell lennie!
- Ha egy osztályban van abstract metódus, akkor azt az osztály fejrésznél is jelezni kell.
- Abstract osztálynak nem kötelező minden függvényének, hogy abstract legyen. Abstract metódus nem lehet private és final!

1. Feladat – Absztrakt osztály

Készíts **Mesterember** absztrakt osztályt, mely tartalmaz:

- egy *nev* nevű string adatmezőt
- egy *napiDij* nevű adatmezőt
- egy *foglaltNapok* nevű 31 bool elemből álló tömb adatmezőt, mely tömb adott eleme azt mutatja meg, hogy az adott mesterember a hónap adott napján foglalt-e (true=szabad, false=foglalt)
- egy konstruktort, mely a fenti két adatot (név, napidíj) paraméterként kapja meg. (Kezdetben minden nap szabad a mesterember)
- egy *toString()* nevű metódust, mely az objektum valamennyi adatát adja vissza egy karaktersorozatként
- valamint egy *MunkatVallal ()* nevű bool visszatérési értékkel, és egész típusú paraméterrel rendelkező metódust. A metódus nem tartalmaz megvalósítást, ezért az legyen abstrakt!

Készíts **Burkoló** osztályt, mely a Mesterember osztály leszármazottja és tartalmaz:

- egy *szakterulet* nevű adatmezőt, amelynek a típusa egy *Helyszin* felsorolás, melynek értéke csak „Belső”, vagy „Külső” lehet.
- egy konstruktort, mely az összes szükséges adatot paraméterként kapja meg
- egy o*sszesSzabadnap()* nevű metódust, mely az adott Burkoló szabadnapjainak számát adja vissza
- egy felüldefiniált *toString()* nevű metódust, mely az adott objektum valamennyi adatát adja vissza egy karaktersorozatként!
- egy *MunkatVallal()* nevű metódust. A metódus paramétereként kapott értéke a hónap adott számú napját tartalmazza. Ellenőrizze le, hogy a Burkoló szabad még azon a napon, ha igen foglalja le és a visszatérési érték legyen igaz. Ha nem szabad, akkor a visszatérési érték legyen hamis.

Készíts **VízvezetékSzerelő** osztályt, mely az Mesterember osztály leszármazottja és tartalmaz:

- egy egész típusú adatmezőt, mely azt tartja nyilván, hogy hány év *tapasztalat*tal rendelkezik.
- egy konstruktort, mely az összes szükséges adatot paraméterként kapja meg
- egy felüldefiniált *toString()* nevű metódust, mely az adott objektum valamennyi adatát adja vissza egy karaktersorozatként!
- egy *MunkátVállal()* nevű metódust. A metódus paramétereként kapott értéke a hónap adott számú napját tartalmazza. A VízvezetékSzerelőnek 3 napra van szüksége, hogy a feladatát elvégezze. Egy napra a megadott érték előtt és egy napra a megadott érték után. Ellenőrizze le, hogy a VízvezetékSzerelő szabad még a szükséges napokon, ha igen foglalja le, s a visszatérési érték legyen igaz. Ha nem szabad, akkor a visszatérési érték legyen hamis. Ügyeljen a hónap elejére és végére.

Készíts **Szakember** osztályt, amely a fenti három osztály működését mutatja be a következő módon:

- készítsen egy 6 elemből álló listát, mely Mesterember típusú objektumok tárolására képes.
- Töltsd fel a következő elemekkel:

objektum típusa	név	napidíj	szakterület	tapasztalat
Burkoló	Tapéta Lajos	60.000	Belső	-
VízvezetékSzerelő	Megszer Elek	12.000	-	3
Burkoló	Vakolat Péter	50.000	Külső	-
VízvezetékSzerelő	Víz Elek	15.000	-	5
Burkoló	Eresz János	30.000	Külső	-

- A 6. elemet a felhasználó adja meg.
- Az alkalmazás kérdezze meg a felhasználót, hogy mely típusú mesteremberek adataira kíváncsi, azok minden adatát írassa ki.
- Szimuláljon pár megrendelést a burkolók számára!
- Majd írassa ki a legtöbb szabadnappal rendelkező Burkoló minden adatát.

