Toteuta lista (list) käyttäen arrayta tietorakenteena.

- Toteutuskieli C#
- Luokan nimi `IntLista`
- Käytetään kielestä löytyvää arraytä tietorakenteena
- Ei vaadita kaikkia (edes tärkeitä) listan toimintoja, kuten:
  - Iterointi, eli jäsenten läpikäynti toistorakenteella
  - Haku indeksin perusteella
  - Ei useampia konstruktorimetodeja
- Osaa käsitellä vain kokonaislukuja
- Tietorakenteena käytettävän arrayn alkuperäinen koko enintään kymmenen jäsentä
  - Suuremmat arvot aiheuttavat tehtävän hylkäämisen
  - Sinun tarvitsee siis huolehtia sen kasvattamisesta
- Julkiset metodit
  - `Add(int luku)`, lisää luku-argumentin arvon listaan (eli arrayyn, koska käytät sitä tietorakenteena), ei paluuarvoa
  - `Remove(int luku)`, poistaa ensimmäisen vastaantulevan luku-argumentin arvon listasta (eli arraysta, koska käytät sitä tietorakenteena), paluuarvo true jos poisto tapahtuu, paluuarvo false jos poisto ei tapahdu
  - `Clear()`, tyhjentää listan (eli arrayn, koska käytät sitä tietorakenteena) kaikista sinne lisätyistä jäsenistä, ei paluuarvoa
  - `Contains(int luku)`, tarkistaa sisältääkö lista (eli array, koska käytät sitä tietorakenteena) luku-argumentin arvon. Paluuarvo true jos sisältää, paluuarvo false jos ei sisällä
- Ominaisuus `Count`, joka kertoo listan (eli arrayn, koska käytät sitä tietorakenteena) sisältämien jäsenten määrän
  - Voidaan lukea ulkopuolelta, kirjoitus vain luokan sisällä
- Yksi konstruktorimetodi, jolla ei ole argumentteja
  - Olio luodaan `IntLista nums = new IntLista();`
- Toteutuksessasi oleva array voi sisältää oletuksena tulevia alkuarvoja
  - Esimerkiksi uusi kokonaislukutaulukko (int array) alustetaan aina arvoilla 0
- Ei tarvitse pohtia algoritmien suorituskykyä
  - Keskity siis siihen, että vaaditut toiminnallisuudet tapahtuvat
- Luokan ja metodien nimet (ja argumenttien tyypit) pitävät olla määritysten mukaisia
  - Poikkeaminen näistä johtaa palautuksen hylkäämiseen (koska luokkasi testataan ohjelmallisesti ja opettaja ei tee siihen mitään muutoksia)

    - Ohjelmallinen testaus tarkoittaa sitä, että kirjoittamasi luokka liitetään projektiin ja siitä pyritään tekemään instanssi (olio), jonka metodeja ja ominaisuuksia testataan (ensinnäkin että ne löytyvät, jonka jälkeen testataan argumentit ja paluuarvot ja toiminnallisuudet monilla eri tavoin)
    - Jotta testaus voidaan suorittaa (eli kääntää ohjelma) luokkasi nimi, metodien ja toiminnallisuuksien nimet, paluuarvot ja argumentit pitävät olla määritysten mukaisia
- Palautukseen liitetään vain luokan implementoiva (toteuttava) koodi, ei yhtään mitään muuta
    - Pro tip: Tee `IntLista.cs`-niminen tiedosto johon koodaat luokan ja palauta vain tämän tiedoston sisältö.
- Koodi ei saa tulostaa mitään ylimääräisiä
    - Muista siis poistaa kaikki mahdolliset tarpeettomat tulostuskutsut
    - Kommentit eivät ole tulostamista
- Määriteltyjen julkisten metodien lisäksi luokassasi voi tietenkin olla tarpeellisia yksityisiä metodeja ja muuttujia
- Sinun pitänee testailla luokkaa huomattavan monella tavalla, jotta voit varmistua sen toimivuudesta
- Huomion arvoista on myös se, että jos listaan lisäys ei toimi täysin oikein, muita toiminnallisuuksia ei voida testata
    - Toimii myös toiseen suuntaan; jos `Count` ei palauta oikeaa arvoa, ei voida varmistua `Add`:in toiminnasta (tämän toki huomaat itsekin testatessasi koodiasi)
    - Useamman toiminnon testauksessa on riippuvuuksia muihin toimintoihin
- Listasi toimii siis kuin `List<int>`, mutta huomattavasti vähemmillä toiminnallisuuksilla
    - Tässä tehtävässä pointtina on pienen olio-ohjelmointi osaamisen lisäksi arrayn peruskäytön testaaminen