Dr. Andreas Borchert
Ingo Melzer Sektion Angewandte Informationsverarbeitung 26. Mai 1999
Blatt 6

Allgemeine Informatik II (SS 1999)

Abgabetermin 10. Juni 1999

7 Verzeigert (20 Punkte)

Nur zur Übung soll dieses Mal eine simple Liste implementiert werden.

- Jedes Listenelement sollte eine Zahl vom Typ *INTEGER* aufnehmen können. Überlegen Sie sich als erstes eine brauchbare Datenstruktur für Ihre Liste (Seite 112 im Skript ist schon fast brauchbar). Schreiben Sie dann eine Prozedur *Create*, die eine Liste initialisiert, und eine Prozedur *AddFirst*, die eine Zahl als Listenelement an den Anfang einer Liste anfügt. Als Test sollten Sie die Zahlen von null bis neun in Ihre Liste stopfen und dann die Liste wieder ausgeben. Achten Sie darauf, daß Sie den Anfang der Liste nicht verlieren. Damit haben Sie sich schon einmal vier Punkte verdient.

- Um die Liste später flexibler einsetzen zu können, schreiben Sie als nächstes eine Prozedur *Append*[1], die ein Element an das Ende der Liste anhängt. Um einige Abfragen zu erleichtern, implementieren Sie zusätzlich noch die Funktion *Length*, die (Überraschung) die Anzahl der Elemente in der Liste liefert. Zum Testen lesen Sie bitte einige Zahlen in zwei Listen ein, wobei die Zahlen jeder dieser Listen sortiert sein sollten. Schreiben Sie nun eine Prozedur *Merge*, welche die Zahlen der beiden sortierten Listen in eine neue sortierte Liste stopft.

- Da bei einfach verketteten Listen das Navigieren in der Liste sehr mühsam sein kann und man auch immer den Anfang aufheben muß, beheben Sie diese Nachteile (falls Sie es nicht eh schon getan haben), indem Sie eine doppelte Verkettung einfügen. Dabei hat jedes Listenelement einen Zeiger auf seinen Vorgänger und seinen Nachfolger. Am Listenanfang und Ende sind die entsprechenden Zeiger natürlich *NIL*.

- Bis jetzt handelt es sich bei Ihrer Liste um ein wunderbares Datengrab, da brauchbare Möglichkeiten zum Lesen von Elementen der Liste noch fehlen. Schreiben Sie daher die Prozeduren *GetFirst*, *GetLast* und *GetN*, wobei die letzte das n-te Element der Liste besorgt.

- Da es bei vielen Anwendungen wünschenswert ist, unbenötigte Elemente zu entsorgen, schreiben Sie die drei Prozeduren *RemoveFirst*, *RemoveLast* und *RemoveN*. Diese sollten

[1] Für FIFO notwendig.

sich analog zu den Prozeduren des letzten Punktes verhalten. Stellen Sie sicher, daß N einen gültigen Wert enthält. Schreiben Sie sich eine neue Ausgabeprozedur, die eine Liste mit *GetFirst* und *RemoveFirst* ausgibt und dabei löscht. Die notwendige Schleife sollte laufen bis *Length* null liefert.

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.
- Die einzelnen Punkte sind zwar der Schwierigkeit nach sortiert, wenn Sie aber Probleme mit einem Punkt haben sollten, können Sie sich auch gerne zuerst mit dem nächsten versuchen. Jeder Teil wird vier Punkte geben.
- 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.