Készítsen versenybeosztó rendszert

Szükség volna az Ön szakértelmére a Tour de NIK verseny lebonyolítása keretein belül. A versenyzők kerékpárjait kell megfelelő módon elrendezni és ehhez Önt kérték fel, hogy biztosítsa a technikai hátteret.

E kerékpártároló rendszerben különböző versenyzők kerülnek felvitelre, más-más helyekről és teljesítménnyel, ezért, hogy a rendszer megfelelően működjön egy `IVersenyzo` interfészen keresztül biztosítsa a működést, ami tartalmaz:

- *Név* tulajdonságot
- *VersenyzőAzonosító* tulajdonságot
- *Fogyasztás* (óra) metódust, amely megadja, hogy adott versenyző egy verseny alatt hány liter folyadékot kér
- *Terhelés* (óra) metódust, amely adott versenyzőt leterhel a versenyórák számával
- *Teherbírás* ( ) metódust, amely adott versenyző teherbírását adja vissza, amely 0 és 1 közötti érték
- *TerhelhetőMég* ( ) metódust, amely igaz/hamis értéket ad vissza
- *Versenyek tulajdonságot, amely lista formájában tárol el Verseny típusú elemeket*

Készítsen el és építsen ki megfelelő osztályhierarchiát, amelyekkel megvalósíthatja a feladatokat. Tartalmazzon absztrakt és lezárt osztályokat, ahol az életszerűség megköveteli. A szükséges osztályok valósítsák meg a fentebbi interfészt, és ezekből hozzon létre mintapéldányokat.

- Pl. Edsger Wybe Dijksrta (név: EWD, versenyzőAzonosító: F-123AA, lakhely: ...)

Biztosítsa, hogy a minimálisan szükséges adatok mellett, legyenek még az életszerűséget reprezentáló mezők/tulajdonságok és metódusok az egyes osztályokban! A `Verseny` osztály egy string típusú megnevezés mezőt és hozzá tartozó tulajdonságot tartalmazzon minimum, tetszés szerint ha a feladat megköveteli bővíthető.

Készítsen egy `VersenyBrigad` osztályt, ahol n darab indulót tudunk egy csoportba tenni. A belső adatszerkezet rendezett láncolt lista legyen. A rendezettséget a versenyzők azonosítójának számjegyeket tartalmazó részét vegye alapul. Egy darab versenybrigád példány, egy darab brigádba tartozó indulókat reprezentál. A listánál biztosítsa, hogy az utolsó elemből elérhető a legelső, feldolgozáskor pedig vegye ezt figyelembe, ne alakuljon ki emiatt végtelen kör.

Versenyzők létrehozását egy dedikált, `VersenyzoKezelo` osztályon keresztül végezze, metódusokkal. A Main függvényből ne legyen direktben példány lértehozva, ezt a funkciót szervezze ki ide, az osztályba. Hasonlóképpen, adott versenybrigádhoz is ezen a vezérlő osztályon keresztül tudjon indulót hozzáadni. Szintén ezen osztályon belül:

[1] Készítsen metódust, amellyel *VersenyBrigad* típusokat tudunk felvinni, amelyeket az osztályon belül egy láncolt lista adatszerkezetben tároljon el. A listát legyen képes rendezni. A rendezés alapján a versenybrigád első emberének versenyzői azonosítója (azon belül a 3 számjegy) adja a sorrendet.

[2] Készítsen egy belső használatra való metódust, amely az induló azonosítójának generálását végzi el. A generálás eredménye, egy 5 karakterből álló string kell legyen, ahol az első karakter a férfi/nő megfeleltetést jelenti. Ezt jelölje F (mint férfi) vagy N (mint nő). Ezt követően kötőjellel elválasztva egy generált 3 számjegyű érték, majd egyből utána két darab generált nagybetűs karakter.

[3] A versenyzők rendelkezzenek egy *NemVersenyzikTobbet* nevű eseménnyel. Amennyiben ez aktiválódik, akkor erről a szükséges rendszer is értesüljön és távolítsa el a versenyzőt az adott versenybrigádból. Ha ő volt az utolsó ember e csoportban, úgy a versenybrigádot is szüntesse meg, és erről esemény formájában jelezzen.

[4] Készítsen egy optimális brigádbeosztás funkciót, amely visszalépéses keresés segítségével minden egyes brigádot megvizsgál és azon belül a csoportban úgy osztja be az embereket, hogy mindenki közel hasonló mennyiségű versenyen vegyen részt. A versenyek száma mellett, vegye figyelembe egy versenyző leterhelhetőségét is. Túl- vagy alulterhelés esete lehetőleg ne álljon fent. Túlterhelésről akkor beszélhetünk, ha a versenyző 95%-osan vagy a fölött van terhelve, alulterhelésről pedig ha 15% vagy annál alacsonyabb. Ehhez használja a versenyző versenyek listáját, amelyből látható, hogy kit mely versenyen lehet indítani.

A feladat megoldása során tartsa be a tanult OOP alapelveket, kivételkezelés segítségével kezelje a felmerülő problémás eseteket. Ahol szükségesnek érzi, egészítse ki a megadott osztályhierarchiát, illetve használja a tanult technikákat (pl. generikus típusok). A tananyaghoz kapcsolódó adatszerkezeteket (lista, bináris keresőfa, stb.) ön valósítsa meg és működésükhez implementálja a szükséges metódusokat (pl. Hozzáadás, Törlés, Keresés, Beszúrás).