
Lehrveranstaltung "Algorithmen und Datenstrukturen" **Übungsblatt 7**

Hinweise:

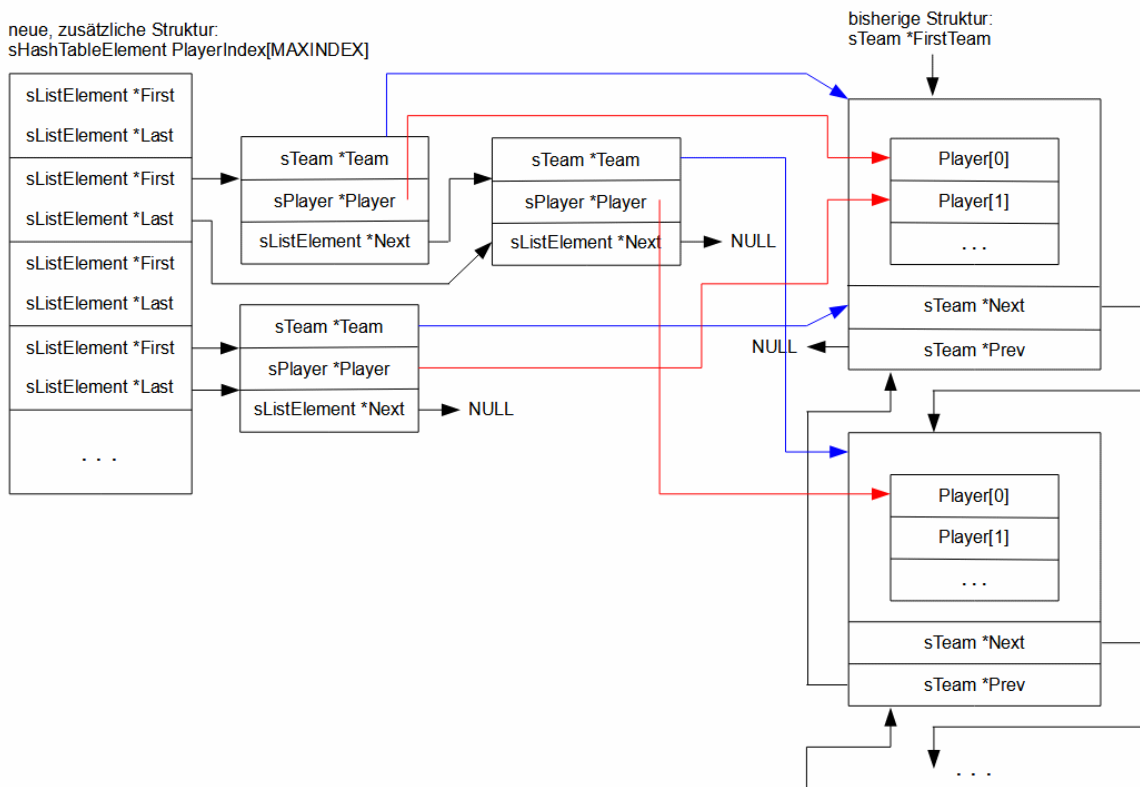
Dieses Übungsblatt ist zur Zulassung zu der Klausur erfolgreich zu bearbeiten ("*Erfolgreich*" bedeutet: Keine Programmabstürze bzw. Endlosschleifen, Aufgabenstellung einschließlich der Nebenbedingungen müssen eingehalten sowie Kommentierung und Einrückung korrekt sein!).

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 **17. Januar 2025 (Gruppe 2a/1a) bzw. 24. Januar 2025 (Gruppe 2b/1b)**.

Aufgabe:

In der siebten und letzten Übungsaufgabe des Projektes "Mannschafts-Verwaltung" soll die Suche nach Spielern implementiert werden. Als Algorithmus soll eine berechnete Suche (Hashing) verwendet werden. Um die Kollisionen beim Hashing zu vermeiden, sollen Elemente mit gleichem Hashwert in einer einfach verketteten Liste gespeichert werden.

Die bestehenden Datenstrukturen sollen unverändert bleiben, d.h. die Mannschaften bleiben in einer doppelt verketteten Liste und die Spieler bleiben in einem Array; damit bleiben auch alle bisherigen Funktionen erhalten. Es kommen aber neue Datenstrukturen hinzu: Als Hash-Tabelle soll ein Array von Hash-Elementen angelegt werden. Ein Hash-Element ist jeweils der Beginn einer einfach verketteten Liste, d.h. besteht aus einer Datenstruktur, die jeweils zwei Zeiger (First und Last) auf Listenelemente enthält. Ein Listenelement enthält einen Zeiger auf eine Mannschaft (mit Hilfe dieses Zeigers kann der Name der Mannschaft des gesuchten Spielers ausgegeben werden), einen Zeiger auf einen Spieler sowie einen Zeiger auf das nächste Listenelement. Die Zeiger auf Mannschaft und Spieler zeigen jeweils auf die zugehörigen Datensätze in der bisherigen Datenstruktur. Das ganze kann wie folgt skizziert werden:



Folgende Anpassungen müssen für diese Übung durchgeführt werden:

In der Datei `datastructure.h`:

Es wird eine neue Konstante für die Hash-Tabelle benötigt:

```
#define MAXINDEX 307
```

Diese Konstante sollte auf eine Primzahl gesetzt werden, damit die Datensätze in der Hash-Tabelle gut gestreut und damit die Kollisionen minimiert werden. Ferner werden zwei neue Datenstrukturen benötigt:

`sListElement`:

Beinhaltet einen Zeiger auf eine Mannschaft (`sTeam`), einen Zeiger auf einen Spieler (`sPlayer`) und einen `Next`-Zeiger auf die eigene Datenstruktur.

