Tapasztalatok az osztály-sablon könyvtár bevezetésével

Gregorics Tibor, Nagy András

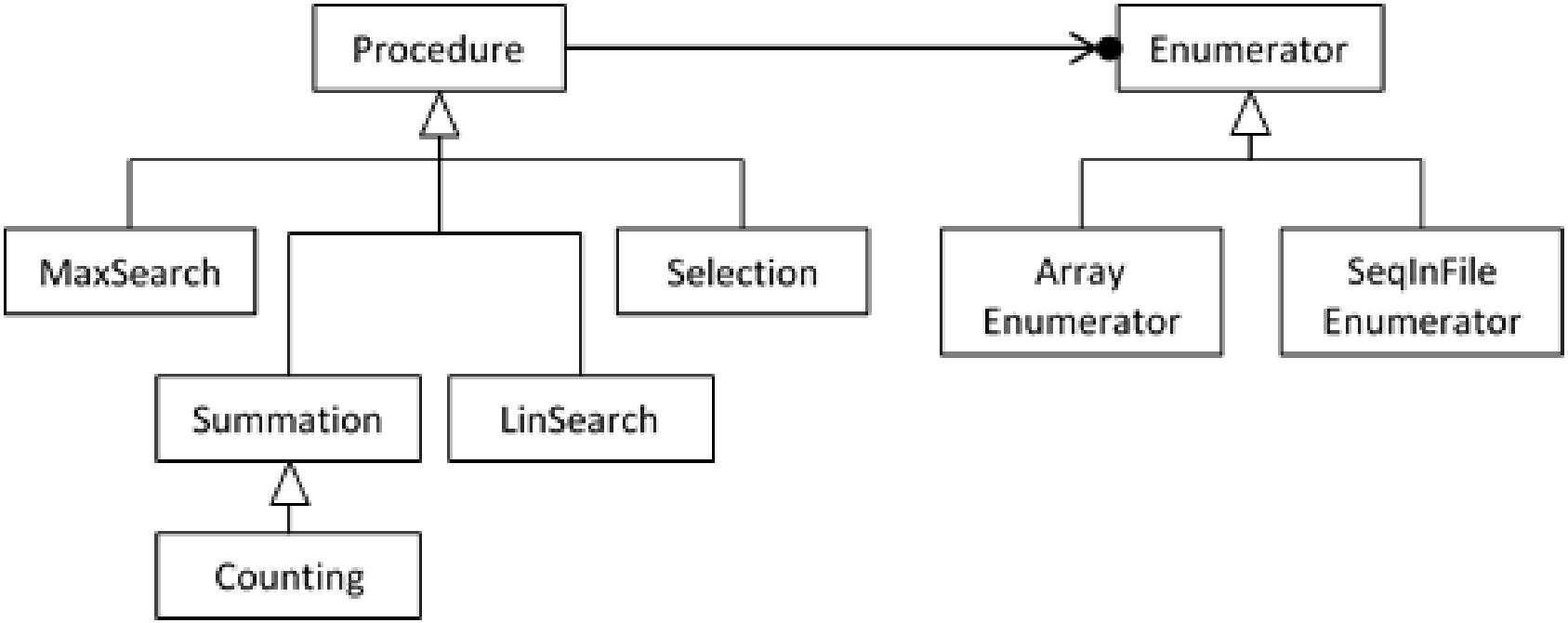
2018. november 24.

SZÉCHENYI 2020
MAGYARORSZÁG KORMÁNYA
Európai Unió
Európai Regionális Fejlesztési Alap
BEFEKTETÉS A JÖVŐBE

Osztálysablon könyvtár bemutatása

- Nevezetes programozási tételeket leíró osztályokat tartalmaz.
- Konkrét feladatok megoldása esetén leszármazással dolgozunk, visszavezetve a problémát egy nevezetes tételre.
- A leszármazott osztályban az adott feladatra jellemző tulajdonságokat és értékeket adhatjuk meg. (Pl. számlálásnál a feltételt)
- A feldolgozandó elemek felsorolóját futási időben adjuk meg. (Függőség befecskendezés)

Osztálysablon könyvtár bemutatása



run() metódus

```
init();
for (enor->first(); !enor->end(); enor->next()) {
    body(enor->current());
}
```

Osztálysablon könyvtár bemutatása - Példa

Feladat

Számítsuk ki egy C++ vektorban elhelyezett számok összegét.

Az alábbiakat kell meghatározni a megoldásához:

- Összegzéssel oldjuk meg a feladatot.
- A feldolgozandó elemek, valamint az eredmény típusa is ugyanaz (egész szám).
- Az összegző művelet egyszerű összeadás, a neutrális elem a 0.
- Egy *std::vector* elemeit kell felsorolni.

