OBJEKTUM ELVŰ ALKALMAZÁSOK FEJLESZTÉSE
Dokumentáció a 4. házi feladathoz

Név: Nagy Richárd Tibor
Neptun kód: GWSAZV
Elérhetőség: ricso1996@gmail.com

Csoport: 3.
Feladatszám: 4.
2016.11.30

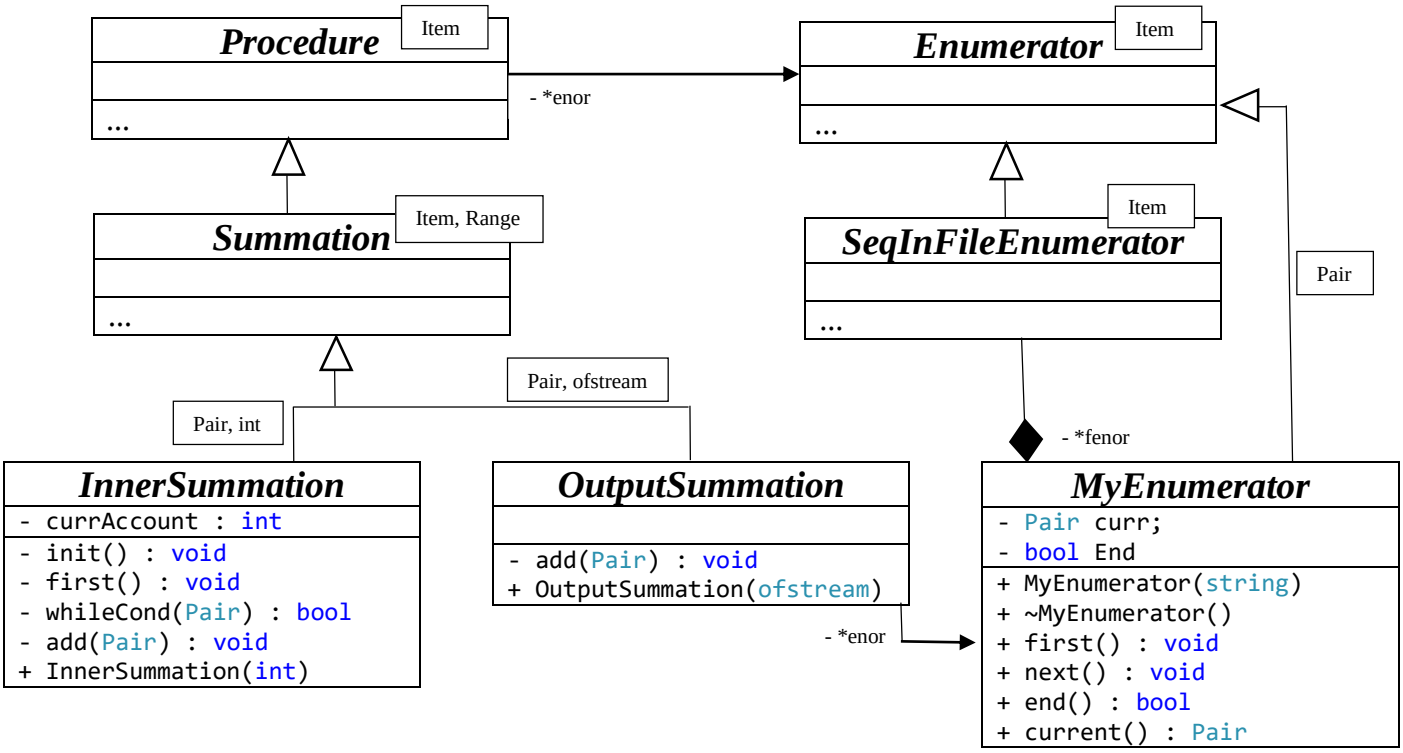
Feladatleírás

Egy szöveges állományban számlaszám-befizetés párokat tartalmazó sorokat helyeztünk el. (A számlaszám 8 karakter hosszú, utána egy szóköz jön, azt követően pedig egy egész szám.) Az állomány számlaszám szerint növekedően rendezett. Írjuk ki egy szöveges állományba az egyes számlák összesített forgalmát számlaszám-forgalom párokat tartalmazó sorok formájában! (A kiíratáson kívül csak egyetlenegy üres „else” ágú elágazást használjon!)