`sHashTableElement`:

Beinhaltet einen `First`- und einen `Last`-Zeiger auf `sListElement`.

Ferner wird eine Deklaration (das war die Sache mit dem `extern`) des Arrays `PlayerIndex` benötigt, wobei jedes Element dieses Arrays vom Typ `sHashTableElement` ist.

In der Datei `teams.c`:

Hier wird das Array `PlayerIndex` mit `MAXINDEX` Elementen global definiert. Dieses neue Array muss nun in folgenden Funktionen mit berücksichtigt werden:

- Erzeugen eines neuen Spielers: der Spieler muss in einem neuen Listenelement der Hash-Tabelle hinzugefügt werden,
- Löschen / Freigeben eines Spielers: es muss das dazugehörige Listenelement aus der Hash-Tabelle entfernt werden,
- Tauschen von zwei Spielern (wird für das Sortieren benötigt; das Sortieren soll ja auch weiterhin funktionieren!): auch die dazugehörigen Listenelemente in der Hash-Tabelle müssen getauscht werden,

- Suchen nach Spielern (neu): ruft die Suche nach einem Spieler auf. Folgender Ablauf wird empfohlen: Eingabe des zu suchenden Spielernamen, Such-Funktion aufrufen, gefundenen Spieler einschließlich der Mannschaft, zu der er gehört, auf dem Bildschirm anzeigen, bzw. eine Meldung ausgeben, dass der gesuchte Spieler nicht gefunden wurde.

In der Datei `database.c`:

Nach dem erfolgreichen Einlesen eines Spielers muss ein entsprechendes Listenelement im Array `PlayerIndex` eingefügt werden.

In der Datei `list.c`:

In diesem Modul werden zwei neue Funktionen benötigt (anders als die bereits für die vorige Aufgabe erstellten Funktionen behandeln diese neuen Funktionen eine einfach verkettete Liste!):

- `appendInEVList`:

Diese Funktion erhält einen Zeiger auf ein Element im Array `PlayerIndex` (nicht auf das erste Element, sondern auf das Element des errechneten Hashwertes!) und je einen Zeiger auf die Mannschaft und auf den Spieler. 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 `sListElement` erzeugt und an die Liste im angegebenen Arrayelement vom Typ `sHashTableElement` angehängen.

- `removeFromEVList`:

Diese Funktion erhält einen Zeiger auf ein Element im Array `PlayerIndex` (siehe vorige Funktion) und einen Zeiger auf den Spieler (im Spieler-Array der entsprechenden Mannschaft), dessen Listenelement aus der verketteten Liste gelöscht werden soll. Als Ergebnis wird wieder der Erfolg bzw. der Nicht-Erfolg als Wahrheitswert zurückgegeben. In der Funktion wird in der verketteten Liste nach dem Element gesucht, dessen Spieler-Zeiger auf den angegebenen Spieler zeigt, dieses Element dann aus der Liste entfernt und der Speicherplatz des Datensatzes vom Typ `sListElement` wieder freigegeben.

Neue Datei `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 (Name des Spielers) und errechnet daraus den Hashwert, der als Funktionsergebnis zurückgegeben wird.

- `search`:

Diese Funktion erhält drei Zeiger als Parameter: einen Zeiger auf das Indexarray `PlayerIndex`, einen Zeiger auf die Vergleichsfunktion zum Vergleichen von zwei Spielernamen (diese Vergleichsfunktion existiert ja bereits vom Sortieren) sowie einen Zeiger auf einen Spieler, in dem der gesuchte Spielername steht. In der Funktion muss der Hashwert errechnet werden (siehe vorige Funktion), um dann in der entsprechenden verketteten Liste linear nach dem gewünschten Spieler zu suchen. Wird der Spieler gefunden, soll ein Zeiger auf das gefundene Listenelement (`sListElement`) zurückgegeben werden; ansonsten ein `NULL`-Zeiger.

Zusätzlich soll im Hauptmenü eine Funktion zum Anzeigen der Hash-Tabelle (Array `PlayerIndex`) implementiert werden. Diese Funktion soll für jeden belegten Eintrag der Hash-Tabelle den Hashwert und eine Liste der dazugehörigen Spielernamen anzeigen (siehe Beispielausgabe). In der vorgegebenen Datendatei `teams.xml` gibt es gleich mehrere Kollisionen, d.h. Spieler mit dem gleichen Hashwert, die in dieser Auflistung anhand der gleichen Hashwerte gut zu sehen sind.

