universität wien

Und wieder die Klasse Vector

Kopierkonstruktor und Kopierzuweisungsoperator

Diese Unterlagen sind verfügbar unter /home/Xchange/ue8/uebung8.pdf

Ziel

Unsere Klasse Vector alloziert Speicherplatz mit new, der im Destruktor wieder freigegeben wird.

Wir wissen, dass in diesem Fall für ein korrektes Funktionieren der Klasse auch ein Kopierkonstruktor und ein Kopierzuweisungsoperator implementiert werden müssen.

1. Schreiben Sie ein Programm, das Vector verwendet und die Fehler aufzeigt, die durch das Fehlen der beiden Methoden verursacht werden.
2. Implementieren Sie die beiden Methoden.
3. Prüfen Sie mit Ihrem Testprogramm, ob die Fehler nun behoben sind.
4. Welche Fehler werden durch Ihr Testprogramm nicht erfasst?

universität wien

Eine Programmieraufgabe

Diese Unterlagen sind verfügbar unter /home/Xchange/ue8/uebung8.pdf

universität wien

Bezahlen mittels Konten

Ein Konto hat eine eindeutige Kontonummer (ganze Zahl, die beim Erstellen von Konten laufend, beginnend mit 1000000 vergeben wird), sowie einen aktuellen Kontostand (double Zahl >=0). Auf ein Konto kann ein beliebiger, positiver Betrag eingezahlt werden. Von einem Konto kann ein positiver Betrag behoben werden, so weit dadurch nicht der Kontostand negativ wird.

Eine Person hat einen Zunamen, einen Vornamen (beide nicht leer) und besitzt eine beliebige Anzahl (auch 0 ist möglich) von Konten. Eine Person kann eine Zahlung durchführen, dabei kann entweder eine Kontonummer angegeben werden, oder nicht.

universität wien

Zahlung mit Angabe einer Kontonummer

Der Betrag wird, so weit das Konto mit der entsprechenden Nummer der Person gehört und eine genügend hohe Deckung aufweist, von diesem Konto abgebucht andernfalls ist eine Exception zu werfen.

universität wien

Zahlung ohne Angabe einer Kontonummer

Der Betrag wird, vom ersten gefundenen Konto, das der Person gehört und eine genügend hohe Deckung aufweist, abgebucht. Weist keines der Konten eine genügend hohe Deckung auf, so ist zu prüfen, ob die Summe der verfügbaren Beträge aller Konten zur Zahlung ausreicht. Falls nicht, ist eine Exception zu werfen, andernfalls sind Teilbeträge von den einzelnen Konten abzubuchen, sodass der gewünschte Betrag insgesamt erreicht wird (am einfachsten von jedem Konto den gesamten verfügbaren Betrag abziehen, so lange bis der noch zu zahlende Betrag kleiner als der verfügbare Betrag am aktuell bearbeiteten Konto ist. Dann diesen Restbetrag vom aktuell bearbeiteten Konto abziehen).

Z.B.: Konten mit 100 €, 500 €, 600 € und 300 €. Zahlung von 1000 €. Die Kontostände nach der Zahlung sind 0 €, 0 €, 200 € und 300 €.

universität wien

Vorgehensweise

- Implementieren Sie die Klasse Konto mit einem Defaultkonstruktor und den Methoden einzahlen und abheben.
- Implementieren Sie die Klasse Person mit einem Konstruktor mit 3 Parametern (Vorname, Zuname, std::vector<Konto>).
- Implementieren Sie die Methoden zahlung(double betrag) und zahlung(int kontonummer, double betrag) in der Klasse Person.

- Testen Sie Ihre Klassen mit Hilfe eines geeigneten Hauptprogramms.