technische universität dortmund

fakultät für informatik

Andre Droschinsky
Roman Kalkreuth
Denis Kurz
Bernd Zey

Dortmund, den 24. Januar 2019

Praktikum zur Vorlesung Einführung in die Programmierung WS 18/19

Blatt 12

Es können 30 Punkte erreicht werden.

Aufgabe 1: Kreditkarten mit Ausnahmen und virtuellen Methoden (10 Punkte)

In dieser Aufgabe soll in Anlehnung an die Konten aus der letzten Woche eine zugehörige Kreditkarte implementiert werden. Legen Sie ein neues Projekt `Aufgabe_12_1` an. Fügen Sie die C++-Quelltextdatei `kreditkarte.cpp` und die zugehörige C++-Headerdatei `kreditkarte.h` aus dem Downloadbereich hinzu. Fügen Sie außerdem die C++-Quelltextdatei `kreditkarte_test.cpp` aus dem Downloadbereich hinzu, in deren `main`-Funktion eine Testumgebung bereitgestellt wird.
Im Folgenden soll eine Klasse für Kreditkarten vervollständigt werden, wobei eine Kreditkarte über einen Saldo und einen Überziehungsrahmen verfügt. Neben der eigentlichen Funktionalität als Zahlungsmittel sollen in der Implementierung Fehlersituationen wie Überschreitungen des Kartenlimits mit Hilfe von Ausnahmen abgefangen werden.

a) Implementieren Sie eine Methode

- `void bezahlen(int betrag)`, welche die Karte entsprechend belastet,
- `void aufladen(int betrag)`, welche den übergebenen Betrag auf der Karte gutschreibt; das maximal verfügbare Guthaben darf jedoch 5.000 € nicht überschreiten, und
- `int getSaldo() const`, welche den aktuellen Saldo der Karte zurückgibt.

Falls die Karte beim Bezahlen den maximal verfügbaren Betrag oder beim Aufladen das maximale Guthaben überschreitet, soll eine Ausnahme (`Exception`) geworfen werden. Definieren Sie hierfür in `kreditkarte.h` geeignete Ausnahme-Klassen.

_____ (3)

b) Implementieren Sie eine Methode `Kreditkarte &uebertragen(Kreditkarte &v, int betrag)`, die Geld von einer Kreditkarte auf eine andere überträgt. Das Geld wird dabei von der als Parameter übergebenen Karte abgebucht. Dabei sollen dieselben Einschränkungen wie in der vorigen Aufgabe beachtet werden. Zusätzlich gibt `uebertragen` eine Referenz auf die Kreditkarte selbst zurück. Es soll kein neues `Kreditkarte`-Objekt erzeugt werden!

_____ (2)

c) Implementieren Sie eine Methode `void setLimit(int limit)`, welche das maximale Limit der Karte ändert. Sollte der aktuelle Saldo das Limit bereits überschreiten, so soll eine entsprechende Ausnahme geworfen werden. Definieren Sie auch diese Ausnahme-Klasse.

_____ (1)

d) Erstellen Sie eine abstrakte Ausnahme-Oberklasse `CreditException` mit der virtuellen Methode `void show()`. Die Ausnahme-Klassen der vorherigen Aufgabenteile sollen von dieser abgeleitet werden und die Methode `show()` so implementieren, dass eine Ausgabe ähnlich wie in Aufgabenteil e) erfolgt.

_____ (2)

e) Ergänzen Sie die Testumgebung `kreditkarte_test.cpp` derart durch `try-/catch`-Konstrukte für `CreditExceptions`, dass die durch inkorrekte Methodenaufrufe erzeugten Fehler abgefangen werden.
Beispielausgabe:

```
!!! LimitException:          -- betrag:    3000 -- limit:         500
!!! GuthabenException:       -- betrag:    8000 -- max guthaben: 5000
!!! LimitException:          -- betrag:     900 -- limit:         200
!!! LimitUngueltigException: -- betrag:       0 -- neues limit:     0
```

Testen Sie Ihre Implementierung mit der so erweiterten Testumgebung.

_____ (2)

Aufgabe 2: Dateiein-/-ausgabe (20 Punkte)

In dieser Aufgabe geht es darum, eine Klasse `PortableBitmap` für einfache Grafikdateien zu programmieren. Sie soll eine Datei sequenziell einlesen, den Inhalt modifizieren und den modifizierten Inhalt zurück in eine (andere) Datei schreiben können. In dieser Übung beschränken sich die Modifikationsmöglichkeiten auf das Substituieren (Ersetzen) von einzelnen Pixeln.

