4. Beadandó feladat dokumentáció

Készítette:
Giachetta Roberto
EHA: GIRIAAT.ELTE
E-mail: groberto@inf.elte.hu

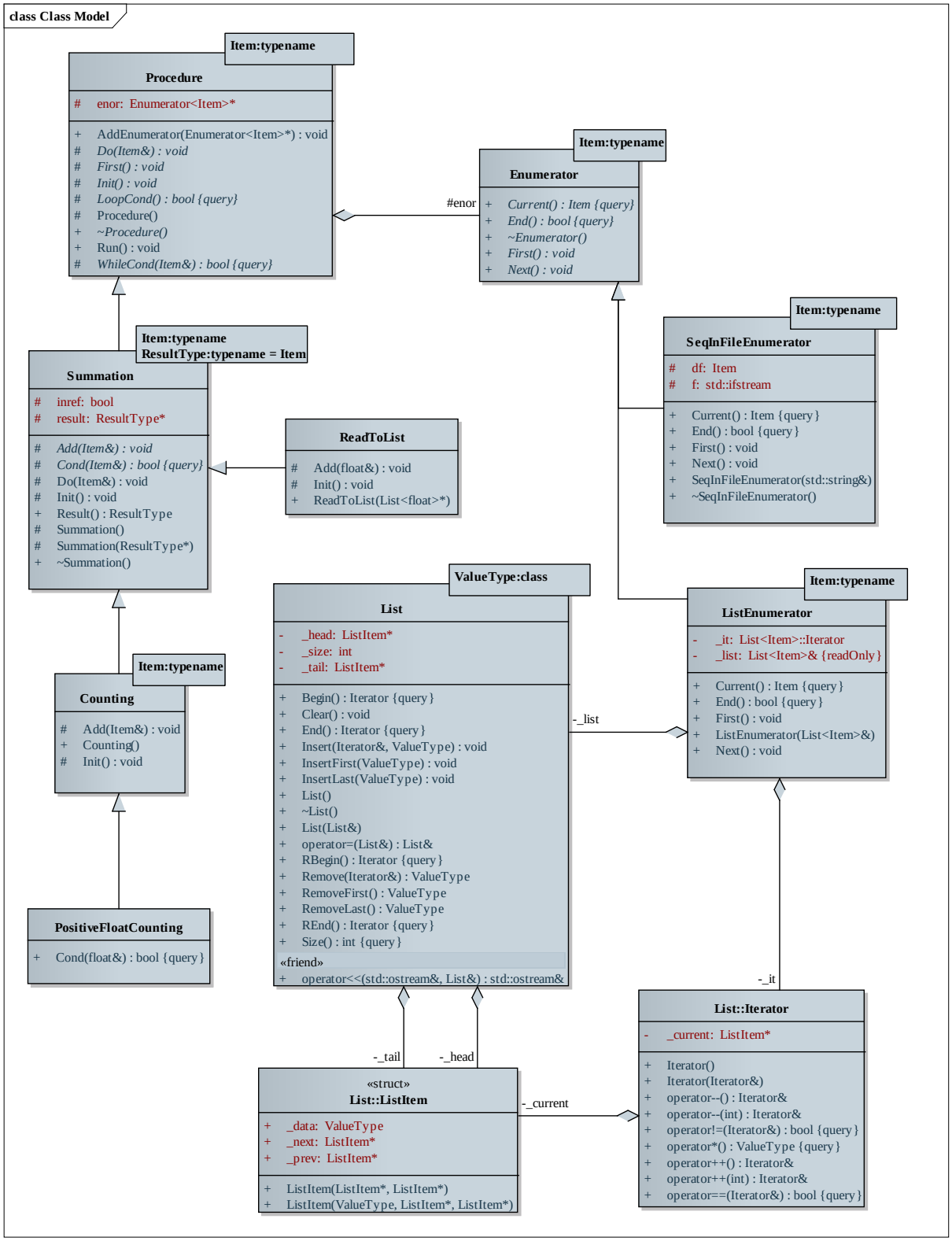
Feladat: Készítsünk egy olyan programot, amely egy szekvenciális fájlból beolvas valós számokat egy listába, majd ezt követően összeszámolja a pozitív számokat. A megoldáshoz használjuk fel az általános felsoroló/programozási tétel osztálygyűjteményt.

