

e-Banking

Termín odevzdání:	16.04.2017 23:59:59	879654.325 sec
Hodnocení:	0.0000	
Max. hodnocení:	5.0000 (bez bonusů)	
Odevzdaná řešení:	0 / 20 Volné pokusy + 20 Penalizované pokusy (-2 % penalizace za každé odevzdání)	
Nápovědy:	0 / 2 Volné nápovědy + 2 Penalizované nápovědy (-10 % penalizace za každou nápovědu)	

Úkolem je realizovat třídu CBank, která bude implementovat internetové bankovníctví.

Uvažovaná třída je specifická tím, že chceme ukládat všechny změny (všechny transakce). Dále, pro účely archivace chceme ukládat nezávislé kopie instancí.

Úloha je směřována k procvičení pochopení mělké a hluboké kopie objektu. Má části povinné, dobrovolné a bonusové. Pro splnění povinné části úlohy postačuje vytvořit funkční třídu, která bude splňovat požadované rozhraní. Pro zvládnutí nepovinných a bonusových částí je potřeba rozmyslet ukládání dat tak, aby se při kopírování zbytečně neplýtkalo pamětí.

Úloha má procvičit práci s kopírováním objektů. Z tohoto důvodu jsou v úloze potlačené části standardní C++ knihovny, zejména STL a datový typ C++ string.

Požadovaná třída CBank má následující rozhraní:

implicitní konstruktor

vytvoří prázdnou instanci el. bankovníctví,

kopírující konstruktor

vytvoří hlubokou kopii instance. Implementujte jej, pokud kompilátorem automaticky vytvořený kopírující konstruktor nevyhovuje (zkuste si odhadnout - bude potřeba?),

destruktor

uvolní prostředky alokované instancí.

operátor =

zkopíruje obsah jedné instance do druhé (hluboká kopie). Implementujte jej, pokud kompilátorem automaticky vytvořený operátor = nevyhoví,

NewAccount(accID,initBalance)

metoda přidá nové konto:

- accID udává jednoznačný identifikátor konta. Jedná se o řetězec, délka a obsah může být libovolný, je požadována pouze unikátnost,
- initBalance udává počáteční vklad (celé číslo). Implementovaná banka je velmi vstřícná, za založení konta je ochotna i vyplatit prémii, tedy počáteční vklad může být i záporný.

Metoda NewAccount vrací hodnotu true pro signalizaci úspěchu, false pro neúspěch (konto se stejným id již bylo zadáno).

Transaction(debitAcc,creditAcct,amount,sign)

metoda realizuje bezhotovostní převod z jednoho konta na druhé. Parametry jsou:

- debitAcc udává identifikátor konta plátce (z něj se částka odečte),
- creditAcc udává identifikátor konta příjemce (na něj se částka přičte),
- amount udává převáděnou částku (není záporná, může být nulová nebo kladná),
- sign udává elektronický podpis autorizace převodu. Jedná se o řetězec, který je potřeba uložit pro účely dokladování (aplikace s ním jinak nemusí nijak pracovat).

Metoda Transaction vrací hodnotu true pro signalizaci úspěchu, false pro neúspěch (zadané konto neexistuje).

Metoda nekontroluje zůstatek na účtu, předpokládáme, že lze konto bez problémů přecherpat. Přebod ale selže, pokud se konto plátce shoduje s kontem příjemce.

TrimAccount(accID)

metoda odmaže archivované transakce z daného konta. Tedy aktuální zůstatek na kontě se stane počátečním stavem a archivované transakce zmizí. Volání lze chápat tak, že klient potvrdil předchozí transakce, tedy nemá cenu je dále archivovat. Metoda TrimAccount vrací hodnotu true pro signalizaci úspěchu, false pro neúspěch (zadané konto neexistuje).

cout << Account(accID)

výstup metody Account zasláný do streamu zobrazí počáteční stav konta, seznam archivovaných transakcí a konečný stav konta. Formát je zřejmý z ukázky. Pokud účet zadaného čísla neexistuje, je vyhozena (libovolná) výjimka.

Account(accID) . Balance()

výstup metody Account musí reagovat na volání Balance, toto volání vrátí konečný stav zadaného konta. Pokud účet zadaného čísla neexistuje, je vyhozena (libovolná) výjimka.

Odevzdávejte soubor, který obsahuje implementovanou třídu CBank a další Vaše podpůrné třídy. Třída musí splňovat veřejné rozhraní podle ukázky - pokud Vámi odevzdané řešení nebude obsahovat popsané rozhraní, dojde k chybě při kompilaci. Do třídy si ale můžete doplnit další metody (veřejné nebo i privátní) a členské proměnné. Odevzdávaný soubor musí obsahovat jak deklaraci třídy (popis rozhraní) tak i definice metod, konstruktoru a destrukturu. Je jedno, zda jsou metody implementované inline nebo odděleně. Odevzdávaný soubor nesmí obsahovat vkládání hlavičkových souborů a funkci main. Funkce main a vkládání hlavičkových souborů může zůstat, ale pouze obalené direktivami podmíněného překladu jako v ukázce níže.

Pokud se rozhodnete řešit i nepovinné a bonusové části úlohy, můžete pro nalezení vhodné reprezentace využít následující pozorování:

- Vzniká mnoho kopií instance, které ale dále nejsou měněné. Tedy vyplatí se uvažovat realizaci, kde jsou data sdílená s originálem dokud nedojde k první modifikaci originálu nebo kopie.
- Změny v instanci jsou "malé", tedy velká část obsahu zůstane stejná jako v originálu. Vyplatí se tedy i uvažovat o sdílení nezměněných částí instance. Pokud byste si např. konta realizovali jako třídy, dá se předpokládat, že mezi originálem a kopií se změní pouze několik kont.

Rozhraní a ukázka použití třídy je dostupná v přiloženém archivu.

Vzorová data:

Download

Odevzdat:

Vybrat soubor

Soubor nevybrán

Odevzdat



Referenční řešení