3 Multiset

`Multiset<class T, class F = Comparator<T>>`

Implementați o clasă template pentru multiset. Clasa oferă access rapid la elementele pe care aceasta le memorează, fără a impune o anumită restricție pe ordinea elementelor. Un element poate apărea de mai multe ori. lasa Comparator <T> va fi folosita pe post de comparator default pentru determina daca o valoare exista deja in multiset. Clasa Multiset trebuie să ofere:

- constructor fără parametrii care inițializează un multiset gol;
- constructor de copiere și operatorul de atribuire;
- metodă pentru adăugare și ștergere element din multiset (se șterge prima apariție);
- metodă care întoarce numărul de apariții ale unui element;
- metodă care verifică dacă un element se află în multiset;
- metodă care elimină toate aparițiile unui element din multiset;
- metodă care întoarce numărul de elemente distincte din multiset;
- supraîncărcarea operatorului de afișare;
- specializarea clasei Comparator pentru double, care sa considere doua valori egale daca partea zecimala este egala.

Cerințe globale și obligatorii

- Alocare dinamică a memorie;
- Indentare și comentarea adecvată a codului;
- Utilizarea unei convenții de denumire a variabilelor, metodelor și claselor, cu specificarea acesteia;
- Este interzisă folosirea STL-urilor dacă, prin folosirea lor, rezolvarea devine trivială (e.g. daca tema voastra este multiset si folositi clasa multiset din STL, nota pe rezolvare va fi 0);
- Utilizarea a cat mai multe concepte POO invatate
- Utilizarea assert pentru testarea funcționalităților;
- Tema trebuie să compileze fără a utiliza anumite flag-uri de compilare (cu excepția cazurilor în care pentru compilare este necesară o anumită versiune de C++) și să respecte standardele C++ pentru sintaxă;
- Deadline: 17 mai 23:59;