A feladatot **C++** nyelven kell megoldani, ügyeljünk a kód olvashatóságra és következetes elnevezésekre. Az elkészített programnak parancssoros **GNU g++** fordítóval is fordulnia kell (`g++ -ansi -pedantic container_main.cpp`).

A részfeladatok csak akkor fogadhatóak el, ha az elvárt STL-es adatszerkezetek és algoritmusok lettek alkalmazva. STL dokumentáció: **http://www.cplusplus.com/reference/stl/**

A feladat egy **container.h** nevű fejállomány elkészítése egy **Container** nevű osztállyal. Az osztályban különféle elemeket fogunk tárolni rendezetten. A feladathoz tartozik egy egyszerű tesztelő is, amely az ezen az oldalon érhető el:

https://github.com/PlCpp/bead_2/blob/master/container_main.cpp

A tesztelő minden részfeladatot külön tesztel, de nem kötelező sorban haladni. Teszteléshez elég a tesztelőt (**container_main.cpp** fájlt) fordítani és futtatni. A Container osztály függvényeit (elnevezések, paraméterek) úgy kell elkészíteni, ahogy a tesztelőben használatra kerülnek. **A tesztelő eredménye tájékoztató jellegű a végleges értékelés a bemutatás alapján történik, de az elfogadás feltétele, hogy a tesztelővel is működjön az elkészített osztály.**

- **2-es**: az osztály **tetszőleges típusú elemeket** tudjon tárolni, azaz legyen **típussal paraméterezhető**. Az elemeket olyan **STL-es adatszerkezetben tároljuk**, amely alapértelmezetten rendezi az elemeket. Tudjuk, hogy ugyanaz az elem többször is előfordulhat a gyűjteményünkben. A gyűjteményhez írjunk egy eljárást, amivel **új elemet tehetünk bele**. Legyen egy függvény, amellyel a gyűjtemény méretét kérdezhetjük le.
- **3-as**: az osztálynak legyen egy olyan függvénye, amely egy a tárolt típusnak megfelelő típusú paramétert vár és visszaadja, hogy a **kapott elem hányszor szerepel a gyűjteményünkben**. A megoldáshoz **STL-es algoritmust** vagy a **tároláshoz használt STL-es adatszerkezet tagfüggvényeit** használjuk.
- **4-es**: Az osztály tartalmazzon egy olyan **konstruktort amely egy std::vectort-t vár paraméterül**, amely a típusparaméternek megfelelő típusú elemeket tartalmaz és a vektor alapján feltölti a gyűjteményt. Készítsünk egy függvényt, amely **visszaadja a legnagyobb elemet**. Figyeljünk, hogy **konstans** környezetben is működjön a függvény és ne lehessen megváltoztatni a visszaadott elemet. Az üres gyűjtemény esetét nem kell vizsgálni.
- **5-ös**: készítsünk egy olyan **visszatérési érték nélküli eljárást**, amely kit**örli a legnagyobb elemet**, ha a gyűjtemény **üres ne csináljon semmit, de ne keletkezzen hiba** abból, hogy üres adatszerkezetből próbálunk elemet kivenni.