Megoldó osztály

```
class SimpleSum : public Summation<int,int> {
public:
 int add(const int& a, const int& b) const {
  return a + b;
 }
 int neutral() const {return 0;}
 int func(const int& e) {return e;}
}
```

Osztály használata

```
SimpleSum sum;
ArrayEnumerator e({1,2,3});
sum.addEnumerator(&e);
sum.run();
```

Az osztálysablon könyvtár bevezetése az oktatásba

- 2009-ben vezettük be az oktatásba.
- A célja az újrafelhasználás gyakorlása volt.
- Számos negatív és pozitív tapasztalat gyűlt össze.
- Ezeket tapasztalatokat felhasználva megreformáltuk a könyvtárat.

Pozitívumok

- Robusztus könyvtár, sok feladat megoldására alkalmas.
- Objektumorientált technológiák bemutatása, gyakorlása. (leszármazás, függőség befecskendezés, stb..)
- Visszavezetéses technikák gyakorlása.
- Rákényszeríti a hallgatót az elemzésre, tervezésre egyszerű feladatok esetén is.
- C++ objektumorientált nyelvi elemek mélyítése, gyakorlása. (sablonok, metódusok felül definiálása, stb..)

Problémák, kritikák

- Kritikák
  - Iparban nem használt könyvtár.
  - Nagyobb kódméret, túl erős technológiákra épül.
  - Elavult C++ nyelvi konvenciók.
- Problémák
  - Nem szakszerűen használták: Egyre bővülő szabályrendszert kellett bevezetni, hogy helyesen használják a hallgatók (mit lehet felüldefiniálni, kiből lehet származni, stb..)
  - Összegzés tétele túl általános volt.

Összegzés módosításai

Új összegzés metódusai

```
void  init() override final{_result=neutral();}
void  body(const Item& e) override final {
 if(cond(e))_result = add(_result,func(e));
}
void  cond(const Item& e){return true;}
Value add(const Item& a, const Item& b) const = 0;
Value neutral() const = 0;
Value func(const Item& e) const = 0;
```

- Neutrális elem, összeadás művelet, leképező művelet bevezetése.
- Az *add* nem volt konstans, nagyon általános volt, ő változtatta meg az eredmény értékét tetszőleges módon.

Új nyelvi elemek

- Final: Nem engedi a leszármazott osztályokban felül definiálni az adott metódust. (pl. az összegzésben a body-t) Nem kell külön kikötni, mit nem definiálhatnak felül a használata során.
- Override: Jelzi, hogy felül definiáljuk egy ősosztály metódusát. Csökkenti a hibázási lehetőségeket a zárthelyiken, ha a hallgatók is használják.
- Barát osztályok bevezetése: Ezzel korlátozzuk, hogy a *Procedure* ősosztályból csak a tételek osztályai származhatnak le. (Nem kell külön kikötni, hogy nem származhatnak le az ősosztályból.)
- Template specializáció: Két helyen alkalmaztuk. Egyrészt a *SeqInFileEnumerator*-nál (a konstruktorban hívott *create_ifstream()* metódusra), amikor egy szöveges állomány elemeit karakterenként kell olvasni, másrészt a *Summation*-nél (a *body* metódusra), amikor az összegzés eredménye egy kimeneti folyam
- Egyebek: *nullptr*, *#pagrma once*.

Új felsorolók

StringStreamEnumerator

- Olyankor hasznos, amikor egy sztringből (például egy szöveges állomány egy sorából) előre nem ismert számú adatok kell kinyerni.
- C++ specifikus.
- Példa: Egy sor egy receptet tartalmaz (pl: "tej 1 liter búzadara 13 evőkanál vaj 6 dkg cukor 5 evőkanál") A recept összetevőit szeretnék eltárolni egy tömbben.
- Összegzéssel megoldva: *StringStreamEnumerator* sorolja fel egyesével a recept összetevőit, mint olyan struktúrákat, amelyek anyagnévből, mennyiségből és mértékegységből állnak

IntervalEnumerator

- Egy $[m, n]$ diszkrét intervallum értékeit sorolja fel.
- Példa: Egy n szám faktoriálisa összegzéssel. Ilyenkor az intervallum $[2, n]$, az *add* művelet egy összeszorzás, a semleges elem pedig az *1*.

Összefoglalás, eredmények

- Ma már a hallgatók egy másik tantárgy keretében találkoznak a könyvtárral. Ott, ahol a felsorolós programozási tételek megismerése után az azokra történő visszavezetési technikát tanulják
- Javult a könyvtár kódjának tisztasága.
- Az új nyelvi elemekkel elősegítettük a szakszerű használatot, nincs szükség annyi szöveges kikötésre.
- A könyvtár hasznos didaktikai eszköznek bizonyult. (Hangsúlyozni kell a hallgatóknak a könyvtár célját..)

Köszönöm a figyelmet!

A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg (EFOP-3.6.3-VEKOP-16-2017-00002).