11. heti feladatok

1 Készítsen egy osztályt (**SetOfInts**), amely egész számok rendezett, ismétlődő elemeket nem tartalmazó listáját tárolja el.

- Az osztály egy példányának létrehozásakor a konstruktor ellenőrizze le, hogy a paraméterként átadott egészek tömbje rendezett-e, és minden elem egyedi-e.
- A tömbben lehessen keresni egy egész számot. (*Bináris keresés*)
- Legyen egy `bool Subset(SetOfInts other)` metódusa, ami határozza meg, hogy a `other` részhalmaza-e annak a halmaznak, amire meghívjuk a metódust.
- Legyen egy `SetOfInts Intersection(SetOfInts)` metódusa, ami előállítja két egészeket tartalmazó halmaz metszetét.
- A metszethez hasonlóan implementálja az unió, differencia és szimmetrikus differencia előállítását lehetővé tevő metódusokat is.

2 Készítsen egy alkalmazást, ami egy meetingen résztvevők listájának kezelésére alkalmas. Minden meetingen vannak elvárt és opcionális résztvevők, akik meghívást kapnak a meetingre, és a meetinget követően tudjuk, hogy kik vettek részt a meetingen. Az implementáció érdekében javasolt, hogy implementáljon egy **ParticipantList** osztályt, amely képes eltárolni résztvevők rendezett, ismétlődő elemeket nem tartalmazó listáját. Ezt az osztályt úgy érdemes megvalósítani, hogy ne függjön (vagy csak minimálisan függjön) a résztvevőket reprezentáló osztálytól. Ennek érdekében érdemes azt meggondolni, hogy ha egy résztvevőt reprezentáló osztály megvalósítja az **IComparable** interfészt, akkor a **ParticipantList** osztályban **IComparable** elemek tömbjét tároljuk, és mindenhol az elemek összehasonlíthatóságát használjuk ki. A **Meeting** osztály pedig reprezentálhat egy konkrét meetinget.

Egy lehetséges implementáció, aminek a minél szélesebb körű tesztelése is javasolt:

- **Participant** osztály
  - Egy résztvevő adatait tartalmazza (legalább a nevét).
  - Konstruktorban adhatók meg a résztvevő adatait.
  - Megvalósítja az **IComparable** interfészt. A `CompareTo` metódusa össze tud hasonlítani két résztvevőt és 0 értéket pontosan akkor ad vissza, ha a két résztvevő azonos, azaz a felülírt `Equals` metódus szerint is egyenlők.
  - Felülírja a `ToString()` metódust.
  - Létezik sztringől parszoló statikus metódusa.
- **ParticipantList** osztály
  - Eltárolja **IComparable** elemek tömbjét.
  - Konstruktorban állítható be a tárolt elemek tömbje. Ha a tömb rendezett vagy van benne ismétlődő elem, akkor eldobja a **NotSetException** kivételt, amely tartalmazza azt a tömböt, ami nem tekinthető halmaznak.
  - A `bool Contains(object participant)` metódusa megadja, hogy a paraméterként átadott résztvevő benne van-e az eltárolt tömbben. (Bináris „eldöntés” algoritmust használjon!)

- A `bool IsSubset(ParticipantList other)` meghatározza, hogy az `other` lista minden eleme benne van-e az aktuális listában. (Részhalmaz vizsgálat algoritmust használjon!)
- A `static bool IsSubset(ParticipantList list1, ParticipantList list2)` metódus meghatározza, hogy `list1` részhalmaza-e `list2`-nek.
- A `ParticipantList Intersection(ParticipantList other)` metódus előállítja az aktuális lista és az `other` lista metszetét. (A halmazok metszete algoritmust használja!)
- A `static ParticipantList Intersection(ParticipantList list1, ParticipantList list2)` metódus előállítja a `list1` és `list2` metszetét.
- A `ParticipantList Difference(ParticipantList other)` metódus előállítja az aktuális és az `other` list különbségét. (A halmazok különbsége algoritmust használja!)
- A `static ParticipantList Difference(ParticipantList list1, ParticipantList list2)` előállítja `list1` és `list2` különbségét.
- A `ParticipantList Union(ParticipantList other)` metódus előállítja az aktuális és az `other` lista unióját. (A halmazok uniója algoritmust használja!)
- A `static ParticipantList Union(ParticipantList list1, ParticipantList list2)` előállítja `list1` és `list2` nukóját.
- Írja felül a `ToString()` metódust.
- Legyen egy `static ParticipantList Parse(string[] inputs)` metódusa, ami képes egy sztring tömbből előállítani egy új résztvevő listát. Ez az egyetlen metódus direkt módon ráfüggheti a **Participant** osztályra.

- **Meeting** osztály
  - Eltárolja a kötelező és opcionális meghívottak listáját, illetve a résztvevők listáját.
  - A két paraméteres konstruktora csak a két meghívotti listát kapja bemenetként. Ha a két listának van közös eleme, akkor valami nem OK, ezért dobjon el egy **NotDisjointRequiredOptinalListException** kivételt.
  - A három paraméteres konstruktora a két meghívotti lista mellett a résztvevők listáját is megkapja. Itt is ellenőrizze le, hogy két meghívott lista diszjunk-e, illetve ellenőrizze azt is hogy csak olyan vehessen részt a meetingen, aki meg volt hívva. Ha más is szerepelne a résztvevők listájában, akkor dobjon el egy **NotInvitedParticipantsException** kivételt.
  - Legyen egy `void AddParticipants(ParticipantList participants)` metódusa, amivel be lehet állítani a résztvevők listáját. Ha olyan résztvevő is van a listában, aki nem lett meghívva, akkor dobja el a megfelelő kivételt. Ezt a metódust csak akkor lehessen használni, ha a példányosításkor még nem adtuk meg a résztvevők listáját. Ha már adott a résztvevői lista, akkor dobjon el egy **ParticipantListEarlierAddedException** kivételt.
  - A `ParticipantList ParticipatedRequired()` adja vissza azoknak a résztvevőknek a listáját, akiknek kötelező volt a részvétele és meg is jelentek a meetingen.
  - A `ParticipantList NotParticipatedRequired()` adja vissza a kötelezően meghívottak közül azokat, akik nem jelentek meg a meetingen.
  - A `ParticipantList ParticipatedOptional()` és a `ParticipantList NotParticipatedOptional()` metódusok hasonlóan működnek az előző kettőhöz, csak az opcionális meghívottakkal foglalkoznak.
  - A `ParticipantList NotParticipated()` visszaadja mindazok listáját, akik meg voltak hívva, de nem vettek részt a meetingen.
  - A `static Meeting Parse(string[] required, string[] optional, string[] participants)`

metódus lehetővé teszi, hogy **Meeting** példányt három sztring tömbből is elő lehessen állítani.

Érdemes végiggondolni, hogy mit kéne ahhoz átalakítani, illetve milyen további osztályokra lenne szükség, hogy az alkalmazás használható legyen arra, hogy hallgatók előadásokon és laborokon való részvételét lehessen vele nyomon követni.