Elemzés:

- A feladatot két lépésben hajtjuk végre, először a fájlból listába történő beolvasást, majd a listában való számlálást.
- Felhasználjuk a rendelkezésünkre álló sablonos lista osztályt (*List*), amely rendelkezik egy saját bejáróval (*Iterator*).
- Az első részfeladathoz szükségünk van egy szekvenciális fájl felsoroló típusra, amely rendelkezésünkre áll (*SeInFileEnumerator*). A szekvenciális fájlból valós számokat olvasunk be, amelyeket egy listában gyűjtünk össze. Ez tulajdonképpen az összegzés azon speciális esete, amikor az összeg egy teljes listát ad (*ReadToList*), így az előre adott összegzés programozási tételből (*Summation*) kell örököltetnünk.
- A második részfeladat megoldásához szükségünk van egy bejáró típusra a listához, mivel a lista beépített szerkezete nem illeszkedik az általános *Enumerator* osztályhoz. Ezért létrehozzuk a lista felsorolót (*ListEnumerator*), amelyhez természetesen felhasználjuk a lista bejáró műveleteit. Erre a lista felsorolóra hajtjuk végre a pozitív számok megszámlálását egy új osztályban (*PositiveFloatCounting*), melyet az általános számlálásból (*Counting*) származtatunk.
- A főprogram feladata a felsorolók, valamint a programozási tételek példányosítása. Továbbá ügyelünk a sikertelen fájlbetöltés kivételére is.

Szerkezet:

- Az általános általános felsoroló/programozási tétel osztálygyűjteményből felhasználjuk a következő osztályokat: általános tevékenység (*Procedure*), összegzés (*Summation*), számlálás (*Counting*), általános felsoroló (*Enumerator*), szekvenciális fájl felsoroló (*SeqInFileEnumerator*)
- A listát három osztály alkotja: listaelem (*ListItem*), bejáró (*Iterator*), valamint a teljes lista (*List*), amelynek előbbi kettő beágyazott osztálya lesz.
- Az *Enumerator* osztályból származó *ListEnumerator* szintén sablonos osztály lesz, lényegében a konstruktorban paraméterként kapott listára hoz létre egy bejárót, amelynek műveleteit felhasználja, azokban megfelelően definiáljuk felül a *First*, *Next*, *End* és *Current* műveleteket.
- A *Summation* osztályból leszármazó *ReadToList* elemtípusa a *float*, eredménytípusa a *List<float>* lesz, amit a konstruktor átvesz paraméterként. az *Init* művelet kiüríti a listát, míg az *InsertLast* művelet beszúr egy elemet a lista végére.
- a *PositiveFloatCounting* osztályt a *Counting<float>*-ból származtatjuk, itt csak a *Cond* műveletet kell feldefiniálnunk, amely megadja, hogy a szám pozitív-e.



1. ábra: A program osztálydiagramja

Implementáció:

- Mivel az osztályok mind sablonosak, nem választjuk külön a megvalósítást az osztályfelület leírástól.
- A megvalósításhoz a következő modulokat használjuk: **`enumerator.hpp`**, **`seqinfileenumerator.hpp`**, **`listenumerator.hpp`**, **`list.hpp`**, **`procedure.hpp`**, **`summation.hpp`**, **`counting.hpp`**. Továbbá a két speciális programozási tételt (*ReadToList*, *PositiveFloatCounting*) a főprogram fájljában (**`main.cpp`**) helyezzük el.

Tesztelés:

- **Egységtesztek:**
  - A lista műveleteinek tesztelése valós számokkal, és szöveggel a sablon helyén.
  - A lista felsoroló tesztelése üres listára, hosszú (egy millió adatot tartalmazó) listára, valamint nem létező listára.
  - Pozitív számlálás tesztelése üres sorozatot tartalmazó fájlra, valamint hosszú fájlra.
  - Listába való beolvasás tesztelése üres fájlra, hosszú fájlra.
- **Integrációs tesztek:**
  - Pozitív számlálás tesztelése üres sorozatot tartalmazó listára, valamint hosszú listára.
  - Lista manipulálása (elem hozzáadás, kivétel, kiürítés) a beolvasás és a feldolgozás között.
- **Rendszertesztek:**
  - Program futtatása nem létező fájlra, üres fájlra, valamint hosszú fájlra.