

Übungsaufgabe 2

Übung IT/EP 188.156

Bernhard Fiser
0109815

Beschreibung

Das Programm 'Rechnungserstellung' generiert Rechnungen, anhand einer Artikelstammdatenliste und einer 'Einkaufsliste'. In den Stammdaten sind die Artikelnummern, Bezeichnungen und Preise der Produkte erfasst, auf den Einkaufslisten scheint die Artikelnummer und gekaufte Menge auf.

Das Problem wurde gelöst, in dem die Artikel zeilenweise gelesen, geparkt, geprüft und in ein Objekt der Klasse 'Artikel' aufgenommen werden. Die Artikel werden danach in die erweiterte Klasse 'ArtikelListe' aufgenommen, die von der LinkedList abgeleitet wurde. Diese Klasse stellt zwei weitere Methoden zur Verfügung, eine zum Auslesen eines bestimmten Artikels, die zweite zum Prüfen, ob ein bestimmter Artikel bereits in der Liste vorhanden ist oder nicht.

Die Rechnungszeilen werden einzeln in Objekte der Klasse 'RechnungsElement' aufgenommen, diese dann wiederum in einer 'LinkedList' verkettet.

Abschließend wird die Liste mit den Rechnungselementen iteriert und pro Element in der ArtikelListe nachgeschaut und danach der Preis errechnet. Die Prüfung, ob ein Rechnungselement überhaupt in der ArtikelListe vorhanden ist, erfolgt gleich beim Einlesen der Rechnungszeilen.

Die Klassen 'RechnungsElement' und 'Artikel' verfügen über die Methoden 'parseArtikel' bzw. 'parseRechnungselement' die dazu da sind, um die Textzeile der Artikeldatei bzw. der Einkaufslisten in entsprechende numerische oder String Variablen zu parsen. Im Fehlerfall, d. h. bei Auftreten eines Syntaxerrors wird 'false' zurückgeliefert, andernfalls 'true'. Die Behandlung des Fehlers muß danach das Aufrufende Programm übernehmen.

Pseudocode

- Neue ArtikelListe erstellen
- Artikeldatei öffnen
- Schleife Zeile lesen
- {
 - Zeilenzähler erhöhen
 - neues Artikelobjekt erstellen
 - Zeile parsen und Werte in Objekt speichern
 - -> Bei Fehler Zeilennummer und Fehler ausgeben und Proggie beenden.
 - Objekt in ArtikelListe aufnehmen
- }
- Artikeldatei schließen
- Neue LinkedList erstellen
- Einkauflistendatei öffnen
- Schleife Zeile lesen
- {
 - Zeilenzähler erhöhen
 - neues Rechnungselement erstellen
 - Zeile parsen und Werte in Objekt speichern
 - -> Bei Fehler Zeilennummer und Fehler ausgeben und Proggie beenden.
 - Prüfen ob Rechnungselement ein Artikel aus der ArtikelListe ist.
 - -> Bei Fehler Zeilennummer und Fehler ausgeben und Proggie beenden.
 - Objekt in LinkedList aufnehmen.
- }
- Datei schließen
- Prüfen ob mehr als 10 Zeilen enthalten sind
- -> Bei Fehler Fehler ausgeben und Proggie beenden.
- Listiterator der RechnungsElementenliste erstellen
- Titelzeile der Rechnung ausgeben
- Schleife Element einzeln aus Liste herausnehmen
- {
 - In Artikelliste nachschauen und Preis berechnen.
 - Artikel mit Preis, Menge, Bezeichnung, ... ausgeben
- }
- Gesamtsumme ausgeben