2. Feladat – Árverés szimulátor

A feladatmegoldás során fontos betartani az elnevezésekre és típusokra vonatkozó megszorításokat, illetve a szövegek formázási szabályait. Segédfüggvények is létrehozhatók, a feladatban nem megkötött adattagok és

elnevezéseik is a feladat megoldójára vannak bízva. Törekedni kell arra, hogy az osztályok belső reprezentációja a lehető legjobban legyen védve, tehát csak akkor és csak olyan hozzáférés megengedett, amelyre a feladat felszólít, vagy amit az osztályt használó kódrészlet megkíván!

A megoldásnak működnie kell a mellékelt tesztprogrammal. A megírt forráskód kellően általános és újrafelhasználható legyen!

auction.Lot osztály: Az osztály egy árverési tételt (műalkotást) reprezentál.

- Az osztálynak három rejtett adattagja van: egy szöveg típusú alkotó, egy szöveg típusú cím és egy egész típusú leütési ár (angolul hammer price).
- Az osztálynak legyen egy rejtett konstruktora, amely paraméterként megkapja az alkotó nevét, a műalkotás címét, valamint a kikiáltási árat, és beállítja a megfelelő adattagokat (a leütési ár legyen a kikiáltási ár).
- Definiáljunk egy osztályszintű make nevű metódust is. A make metódus szintén az alkotó nevét, a műalkotás címét és a kikiáltási árát kapja meg paraméterként. A metódus először ellenőrzi, hogy a paraméterek megfelelőek. Amennyiben igen, akkor létrehozza és visszaadja a paramétereknek megfelelő Lot típusú objektumot. Ha a paraméterek nem megfelelőek, akkor a metódus null-t adjon vissza.
    - Az alkotó neve akkor megfelelő, ha nem egy null referencia.
    - A műalkotás címe akkor megfelelő, ha szintén nem egy null referencia, és legalább 2 karakter hosszú, csak nagybetűkből és szóközökből áll.
    - A kikiáltási ár akkor megfelelő, ha pozitív szám.

  Segítség: a metódusban használható a Character osztály isLetter() és isUpperCase() metódusa.
- Definiáljuk az osztályban az alábbi, paraméter nélküli lekérdező metódusokat: getArtist(), getTitle() és getHammerPrice(), amelyek rendre visszaadják a műalkotás címét és a leütési árat.
- Az osztálynak legyen egy bid nevű metódusa, mely visszatérési érték nélküli, és egy pozitív egész paramétert vár, és amelynek segítségével licitálni lehet az aktuális műalkotásra. A licit a következőképpen történik: ha a paraméter nagyobb, mint műalkotás leütési ára, akkor a leütési árat a paraméterrel tesszük egyenlővé. Különben nem történik semmi.
- Definiáljunk egy paraméter nélküli toString nevű metódust is, amely visszaadja az objektum szöveges reprezentációját. A formátum legyen a következő: alkotó: műalkotás címe (leütési ár GBP). Pl. Henri Matisse: JACQUY (350000 GBP), Salvador Dali: PORTRAIT DE MADAME DUCAS (500000 GBP).
- Definiáljunk egy moreExpensiveThan() metódust, mely egy műtárgyat vár paraméterül, és logikai igazat ad vissza, ha az aktuális műtárgy, melyen a metódust meghívták, drágább, mint a paraméterül kapott, továbbá a paraméter nem null.
- Vegyünk fel egy Lot típusú STATUE osztályszintű adattagot, melynek alkotója Felix W. de Weldon, címe US MARINE MEMORIAL, kikiáltási ára 50000.

auction.Auction osztály: Az osztály egy árverést reprezentál.

