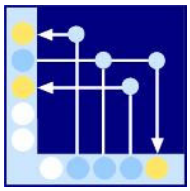
Hochschule Aalen
Fakultät Elektronik und Informatik
Studienbereich Informatik



Algorithmen und Datenstrukturen 2

Vorlesung im Wintersemester 2021/2022
Prof. Dr. habil. Christian Heinlein

1. Praktikumsaufgabe (28. Oktober – 18. November 2021)

Aufgabe 1: Streuwerttabellen

Auf der Vorlesungswebseite befindet sich eine Datei `hashing.h` mit folgenden unvollständigen Klassen (bzw. Strukturen), die jeweils als Schablonen (templates) mit geeigneten Typparametern definiert sind:

- `HashChain<K, V>` implementiert Streuwerttabellen mit Schlüsseltyp `K` und Werttyp `V` mittels Verkettung.
  Hierfür muss eine Funktion `hashval` mit Parametertyp `K` und Resultattyp `uint` (`unsigned int`) zur Berechnung von Streuwerten verfügbar sein.

- `LinProb<K>`, `QuadProb<K>` und `DblHash<K>` implementieren Sondierungssequenzen mit Schlüsseltyp `K` für lineare bzw. quadratische Sondierung bzw. doppelte Streuung.
  Hierfür muss ebenfalls eine derartige Funktion `hashval` verfügbar sein. Für `DblHash` muss außerdem eine zweite Streuwertfunktion `hashval2` mit Parametertypen `K` und `uint` (der zweite Parameter gibt die Tabellengröße an) und Resultattyp `uint` verfügbar sein.

- `HashOpen<K, V, S>` implementiert Streuwerttabellen mit Schlüsseltyp `K` und Werttyp `V` mittels offener Adressierung mit Sondierungssequenzen des Typs `S` (für den also insbesondere `LinProb<K>`, `QuadProb<K>` und `DblHash<K>` eingesetzt werden können).

Implementieren Sie alle leeren und fehlenden Konstruktoren und Elementfunktionen dieser Klassen sowie die hierfür erforderlichen Datenstrukturen, das heißt:

- Den Konstruktor und die Elementfunktionen `put`, `get`, `remove` und `dump` der Klassen `HashChain` und `HashOpen`.

- Den Konstruktor und die Elementfunktion `next` der Klassen `LinProb`, `QuadProb` und `DblHash`.

Beachten Sie unbedingt die in der Datei enthaltenen Kommentare, die das zu implementierende Verhalten spezifizieren!

Um automatisierte Tests der Implementierungen zu ermöglichen, dürfen die vorgegebenen Namen der Klassen und Elementfunktionen nicht verändert werden und es dürfen keine dürfen keine Diagnoseausgaben produziert werden.

Es dürfen keine Klassen der C++-Standardbibliothek verwendet werden!

Abzugeben ist die entsprechend erweiterte Datei `hashing.h`.
Die E-Mail mit der Abgabe muss als Betreff `Algo2 Gruppe NN` mit zweistelliger Gruppennummer `NN` (z. B. `05` oder `12`) und die abzugebende Datei als Anhang haben. Der Nachrichtentext wird ignoriert.

Hinweise zur dynamischen Erzeugung von Feldern in C++

- `new T [n]` erzeugt zur Laufzeit ein Feld mit `n` Elementen des Typs `T` und liefert einen Zeiger des Typs `T*` darauf zurück. Wenn man diesen Zeiger an eine Variable `T* a` zuweist, kann `a` (aufgrund der aus C bekannten Äquivalenz von Zeigern und Feldern) anschließend wie ein Feld verwendet werden; insbesondere bezeichnet `a[i]` für `i` von 0 bis `n-1` das `i`-te Element des Felds.

- Abhängig vom Typ `T`, werden die Elemente des Felds dabei entweder automatisch mit dem parameterlosen Konstruktor von `T` initialisiert, oder sie bleiben uninitialisiert.
  Für elementare Typen sowie einfache Strukturtypen, die nur elementare Typen enthalten, findet z. B. keine Initialisierung statt. Für Typen, die einen oder mehrere Konstruktoren besitzen, wird der parameterlose Konstruktor aufgerufen, den es dann auch geben muss. Die genauen Regeln sind jedoch kompliziert.

- Durch Verwendung von `new T [n] ()` kann die Initialisierung der Elemente mit dem parameterlosen Konstruktor erzwungen werden, was insbesondere für elementare Typen, zu denen auch Zeigertypen zählen, sinnvoll ist; Elemente mit Zeigertypen werden dann automatisch mit `nullptr` initialisiert.

- Ein Feld mit `n` Zeigern des Typs `T*`, die alle mit `nullptr` initialisiert sind, kann somit mittels `new T* [n] ()` erzeugt und an eine Variable `T** a` zugewiesen werden. Jedes Element `a[i]` hat dann Typ `T*`.

Testprogramm

Die ebenfalls auf der Vorlesungswebseite verfügbare Datei `hashtest.cxx` enthält ein einfaches interaktives Testprogramm.

Das Shellskript `hashtest.sh` ruft dieses Testprogramm mit einigen Befehlen jeweils für Verkettung, lineare Sondierung, quadratische Sondierung und doppelte Streuung auf. Bei korrekter Implementierung aller Klassen muss der Befehl `sh hashtest.sh` in einer Linux-Kommandozeile exakt die folgende Ausgabe liefern:

```
c
4 (1, 3) eins
6 (3, 3) drei
6 (2, 4) zwei
l
4
5 (1, 3) eins
6 (2, 4) zwei
7 (3, 3) drei
q
4
5 (1, 3) eins
6 (2, 4) zwei
7 (3, 3) drei
d
1 (1, 3) eins
4
5 (3, 3) drei
6 (2, 4) zwei
```

Organisatorische Hinweise

Beachten Sie hierzu das Dokument „Wichtige Hinweise zur Prüfung“ auf der Vorlesungswebseite!