Sourcecode Artikel.java

```
import java.util.*;

/**
 * Klasse für einzelnen Artikel
 *
 * @author Bernhard Fiser 0109815
 * @version 1.0
 */
public class Artikel
{
    public int    nummer;
    public String bezeichnung;
    public int    preis;

    /** Constructor, Artikel auf 0 initialisieren */
    public Artikel()
    {
        nummer = 0;
        bezeichnung = "";
        preis = 0;
    }

    /** Artikel aus einem String parsen und Werte setzen */
    public boolean parseArtikel(String s)
    {
        StringTokenizer      st;
        int                  i;
        String               t;
        double               d;

        /* StringTokenizer initialisieren */
        i = 0;
        st = new StringTokenizer(s, ",");
        while (st.hasMoreTokens())
        {
            t = st.nextToken();
            switch (i)
            {
                /* Ersten Token in Variable für Artikelnummer schreiben
                */
                case 0 :
                    try { this.nummer = Integer.parseInt(t); }
                    catch (NumberFormatException e) { return(false); }

                    if ((this.nummer < 0) || (this.nummer >= 1000))
                        return(false);
                    break;
                /* Zweiten Token in Variable für Bezeichnung schreiben
                */
                case 1 :
                    bezeichnung = t;
                    break;
                /* Dritten Token in Variable für Preis schreiben */
                case 2 :
                    try { d = Double.parseDouble(t); }
                    catch (NumberFormatException e) { return(false); }

                    if ((d < 0) || (d >= 1000))
                        return(false);
                    /* Preis auf Integer konvertieren um
                    Rundungsfehler auszuschließen */
                    this.preis = (int) (d * 1000.0);
                    break;
                /* Falls mehr Tokens, dann Fehler zurückgeben */
                default :
                    return(false);
            }
            i++;
        }
        /* Falls zu wenig Tokens, Fehler zurückgeben */
        if (i < 3)
            return(false);
        /* Andernfalls alles OK */
        return(true);
    }
}
```

Sourcecode ArtikelListe.java

```
import java.util.*;

/**
 * Klasse zum speichern aller Artikel
 * der Stammdaten. Die Klasse ist ein Child der
 * LinkedList.
 *
 * @author Bernhard Fiser 0109815
 * @version 1.0
 */

public class ArtikelListe extends LinkedList
{
    /** Constructor, LinkedList anlegen */
    public ArtikelListe()
    {
        super();
    }

    /** Methode um bestimmten Artikel aus Liste auszuwählen
     * anhand der Artikelnummer.
     * @param int nummer: Artikelnummer
     * @return Artikel
     */
    public Artikel getArtikel(int nummer)
    {
        ListIterator li;
        Artikel a;

        /* ListIterator initialisieren */
        li = this.listIterator(0);
        a = null;
        /* Elemente der Liste einzeln durchgehen */
        while (li.hasNext())
        {
            a = (Artikel) li.next();
            /* Falls Artikel gefunden, Schleife abbrechen */
            if (a.nummer == nummer)
                break;
        }
        /* Gefundenen Artikel zurückgeben (oder null, falls nicht) */
        return(a);
    }

    /** Methode um zu prüfen, ob spezieller Artikel in der Liste vorhanden ist */
    public boolean hasArtikel(int nummer)
    {
        ListIterator li;
        Artikel a;

        /* ListIterator initialisieren */
        li = this.listIterator(0);
        /* Alle Elemente einzeln durchgehen */
        while (li.hasNext())
        {
            a = (Artikel) li.next();
            /* Falls Element gefunden, true zurückgeben */
            if (a.nummer == nummer)
                return(true);
        }
        /* falls nicht gefunden, dann false zurückgeben */
        return(false);
    }
}
```

Sourcecode RechnungsElement.java

```
import java.util.*;

/** Klasse für einzelnes Rechnungselement
 * @author Bernhard Fiser 0109815
 */
public class RechnungsElement
{
    public int    nummer;
    public int    menge;

    /** Constructor, initialisiert das Element auf 0 */
    public RechnungsElement()
    {
        nummer = 0;
        menge = 0;
    }

    /** aus String ein Rechnungselement parsen */
    public boolean parseRechnungsElement(String s)
    {
        StringTokenizer st;
        int i;
        String t;

        /* StringTokenizer initialisieren */
        i = 0;
        st = new StringTokenizer(s, ",");
        while (st.hasMoreTokens())
        {
            t = st.nextToken();
            switch (i)
            {
                /* Ersten Token in die Variable für Artikelnummer
                schreiben */
                case 0 :
                    try { this.nummer = Integer.parseInt(t); }
                    catch (NumberFormatException e) { return(false); }
                    break;
                /* Zweiten Token in die Variable für die Menge schreiben
                */
                case 1 :
                    try { this.menge = Integer.parseInt(t); }
                    catch (NumberFormatException e) { return(false); }
                    break;
                /* Falls mehr Tokens in Zeile, dann Fehler zurückgeben
                */
                default :
                    return(false);
            }
            i++;
        }
        /* Falls zu wenig Tokens, dann Fehler zurückgeben */
        if (i < 2)
            return(false);
        /* Andernfalls alles ok */
        return(true);
    }
}
```