- Az osztály egy rejtett műtárgy-sorozat adattagban tartsa nyilván, hogy milyen műtárgyakra (Lot típusú objektumok) lehet licitálni. A típus tetszőleges, lehet rögzített méretű sorozat típus is.
- Az osztálynak legyen egy publikus konstruktora, amely műtárgyak tömbjét kapja paraméterként. A konstruktor inicializálja a sorozat adattagot a tömböt használva, ügyelve arra, hogy a belső állapot ne szivárogjon ki. Feltesszük, hogy egyik elem sem null.

- Definiáljunk egy numberOfLots nevű metódust, amely visszaadja az árverésen szereplő műtárgyak számát.
- Definiáljunk egy paraméter nélküli toString nevű metódust is, amely visszaadja az árverés szöveges reprezentációját. Az egyes alkotásokat sortörés vagy szóköz karakter is elválaszthatja. A szöveg összeállításakor a műtárgyak olyan formában szerepeljenek, ahogyan a Lot toString nevű metódusa előállítja őket. Az utolsó műtárgy után opcionálisan lehet sortörés vagy szóköz.

A **auction.Auction osztály**ban definiáljuk az alábbi publikus metódusokat:

- browseLots(): a metódus lehetővé teszi a műtárgyak közötti böngészést. Egy alkotó nevét kapja paraméterként és egy listában visszaadja azon műtárgyakat, melyek az adott alkotó művei. Ha az árverezőház nem rendelkezik egyetlen olyan műalkotással sem, mely megfelel a követelménynek, akkor a metódus egy üres listát ad vissza. A végeredmény típusa List legyen, megadva az elemek típusát is.
- priceOfCollection(): a metódus megadja, hogy mennyibe kerülne, ha egy adott alkotó összes művét szeretné megvenni egy rajongó. A metódus egy alkotó nevét várja paraméterül és egy long típusú számot ad vissza eredményül (egy gyűjtemény rengeteg pénzbe kerülhet).
- mostExpensive(): a metódus paraméter nélküli, és az árverezőház legdrágább műalkotását adja vissza (egy Lot típusú objektumot). Ha az árverezőháznak egyetlen műalkotása sincsen, akkor null-t adjunk vissza. Ha több egyformán legdrágább alkotás van, akkor az elsővel térjünk vissza.

Házi feladat - Könnyű

Írj osztályt Bor néven. Egy bor objektum attribútumai a következők: fajta, evjarat. Az attribútumok legyenek privat láthatóságuak.

Írj konstruktort a Bor osztálynak, amely a paraméterei alapján inicializálja az attribútumokat. Az osztálynak ne legyen default konstruktora. Írj publikus függvényeket, amelyekkel lekérdezhetők es módosíthatok az attribútumok. Írj toString metódust az osztálynak.

Írj osztályt Aszu néven, amely származik a Bor osztályból. Az örökölteken felül egy aszu objektum a puttony attribútummal rendelkezik.

Írj konstruktort az Aszu osztálynak, amely 2 paramétert var: puttony es evjarat. Ezeknek a paramétereknek megfelelően inicializálja az attribútumokat. A fajta attribútum aszu objektumok esetében mindig "aszu" legyen. Írj publikus függvényeket, amelyekkel lekérdezhető és módosítható a puttony attribútum. Írj toString metódust az osztálynak.

Írj futtatható osztályt. Az osztálynak legyen egy statikus metódusa kiirBor névvel, amely egy Bor tipusú paramétert fogad. A paraméterül kapott objektumot írja ki a konzolra.

Hozzon létre Bor és Aszu objektumokat, majd írja ki őket a konzolra a kiirBor metódus segítségével.

A megírt osztályokat dokumentáld javadoc kommentekkel es készíts belőlük javadoc-kal dokumentációt. Erről részletesen a https://en.wikipedia.org/wiki/Javadoc oldalon találhat információt.

Házi feladat - Nehéz

Tervezd meg a mellékelt ital UML diagram alapján az előző feladat egy összetettebb megvalósítását is. Az interface új fogalom, nézz utána az interneten, próbáld megérteni a működési elvét.