Dipl.Phys. Gerald Kempfer

Berliner Hochschule für Technik
- University of Applied Sciences -
Fachbereich VI – Informatik und Medien
Studiengang Technische Informatik Bachelor

Lehrveranstaltung "Algorithmen und Datenstrukturen"
Übungsblatt 7

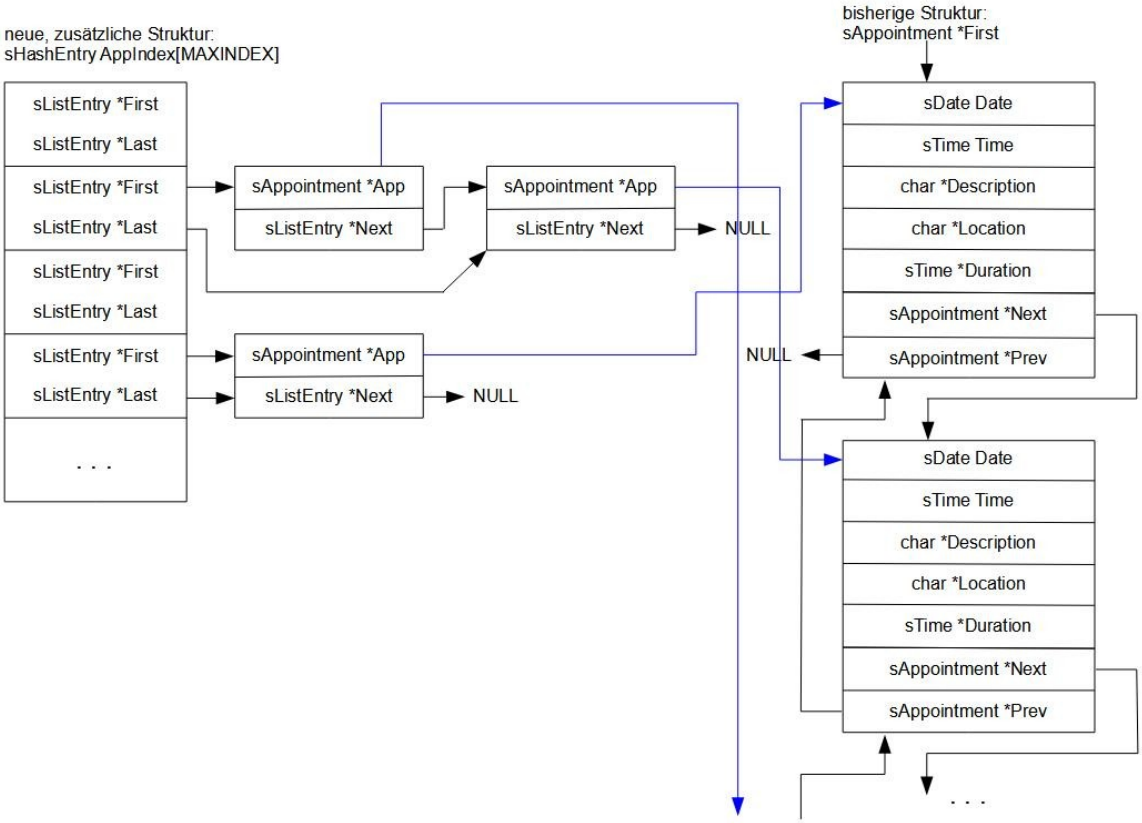
Hinweise:

Dieses Übungsblatt ist zur Zulassung zu der Klausur erfolgreich zu bearbeiten ("*Erfolgreich*" bedeutet: Keine Programmabstürze bzw. Endlosschleifen, Aufgabenstellung einschließlich der Nebenbedingungen müssen eingehalten sowie Kommentierung und Einrückung korrekt sein! Compilerwarnungen sollen möglichst vermieden werden.).

Die Aufgaben werden überwiegend in den Übungszeiten bearbeitet und dort auch abgegeben. Allerdings genügt die Zeit hierfür unter Umständen nicht, so dass Sie auch außerhalb dieser Zeiten die Aufgaben bearbeiten müssen. Der Abgabetermin für diese Aufgabe ist der **18. Januar 2024**.

Aufgabe: In der siebenten Übungsaufgabe des Projektes "Terminverwaltung" soll die Suche nach Terminbezeichnungen implementiert werden. Als Algorithmus soll eine berechnete Suche (Hashing) verwendet werden. Um die Kollisionen beim Hashing zu vermeiden, sollen Elemente mit gleichen Hashwert in einer einfach verketteten Liste gespeichert werden.