Sourcecode Rechnungserstellung.java (Hauptprogramm)

```
import java.util.*;

/**
 * Programm zur Rechnungserstellung
 *
 * @author Bernhard Fiser 0109815
 * @version 1.0
 */

public class Rechnungserstellung
{
    static ArtikelListe al;
    static Artikel a;
    static ReadFile fa;
    static ReadFile fr;
    static int lineno;
    static String line;
    static RechnungsElement re;
    static LinkedList li;
    static ListIterator lit;
    static double total;
    static double sp;
    static double gttotal;

    /* Hauptprogramm */
    public static void main(String argv[])
    {
        /* Prüfen ob genügend Parameter übergeben wurden */
        if (argv.length < 2)
        {
            System.err.println("*** usage: java Rechnungserstellung  
<artikelliste> <rechnung>");
            return;
        }

        /* Neue Artikelliste anlegen und Artikelfile öffnen */
        al = new ArtikelListe();
        fa = new ReadFile(".", argv[0]);
        lineno = 0;
        /* Zeilen einzeln einlesen */
        while ((line = fa.readLine()) != null)
        {
            lineno++;
            /* neuen Artikel anlegen */
            a = new Artikel();
            /* Eingelesene Zeile parsen und gegebenenfalls Fehler ausgeben
            */
            if (!a.parseArtikel(line))
            {
                System.err.println("*** " + argv[0] + ": syntax error in  
line " + lineno);
                return;
            }
            /* Falls alles ok, dann zur Artikelliste hinzufügen */
            al.add(a);
        }
        /* Artikelfile schließen */
        fa.close();

        /* Rechnungsfile öffnen und LinkedList anlegen für einzelne  
Rechnungselemente */
        fr = new ReadFile(".", argv[1]);
        li = new LinkedList();
        lineno = 0;
        /* Rechnungszeilen einzeln einlesen */
        while ((line = fr.readLine()) != null)
        {
            lineno++;
            /* Rechnungselement erstellen, und eingelesene Zeile parsen.
            * Falls Fehler beim parsen, Fehler ausgeben und Programm  
beenden */
            re = new RechnungsElement();
            if (!re.parseRechnungsElement(line))
            {
                System.err.println("*** " + argv[1] + ": syntax error in  
line " + lineno);
                return;
            }
            /* Prüfen ob Artikel in Artikelstammdaten existiert.
            * Falls nicht, Fehler ausgeben und Programm beenden */
            if (!al.hasArtikel(re.nummer))
            {
                System.err.println("*** " + argv[1] + ": artikel " +  
re.nummer + " not found");
                return;
            }
            /* Wenn alles OK, dann Rechnungszeile zu Liste hinzufügen */
        }
    }
}
```

```
        li.add(re);
    }
    /* Rechnungsfile schließen */
    fr.close();
    /* Falls mehr als 10 Zeilen in Rechnungsfile, Fehler ausgeben und
    * Programm beenden */
    if (lineno > 10)
    {
        System.err.println("*** " + argv[1] + ": too much entries (>
10)");
        return;
    }

    /* Variablen für Rechnungsausgabe initialisieren
    * gtotal: Rechnungsgesamtsumme */
    gtotal = 0;
    /* Listiterator */
    lit = li.listIterator(0);
    /* Artikel aus Stammdaten */
    a = new Artikel();
    /* Rechnungselement */
    re = new RechnungsElement();
    /* Rechnungskopfzeile ausgeben */
    System.out.println("ARTNR\tBEZ.\tPREIS/STK\tMENGE\tPREIS");
    /* Elemente der Liste einzeln durchgehen */
    while (lit.hasNext())
    {
        re = (RechnungsElement) lit.next();
        /* Artikel in Stammdaten suchen */
        a = al.getArtikel(re.nummer);
        /* Preis berechnen */
        total = re.menge * a.preis / 1000;
        sp = a.preis / 1000;
        /* Gesamtpreis erhöhen */
        gtotal += total;
        /* Rechnungszeile ausgeben */
        System.out.println(a.nummer + "\t" + a.bezeichnung + "\t" + sp +
"\t" + re.menge + "\t" + total);
    }
    /* Am Ende Gesamtsumme ausgeben */
    System.out.println("GESAMTSUMME: " + gtotal);
}
```