Toteuta jono (queue) käyttäen listaa tietorakenteena.

- Toteutuskieli C#
- Osaa käsitellä vain merkkijonoja
- Käytetään kielestä löytyvää listaa tietorakenteena
- Luokan nimi on `StringQueue`
- Ei vaadita kaikkia (edes tärkeitä) jonon toimintoja, kuten:
  - Iterointia, eli jäsenten läpikäyntiä toistorakenteella
  - Ei useampia konstruktoreja
  - Ei nosteta virheitä
- Julkiset metodit
  - `Enqueue(string arvo)`, lisää arvo-argumentin arvon jonon (eli listan, koska käytät sitä tietorakenteena) loppuun, ei paluuarvoa
  - `Dequeue()`, poistaa ja palauttaa jonon (eli listan, koska käytät sitä tietorakenteena) ensimmäisen jäsenen. Jos jono tyhjä palauttaa merkkijonon "QUEUE EMPTY"
  - `Peek()`, palauttaa jonon (eli listan, koska käytät sitä tietorakenteena) ensimmäisen jäsenen poistamatta sitä. Jos jono tyhjä palauttaa merkkijonon "QUEUE EMPTY"
- Ominaisuus `Count`, joka kertoo jonon (eli listan, koska käytät sitä tietorakenteena) sisältämien jäsenten määrän
  - Voidaan lukea ulkopuolelta, kirjoitus vain luokan sisällä
- Vain yksi konstruktorimetodi, jolla ei ole argumentteja
  - Olio luodaan `StringQueue sq = new StringQueue();`
- 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 `StringQueue.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 monen toiminnallisuuden/ominaisuuden testaus vaatii muiden toiminnallisuuksien/ominaisuuksien oikean toiminnan
    - Esimerkiksi `Count`-ominaisuus on olennainen kun testataan `Enqueue`- ja `Dequeue` -metodeja
- Jonosi toimii siis kuin `Queue<string>`, mutta huomattavasti vähemmillä toiminnallisuuksilla
    - Tässä tehtävässä pointtina on pienen olio-ohjelmointi osaamisen lisäksi listan peruskäytön testaaminen