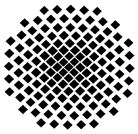
Universität Stuttgart
Fakultät 5, Institut FMI
EfidI 1 Zusatzkurs



WS 2005/2006
Wolfgang Schmid
24. November 2005

Aufgabenblatt 5

Besprechung am Dienstag, den 6. Dezember 2005, 8:00 Uhr V38.01

Aufgabe 1 Einfach verkettete Liste

Es ist eine einfach verkettete Liste zu implementieren mit den Funktionen:

- Einfügen am Anfang
- Einfügen am Ende
- Einfügen vor einem gegebenen Element
- Einfügen hinter einem gegebenen Element
- Suche nach einem bestimmten Element
- Löschen eines bestimmten Elements
- Sortieren der Liste
- Umkehren der Liste

Wie groß ist der Aufwand für Einfügen, Suchen und Löschen?

Welche Veränderungen ergeben sich bei einer doppelt verketteten Liste?

Entwerfen Sie eine Datenstruktur für einen Graph und einen Baum als verkettete Liste.

Aufgabe 2 Listenumordnung

Schreiben Sie eine Prozedur, welche eine gegebene Liste umordnet, so dass alle Knoten mit geradem Wert *val* am Anfang der Liste stehen. Die Ordnung innerhalb der geraden Knoten spielt keine Rolle.

Die Prozedur soll die folgende Schnittstelle haben:

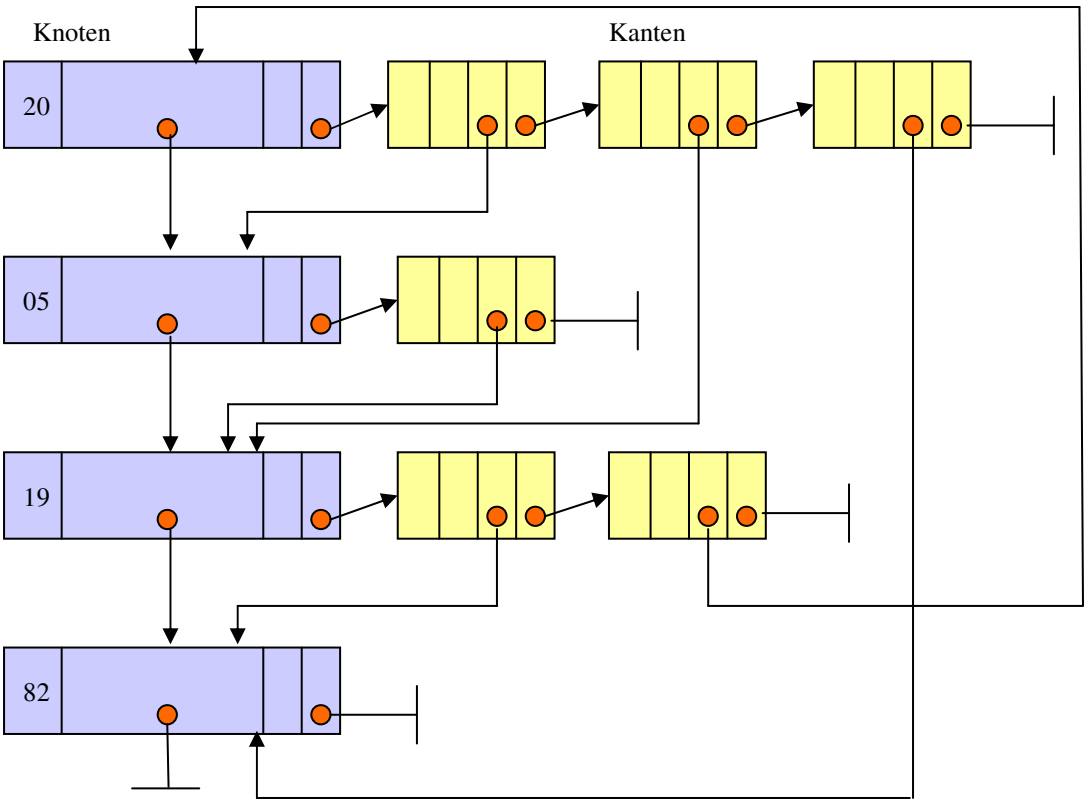
```
type  PNodeDesc;


type   NodeDesc is record
   val: integer;
   next: PNodeDesc;
end record;


type  PNodeDesc  is access NodeDesc;
```

Aufgabe 3 Adjazenzlisten

Wandeln Sie einen Graphen, der als Adjazenzmatrix gegeben ist, in eine Adjazenzliste um.



Beschreiben Sie Tiefensuche und Breitensuche in einem Graphen.

Aufgabe 4 Adressverwaltung

Schreiben Sie ein Programm in einer beliebigen Programmiersprache, das eine Liste von Namen, Adressen und Telefonnummern verwaltet. Schließen Sie ein Menü ein, aus dem der Benutzer eine der folgenden Tätigkeiten auswählen kann:

- 1: Hinzufügen eines neuen Satzes
- 2: Anzeigen eines Satzes zu einem gegebenen Namen
- 3: Sortieren der Daten nach dem Nachnamen mittels Selectionsort
- 4: Löschen eines Datensatzes zu einem gegebenen Namen
- 9: Programmende

Was ist eine linear verkettete, was eine kreisförmig verkettete und was eine doppelt verkettete Liste?