Kommentieren Sie das Programm. Dazu gehört auch ein Modulheader und zu jeder Funktion ein Funktionsheader (siehe Skript "Grundlagen der Informatik" Kapitel 5.3 und 5.4)! Achten Sie auch auf Ihre Programmstruktur (Einrückungen, Leerzeichen und -zeilen).

Beispielausgaben:

Suche:

```
Geben Sie bitte den Namen des gesuchten Spielers ein:  
-> Thomas Mueller
```

```
Suchergebnis:  
-----
```

```
in der Mannschaft FC Bayern Muenchen:  
  01. Thomas Mueller (25; * 13.09.1989; 1 Tor)
```

```
Bitte Eingabetaste druecken ...
```

```
Geben Sie bitte den Namen des gesuchten Spielers ein:  
-> Max Mustermann
```

```
Es wurde kein Spieler mit dem gesuchten Namen gefunden!
```

```
Bitte Eingabetaste druecken ...
```

Ausgabe der Hash-Tabelle:

```
Mannschaften-Verwaltung V0.7  
=====
```

1. Neue Mannschaft anlegen
2. Mannschaft loeschen
3. Suchen
4. Sortieren
5. Auflisten
6. Hash-Tabelle auflisten
7. Programm beenden

```
Ihre Wahl: 6
```

Hashtabelle

=====

Hashwert	Mannschaft	Spieler
0	1. FC Union Berlin	Benedict Hollerbach
2	FC Bayern Muenchen	Joao Palhinha
3	FC Bayern Muenchen	Michael Olise
5	FC Bayern Muenchen	Jamal Musiala
10	Hertha BSC	Marton Dardai
11	1. FC Union Berlin	Aljoscha Kemlein
	1. FC Union Berlin	David Preu
12	FC Bayern Muenchen	Harry Kane
13	FC Bayern Muenchen	Konrad Laimer
18	Hertha BSC	Andreas Bouchalakis
22	Hertha BSC	Tim Goller
23	Hertha BSC	Michal Karbownik
	Hertha BSC	Bilal Hussein
25	FC Bayern Muenchen	Leroy Sane
33	1. FC Union Berlin	Kevin Volland
	Hertha BSC	Linus Gechter
34	FC Bayern Muenchen	Mathys Tel
36	1. FC Union Berlin	Kevin Vogt
40	1. FC Union Berlin	Leopold Querfeld
	FC Bayern Muenchen	Aleksandar Pavlovic
41	FC Bayern Muenchen	Min-jae Kim
	FC Bayern Muenchen	Leon Goretzka
52	Hertha BSC	Toni Leistner
53	1. FC Union Berlin	Frederik Roennow
62	1. FC Union Berlin	Oluwaseun Ogbemudia
	1. FC Union Berlin	Lucas Tousart
65	Hertha BSC	Boris Mamuzah Lum
83	1. FC Union Berlin	Christopher Trimmel
87	Hertha BSC	Diego Demme
92	1. FC Union Berlin	Andrej Ilic
103	1. FC Union Berlin	Danilho Doekhi
108	1. FC Union Berlin	Diogo Leite
119	Hertha BSC	Pascal Klemens
124	1. FC Union Berlin	Jordan Siebatcheu
128	FC Bayern Muenchen	Joshua Kimmich
131	Hertha BSC	Kevin Sessa
134	Hertha BSC	John Brooks
136	FC Bayern Muenchen	Kingsley Coman
140	Hertha BSC	Jeremy Dudziak
143	Hertha BSC	Tjark Ernst
146	FC Bayern Muenchen	Raphael Guerreiro
150	FC Bayern Muenchen	Thomas Mueller
151	Hertha BSC	Marten Winkler
177	Hertha BSC	Julius Gottschalk
188	Hertha BSC	Fabian Reese
189	1. FC Union Berlin	Jerome Roussillon
193	FC Bayern Muenchen	Eric Dier
201	1. FC Union Berlin	Rani Khedira
204	Hertha BSC	Ibrahim Maza
224	FC Bayern Muenchen	Serge Gnabry
232	FC Bayern Muenchen	Manuel Neuer
235	Hertha BSC	Marius Gersbeck
236	1. FC Union Berlin	Tom Rothe
237	1. FC Union Berlin	Ivan Prtajin
238	FC Bayern Muenchen	Dayot Upamecano
239	FC Bayern Muenchen	Sven Ulreich
241	Hertha BSC	Jonjoe Kenny
244	FC Bayern Muenchen	Alphonso Davies
250	Hertha BSC	Derry Scherhant
266	1. FC Union Berlin	Josip Juranovic
267	1. FC Union Berlin	Alexander Schwolow
272	Hertha BSC	Deyovaisio Zeefuik
276	1. FC Union Berlin	Yorbe Vertessen
289	Hertha BSC	Gustav Christensen
297	FC Bayern Muenchen	Sacha Boey
301	1. FC Union Berlin	Janik Haberer

Bitte Eingabetaste druecken ...