Programterv

A feladat megoldásához egy felsorolót hozunk létre, ami a végeredményként várt számlaszám-forgalom párokat sorolja fel. A kimeneti fájlba ennek elemeit összegezzük. Az egyes párok megállapítására a felsoroló next() művelete szintén egy összegzést végez.

Az összegzések megvalósításához kód-újrafelhasználást alkalmazunk. Felhasználjuk a `Procedure`, `Enumerator`, `Summation` és a `SeqInFileEnumerator` osztály sablonokat, valamint ezekből származtatva definiáljuk az `InnerSummation`, `OutputSummation` valamint a `MyEnumerator` osztályokat.



Megvalósítás:

Definiáljuk a felsorolni és összegezni kívánt elemek típusát, és kiterjesztjük rájuk a << és a >> operátorokat.

```
struct Pair {
	int accoutNumber;
	int balance;
	friend ifstream& operator>> (ifstream&, Pair&);
	friend ofstream& operator<< (ifstream&, const Pair&);
};

ifstream& operator>> (ifstream& f, Pair& pair){
	f >> pair.accoutNumber >> pair.balance;
	return f;
}

ofstream& operator<< (ofstream& f, const Pair& pair) {
	f << pair.accoutNumber << ' ' << pair.balance << endl;
	return f;
}
```

A felsorolót a párokra példányosított Enumerator osztályból származtatjuk.

```
class MyEnumerator : public Enumerator<Pair> {
private:
	SeqInFileEnumerator<Pair> *fenor;
	Pair curr;
	bool End;
public:
	MyEnumerator(string input) {
		try	{
			fenor = new SeqInFileEnumerator<Pair>(input);
		}
		catch (SeqInFileEnumerator<Pair>::Exceptions){
			cerr << "Nem letezo file!";
		}
	}
	~MyEnumerator() {
		delete fenor;
	}
	void first() {
		fenor->first();
		next();
	}
	void next();
	bool end() const {
		return End;
	}
	Pair current() const { return curr; }
};
```

A felsoroló next() műveletéhez szükségünk lesz a belső összegzésre, amit a párokra és int eredményre példányosított Summation osztályból származtatunk.

```
class InnerSummation : public Summation<Pair, int> {
private:
	int currAccount;
	void init() { *_result = 0; }
	void first() {}
	bool whileCond(const Pair& pair) const { return currAccount == pair.accoutNumber; }
	void add(const Pair& pair) { *_result += pair.balance; }
public:
	InnerSummation(int accountNumber) : Summation<Pair, int>(), currAccount(accountNumber) {}
};
```

Ezt felhasználva felsoroló next() művelete a következő:

```
void MyEnumerator::next() {
	End = fenor->end();
	if (!End) {
		curr.accoutNumber = fenor->current().accoutNumber;
		InnerSummation is(fenor->current().accoutNumber);
		is.addEnumerator(fenor);
		is.run();
		curr.balance = is.result();
	}
}
```

Az output fájlba való összegzést végző OutputSummation-t a párokra és ofstream eredményre példányosított Summation osztályból származtatjuk és így definiáljuk:

```
class OutputSummation : public Summation<Pair, ofstream> {
private:
	void add(const Pair &pair) { *_result << pair; }
public:
	OutputSummation(ofstream *output) : Summation<Pair, ofstream>(output) { }
};
```

Végül a főprogram a következő:

```
int main() {
	ofstream outfile("output.txt");
	OutputSummation out(&outfile);

	MyEnumerator *me = new MyEnumerator("input.txt");
	out.addEnumerator(me);

	out.run();

	delete me;
    return 0;
}
```

Tesztelési terv

1. Üres fájl
2. Csak elválasztójeleket tartalmazó fájl.
3. A belső összegzés tesztelése intervallum hossza szerint:
    a. Egy hosszú intervallum
    b. Több hosszú intervallum
4. A külső, fájlba való összegzés tesztelése intervallum hossza szerint:
    a. Nulla hosszú intervallum
    b. Egy hosszú intervallum
    c. Több hosszú intervallum