Feladat – Nagyzh – FONTOS INFÓK

- A feladat során alkalmazd a megtanult objektum-orientáltsági elveket!
- A fájlokat egy zip-be csomagolva kell feltölteni.

A feladat leírása

A feladatban egy faáru üzlet készletét kell kezelni. Az üzlet különféle fából készült termékek készítését, nyilvántartását, eladását kezeli. Jelen programban háromfajta terméket szeretnénk kezelni: széket, asztalt, illetve ruhás szekrényt. A faáruk esetén nyilván tartunk egy egyedi azonosítót (szöveg), a felhasznált fa típusát (szöveg), a súlyt (kg-ban, nem feltétlenül egész), a méreteit (szélesség, magasság, hosszúság, cm-ben), valamint az egységárát (Ft-ban). Székek esetén ismerjük még a szék stílusát (szöveg), asztal esetében a lábak számát, valamint, hogy a lábak állíthatóak-e, szekrény esetében pedig az ajtók számát, valamint, hogy van-e rajta tükör.

Az átfogó működést a **Store** osztályon keresztül kell megoldani. Legyen egy *LoadStock* metódusa, amely egy Json fájl nevét kapja, és onnan betölti az üzlet kínálatát. A Json két részre van osztva. A **Products** kulcs alatt vannak a termékek leírásai, míg a **Stock** kulcs alatti lista megadja, hogy melyik termékből (az egyedi azonosítót felhasználva) mennyi darab van aktuálisan raktáron. Feltehetjük, hogy helyesek az adatok, és minden termékhez van pontosan egy bejegyzés a raktár készletben, de a sorrend eltérhet.

A **Store** osztálynak kell egy *ListProducts* metódus, amely megjeleníti az összes terméket, valamint egy *ListStock* metódus, amely megjeleníti a termékeket az elérhető mennyiséggel együtt.

A projekthez tartozik egy másik fajta Json fájl is, ami egy vásárló által leadott rendelés adatait írja le. Ebben egy lista van, amelynek minden eleme egy termék azonosítót és egy darabszámot tárol. A **Program** osztályban legyen egy *LoadOrder* metódus, amely megkapja egy ilyen fájl nevét, beolvassa a tartalmát, és visszaad egy belőle feltöltött **Order** objektumot.

A **Store** osztálynak legyen egy *PriceOfOrder* metódusa, amely megkap paraméterben egy rendelés objektumot, és visszaadja a teljes költségét. A költségbe az áruk összes árán kívül fel kell számolni egy egyszeri ügyintézési díjat is, ami egységesen 1000 Ft, de (bár a *Main* ezt nem teszteli) legyen lehetőség az átállítására. Legyen egy *DeliverOrder* metódus is, amely a raktár készletből levonja a rendelt mennyiségeket. Amennyiben bármelyik termékből nincs elég a raktáron, a teljes rendelés legyen érvénytelen. A rendelés érvényességét a visszatérési érték jelezze.

A **Store** osztálynak legyen egy *Produce* metódusa, amely egy termék azonosítót kap és egy darabszámot, és az adott termékből elérhető mennyiséget megfelelően növeli. Legyen egy *ExportStock* metódus is, ami egy fájlnevet kap, és abba a fájlba Json formátumba kimenti a tárol termékek típusait, valamint az elérhető mennyiségeket (a formátum ugyanaz legyen, mint amit a betöltés is használ).

Pontozás

1. Termékek adatszerkezete, beolvasása, és tárolása a **Store** osztályban **(8 pont)**
2. Elérhető mennyiségek beolvasása és tárolása **(4 pont)**
3. *ListProducts* metódus **(3 pont)**
4. *ListStock* metódus **(4 pont)**
5. Rendelés adatszerkezete (**Order** osztály) és betöltése (*LoadOrder*) metódus **(6 pont)**
6. *PriceOfOrder* metódus **(4 pont)**
7. Az egyszeri ügyintézési díj mértéke állítható **(2 pont)**
8. *DeliverOrder* metódus **(4 pont)**
9. *Produce* metódus **(2 pont)**
10. *ExportStock* metódus **(3 pont)**