HINWEIS: Als Hilfsmittel dürfen Sie die Header zur Dateiein- und -ausgabe `fstream`, Stringverarbeitung `string` und Container `vector` verwenden.

a) Legen Sie ein neues Projekt `Aufgabe_13_2a` an und fügen Sie die Datei `cellular.cpp` aus dem Downloadbereich hinzu. Diese Datei enthält einen sogenannten zellulären Automaten, der in der `main`-Funktion eine Ausgabe auf der Konsole produziert. Ändern Sie den Programmcode so ab, dass die Ausgabe zusätzlich in einer Textdatei gespeichert wird.

_____ (3)

b) Erstellen Sie nun die Klasse `PortableBitmap` mit folgenden Eigenschaften:

- Private Attribute `unsigned int hoehe`, `unsigned int breite`, `vector<bool> bildDaten`.
- Ein Konstruktor `PortableBitmap(string const &textDatei, char nullZeichen, char einsZeichen)`, der die Grafik mit Daten aus einer Textdatei initialisiert, wie wir sie in Aufgabenteil a) erzeugt haben. Die Höhe und Breite des Bildes muss aus der Textdatei ermittelt werden. Die Parameter `nullZeichen` und `einsZeichen` geben an, welche Zeichen in die binären Werte 0 und 1 umgewandelt werden.

HINWEIS: Unter `http://www.cplusplus.com/reference/string/string/getline/` wird eine Methode beschrieben, welche Ihnen das Einlesen einer ganzen Zeile ermöglicht. Sie müssen die Erkennung des Zeilenendes also nicht selbst programmieren. In dem Beispiel wird ein Einlesen über `cin` beschrieben. Sie können hier aber auch `ifstream` verwenden.

_____ (4)

c) Fügen Sie der Klasse zwei Methoden `bool getPixel(unsigned int x, unsigned int y) const` und `void setPixel(bool wert, unsigned int x, unsigned int y)` hinzu. Diese dienen zum Auslesen und Setzen von einzelnen Pixeln. Überlegen Sie sich dazu, wie Sie die zweidimensionalen Koordinaten `x` und `y` in den richtigen Index im eindimensionalen Vektor `bildDaten` umrechnen. Die Koordinaten (0, 0) sollen den Wert an Index 0 adressieren.

_____ (2)

d) Implementieren Sie eine Methode `void save(string const &dateiName)`, die die Grafik als Textdatei im PBM-Format speichert (siehe `https://de.wikipedia.org/wiki/Portable_Anymap#Dateiformat`; beachten Sie aber, dass wir uns auf die textbasierte Variante für Schwarzweißbilder beschränken). Das PBM-Dateiformat ist wie folgt definiert:

- Die Datei hat die Endung `.pbm`.
- Die erste Zeile enthält nur den Text `P1`. Dieser Wert gibt den Dateityp an.
- Die zweite Zeile enthält optional einen mit # eingeleiteten Kommentar. Tragen Sie hier `# EidP Productions 2018/19` ein.
- In der nächsten Zeile stehen zwei durch Leerzeichen getrennte Zahlen, die die Anzahl der Spalten und Zeilen der Bilddaten angeben.

- Der Rest der Datei besteht aus den Bilddaten der Grafik. Dabei enhält jede Zeile so viele durch Leerzeichen getrennte Pixelwerte wie in der dritten Zeile der Datei angegeben.

_____ (4)

e) Entwerfen Sie zudem eine Ausnahmeklasse `PBMException`, welche eine Fehlerbeschreibung enthält. Diese Fehlerbeschreibung soll über die Methode `string getMessage() const` abgefragt werden können. Eine `PBMException` soll geworfen werden, falls

- eine Datei nicht geöffnet werden konnte,
- Zeichen außer dem `nullZeichen` und dem `einsZeichen` in der PBM-Datei vorkommen,
- die Anzahl Zeichen einer nichtleeren Zeile der Textdatei von der Anzahl Zeichen der ersten Zeile abweicht, oder
- in den Methoden `getPixel` oder `setPixel` Bereichsgrenzen überschritten werden.

Für jeden Fehlerfall soll eine eigene Fehlermeldung generiert und der `PBMException` übergeben werden.

_____ (4)

f) Testen Sie Ihre Implementierung, indem Sie die in Teil a) gespeicherte Datei in ein `PortableBitmap`-Objekt einlesen und als PBM-Datei wieder speichern.

_____ (3)