Dr. Andreas Borchert
Ingo Melzer

Sektion Angewandte Informationsverarbeitung

9. Juni 1999
Blatt 7

Allgemeine Informatik II (SS 1999)

Abgabetermin 24. Juni 1999

8 Recycling (10 Punkte)

Beim letzten Blatt wurde eine simple Liste implementiert, die allerdings nicht für jedes Problem, das Listen benötigt, einfach wieder genutzt werden kann. Verallgemeinern Sie daher bitte Ihre Lösung[1] vom letzten Blatt, damit dieses Manko beseitigt wird.

Jedes Listenelement sollte eine Variable vom Typ *Objects.Object* aufnehmen können. Eine brauchbare Datenstruktur kann wieder auf Seite 112 im Skript als Grundlage gefunden werden, wobei es sich empfiehlt, *ListRec* als Erweiterung von Objects.ObjectRec zu definieren. Werfen Sie hierfür auch einen Blick auf Seite 165 im Skript.

Beachten Sie, daß dieses Modul für sich alleine noch nichts tut. Schreiben Sie daher ein Modul *TestIntLists*, das Ihr eben geschriebenes Listenmodul verwendet und lösen Sie damit die Aufgaben des letzten Blattes. Die Prozedur *Merge* ist im Modul *TestIntLists* zu implementieren. Sie dürfen gerne den Rumpf nutzen, den Sie auf dem FTP-Server finden. Bereichern Sie zusätzlich noch Ihre Listen um die Prozedur *InsertN*, die ein Element an der *N*-ten Position einfügt. Diese wird Ihnen bei der nächsten Aufgabe unter Umständen sehr nützlich sein, da dadurch präziseres Einfügen möglich wird.

9 Time Recycling (10 Punkte)

Um die Flexibilität Ihrer Listen zu zeigen, machen Sie sich nun an Listen für zeitliche Ereignisse. Dazu schreiben Sie ein Modul *TimeLists*, das als private Komponente im Listenrecord eine ihrer Listen enthält. Da diese Ereignisse in chronologischer Reihenfolge in der Liste sein sollten, müssen Sie zum Einfügen eine neue Prozedur *Insert* schreiben (dafür werden *AddFirst* und *Append* nicht mehr benötigt). Alle anderen im Definition Modul geforderten Prozeduren können direkt durch Aufrufe Ihres obigen Listenmoduls erledigt werden, das heißt, daß die neuen Prozeduren jeweils nur noch aus einer Zeile mit dem Aufruf der entsprechenden Prozedur des Listenmoduls bestehen.

Da auch dieses Modul getestet werden soll, ändern Sie den *CalendarManager* aus der Vorlesung so ab, daß er Ihre *TimeLists* verwendet. Dabei können im *CalendarManager* die oberen 70 Prozent fast ganz gelöscht werden.

[1] Oder die Beispiellösung

Noch ein paar Tips:

- Machen Sie sich zuerst Gedanken zu diesem Blättchen. VHIT ist bei Zeigern am Anfang keine gute Idee. Zeichnen Sie eine Skizze, die verdeutlicht, wie und auf was die Zeiger wann zeigen sollten.
- Verwenden Sie einen Listenelementrecord.
- Beachten Sie besonders beim Löschen die Ränder der Liste. Prüfen Sie auf entsprechende *NIL*–Zeiger.
- Sie dürfen gerne den Listenrecord (wenn es Ihnen hilft) um eigene Komponenten anreichern.
- Sie können gerne doppelt verkettete Listen verwenden. Dies ist eine gute Übung, es ist aber nicht dringend notwendig.
- Manchmal können sinnvolle Assertions bei der frühen Fehlersuche helfen und inkonsistente Zustände verhindern.
- Beachten Sie bei der zweiten Aufgabe bitte, daß das neue Modul *TimeLists* die privaten Komponenten Ihres Listenmoduls nicht sieht. Es ist dort also nicht möglich direkt auf dessen Zeiger zuzugreifen; stattdessen müssen die entsprechenden Listenprozeduren, die im Definition Modul exportiert werden, aufgerufen werden.

Für alle Module dieses Blattes sind die Definition Module, die Sie auf dem FTP-Server finden, zu verwenden. Diese dürfen nicht verändert werden.