Die bestehenden Datenstrukturen der vorigen Aufgabe sollen unverändert bleiben, d.h. die Termine bleiben in einer doppelt verketteten Liste. Es kommen aber neue Datenstrukturen hinzu: Als Hash-Tabelle soll ein Array von Hash-Elementen namens `AppIndex` angelegt werden. Ein Hash-Element ist jeweils der Beginn einer einfach verketteten Liste, d.h. besteht aus einer Datenstruktur, die zwei Zeiger (`First` und `Last`) auf Listenelemente enthält. Ein Listenelement enthält einen Zeiger auf einen Termin und einen Zeiger auf das nächste Listenelement. Die Zeiger auf Termine zeigen auf die zugehörigen Datensätze in der bisherigen Datenstruktur. Alle neuen Datenstrukturen (also `sListEntry`, `sHashEntry` sowie der `extern`-Verweis auf das Array von `sHashEntry` namens `AppIndex`) werden in der `datastructure.h` gespeichert. Diese Struktur kann wie folgt skizziert werden:



Folgende Anpassungen müssen für diese Übung durchgeführt werden:

`datastructure.h`:
Es wird eine neue Konstante für das Array `AppIndex` benötigt:
`#define MAXINDEX 307`
Diese Konstante sollte auf eine Primzahl gesetzt werden (z.B. 307).
Es werden zwei neue Datenstrukturen benötigt:
`sListEntry`:
Beinhaltet einen Zeiger auf einen Termin (`sAppointment`) und einen `Next`-Zeiger auf die eigene Datenstruktur.
`sHashEntry`:
Beinhaltet einen `First`- und einen `Last`-Zeiger auf `sListEntry`.
Ferner wird eine Deklaration (das war die Sache mit dem `extern`) des Arrays `AppIndex` benötigt, wobei jedes Element dieses Arrays vom Typ `sHashEntry` ist.

`calendar.c`:
Hier wird das Array `AppIndex` mit `MAXINDEX` Elementen definiert. Dieses neue Array muss nun in folgenden Funktionen mit berücksichtigt werden:
- Löschen / Freigeben eines Termins (es muss das dazugehörige Listenelement aus `AppIndex` entfernt werden)
- Nach dem Erzeugen eines neuen Termins muss ein entsprechendes Listenelement im Array `AppIndex` eingefügt werden.
- Wie gut, dass die Sortierfunktion wegen der doppelt verketteten Liste nicht mehr in Betrieb ist, sonst hätten hier bei jeder Tauschaktion innerhalb der Sortierfunktion auch die Einträge in der Hashtabelle angepasst werden müssen.

- Die Funktion `searchAppointment` ruft die eigentliche Suche auf: Eingabe der zu suchenden Terminbezeichnung, Such-Funktion aufrufen, gefundenen Termin auf dem Bildschirm anzeigen bzw. Meldung ausgeben, dass der gesuchte Termin nicht gefunden wurde.

`database.c`:
- Nach dem erfolgreichen Einlesen eines Termins muss ein entsprechendes Listenelement im Array `AppIndex` eingefügt werden.

`list.c`:
In diesem Modul werden zwei neue Funktionen benötigt (anders als die bereits für die sechste Aufgabe erstellten Funktionen behandeln diese neuen Funktionen eine einfach verkettete Liste; daher das „S“ in den Funktionsnamen!):
- `appendInSList`:
Diese Funktion erhält einen Zeiger auf ein Element im Array `AppIndex` (nicht auf das erste Element, sondern auf das Element des errechneten Hashwertes!) und einen Zeiger auf den Termin. Als Ergebnis wird ein Wahrheitswert zurückgegeben, der angibt, ob das neue Listenelement erfolgreich angelegt werden konnte. In dieser Funktion wird ein neues Listenelement vom Typ `sListEntry` erzeugt und an die Liste im angegebenen Arrayelement vom Typ `sHashEntry` angehangen.
- `removeFromSList`:
Diese Funktion erhält einen Zeiger auf ein Element im Array `AppIndex` (siehe vorige Funktion) und einen Zeiger auf den Termin, dessen Listenelement aus der verketteten Liste gelöscht werden soll. Als Ergebnis wird ein Zeiger auf das aus der Liste entfernte Element zurückgegeben. In der Funktion wird in der verketteten Liste nach dem Element gesucht, dessen Termin-Zeiger auf den angegebenen Termin zeigt, und dieses Element dann aus der Liste entfernt.