Welche Art von Vergleichen kann zwischen Zeigervariablen ausgeführt werden? Welche Operationen können dazu benutzt werden?

Aufgabe 5 Monopoly

Im Monopoly-Spiel gibt es unterschiedliche Arten von Feldern auf dem Spielbrett. Das sind unter anderem:

- *Straßen* sind käuflich, können bebaut und beliehen werden. Der Mietpreis für Gegner wird von der Anzahl der Häuser oder Hotels bestimmt, die der Besitzer auf der Straße errichtet hat.
- *Bahnhöfe* sind ebenfalls käuflich und können beliehen werden. Man kann aber nicht darauf bauen. Der Mietpreis hängt davon ab, wie viele Bahnhöfe der Besitzer insgesamt hat.
- *Werke* sind käuflich und können beliehen werden. Wie Bahnhöfe kann man Werke nicht bebauen. Der Mietpreis ist das Produkt aus der Augenzahl des Wurfes, mit dem ein Gegner das Werk erreicht hat, sowie einem Faktor, der von der Anzahl Werke des Besitzers abhängt.

In einem Spielprogramm soll das Spielbrett mit all seinen Feldern modelliert werden. Der Aufbau und die Eigenschaften von Objekten eines vorgegebenen Problembereichs sind in Grundzügen gleich, unterscheiden sich aber in Einzelheiten.

Im obigen Beispiel lassen sich die folgenden gemeinsamen Attribute dieser drei Arten von Spielfeldern identifizieren:

- ein Name,
- ein Kaufpreis,
- ein Hypothekenwert,
- ein möglicher Besitzer.

Darüber hinaus lassen sich einzelnen Typen die folgenden speziellen Eigenschaften zusprechen:

- Straßen: Bebauung, Farbe, Mietstaffel nach Anzahl Immobilien,
- Bahnhöfe: Mietstaffel nach Anzahl Bahnhöfe,
- Werke: Faktor nach Anzahl Werke.

Die Eigenschaften sind z.T. statisch (d.h. unveränderlich), wie z.B. Name, Kaufpreis und Farbe einer Straße. Die anderen Eigenschaften sind dynamisch (d.h. veränderlich), wie z.B. Bebauung und Besitzer.
Entwerfen Sie einen Record-Typ, um diese Monopolyfelder zu modellieren. Setzen Sie variante Records ein, um zu einer kompakten Struktur zu kommen.

Aufgabe 6 Literaturliste

Für eine Literaturdatenbank soll eine Datenstruktur zur Erfassung von Dokumenten entworfen werden. An Dokumentenarten werden vorläufig

- Bücher,
- Zeitschriftenhefte und
- elektronische Dokumente

erwartet. Ein **Buch** soll mit den folgenden Informationen gespeichert werden:

- Titel (`String`)
- bis zu 3 Autoren (jeweils `String`)
- Verlag (`String`)
- Erscheinungsjahr (`1900..2099`)
- Auflage (`1..99`)
- Umfang (`integer`)
- Sprache (`deutsch, englisch, französisch, russisch, andere`)
- ISBN-Nummer (`String`)

Ein **Zeitschriftenheft** wird beschrieben durch:

- Titel (`String`)
- Verlag (`String`)
- Jahrgang (`1900..2099`)
- Heftnummer (`1..99`)
- Sprache (`deutsch, englisch, französisch, russisch, andere`)
- ISSN-Nummer (`String`)

Ein **elektronisches Dokument** wird beschrieben durch:

- Titel (`String`)
- bis zu 3 Autoren (jeweils `String`)
- Dateiname (`String`)
- Archiv (`String`)
- Version (`String`)
- Länge (`integer`)
- Erstellungsjahr (`1900..2099`)
- Sprache (`deutsch, englisch, französisch, russisch, andere`)
- Format (`ascii, tex, postscript, winword, framemaker, andere`)

Entwerfen Sie einen Record-Typ `LitT`, sowie passende Hilfstypen, um Literatureinträge in der Datenbank zu modellieren. Setzen Sie variante Records ein, um zu einer kompakten Struktur zu kommen.

Schreiben Sie drei passende Funktionen bzw. Prozeduren `Book`, `Periodical` und `Edoc`, mit denen ein Buch, ein Zeitschiftenheft bzw. ein elektronisches Dokument erfaßt werden können. Jede dieser Funktionen bzw. Prozeduren wird einen Record vom Typ `LitT` als reference-Parameter bekommen, sowie die oben beschriebenen Einzelinformationen, um den Record damit korrekt auffüllen.

Schreiben Sie auch eine Prozedur `PrintLit`, die einen Record vom Typ `LitT` ausgibt.

Allgemeine Hinweise:

- Bei weiteren Fragen, wenden Sie sich bitte an W. Schmid (sltsoftware@yahoo.de).

Weitere Hinweise finden Sie auf unserer Veranstaltungswebseite unter:
http://www.info2.de.vu
http://www.zusatzkurs.de.vu