`search.c`:
Dieses neue Modul soll zum Einen die Hash-Funktion (z.B. Divisionsrest) und zum Anderen die eigentliche Suchfunktion beinhalten:
- `calcDivisionsrest`:
Diese Funktion erhält eine Zeichenkette (Bezeichnung des Termins) und errechnet daraus den Hashwert.
- `search`:
Diese Funktion erhält einen Zeiger auf das Indexarray `AppIndex`, einen Zeiger auf eine Vergleichsfunktion zum Vergleichen von zwei Terminbezeichnungen (diese Vergleichsfunktion existiert ja bereits vom Sortieren) sowie einen Zeiger auf einen Termin, in dem die gesuchte Bezeichnung steht. In der Funktion muss der Hashwert errechnet werden, um dann in der entsprechenden verketteten Liste linear nach dem gewünschten Termin zu suchen. Wird der Termin gefunden, soll ein Zeiger auf das gefundene Listenelement (`sListEntry`) zurückgegeben werden; ansonsten ein `NULL`-Zeiger.

Zusätzlich soll eine Funktion zum Anzeigen der Hashtabelle (Array `AppIndex`) implementiert werden. Diese Funktion zeigt sehr schön, wo die einzelnen Werte in der Hashtabelle gespeichert werden. In der neuen vorgegebenen Datenbank `calendar.xml` gibt es gleich mehrere Kollisionen, d.h. Terminbezeichnungen mit dem gleichen Hashwert, die in dieser Auflistung anhand der gleichen Hashwerte gut zu sehen sind.

Kommentieren Sie das Programm. Dazu gehören auch Modul- und Funktionsheader (siehe Skript "Grundlagen der Informatik" Kapitel 5.3 und 5.4)! Achten Sie auch auf Ihre Programmstruktur (Einrückungen; Leerzeichen und Leerzeilen, usw.).

Beispielausgabe Suchfunktion:

```
Geben Sie bitte die Beschreibung des gesuchten Termins ein:
-> AuD - Abgabe 6. Aufgabe

Suchergebnis:
-------------

================================================================================
Do, 04.01.2024:
---------------
   17:45 - 19:15 -> Raum D 114      | AuD - Abgabe 6. Aufgabe


Bitte Eingabetaste druecken ...
```

Beispielausgabe Hashtabelle:

```
Hashtabelle
===========

Hashwert | Datum          | Uhrzeit | Terminbeschreibung
---------|----------------|---------|---------------------------------------------------
       1 | Do, 14.12.2023 |  16:00  | AuD Seminaristischer Unterricht
         | Do, 30.11.2023 |  16:00  | AuD Seminaristischer Unterricht
      19 | Sa, 16.12.2023 |  06:00  | Wecker ausschalten und weiterschlafen
      26 | Mi, 06.12.2023 |  04:00  | dem Nikolaus auflauern
      95 | Do, 30.11.2023 |  17:45  | AuD Uebungsgruppe 2 - Vierte Uebungsaufgabe abgeben
     101 | Do, 30.11.2023 |  19:30  | AuD Nachsitzen
     103 | Do, 14.12.2023 |  18:50  | Sandmaennchen
     112 | Fr, 01.12.2023 |  05:30  | Erstes Tuerchen oeffnen
     156 | So, 17.12.2023 |  23:00  | kleines Nachtmahl
     204 | So, 17.12.2023 |  10:00  | ausgiebiges Fruehstueck
     205 | Do, 21.12.2023 |  17:45  | AuD - Letzte Uebung in diesem Jahr
     212 | So, 17.12.2023 |  16:00  | ueppiges Mittagessen
     231 | Sa, 23.12.2023 |  16:00  | noch schnell ein paar Geschenke kaufen
     233 | Do, 04.01.2024 |  17:45  | AuD - Abgabe 6. Aufgabe
         | Do, 30.11.2023 |  15:10  | zur Hochschule fahren
     247 | Do, 25.01.2024 |  16:00  | Klausur Algorithmen und Datenstrukturen
     271 | So, 17.12.2023 |  20:00  | reichhaltiges Abendessen
     281 | Do, 04.01.2024 |  16:00  | AuD - endlich wieder Seminaristischer Unterricht
     306 | So, 31.12.2023 |  20:00  | Silvester-Party

Bitte Eingabetaste druecken ...
```