Fakultät IV Elektrotechnik und Informatik
Einführung in die Programmierung WS 2022/2023
Manfred Hauswirth
Damien Foucard / Uwe Kuehn / Aljoscha Meyer

Programmierblatt 06

Ausgabe: *08.12.2022 16:00*
Abgabe: *03.01.2023 08:00*

Thema: *Binäre Suchbäume*

Abgabemodalitäten

1. Alle abzugebenden Quelltexte müssen ohne Warnungen und Fehler auf Deinem Rechner mit dem Befehl `clang -std=c11 -Wall -g` kompilieren.
2. Die Abgabe für den Quellcode erfolgt ausschließlich über unser Git im entsprechenden Branch. Nur wenn ein Ergebnis im ISIS-Kurs angezeigt wird, ist sichergestellt, dass die Abgabe erfolgt ist. Die Abgabe ist bestanden, wenn Du an Deinem Test einen grünen Haken siehst.
3. Du kannst bis zur Abgabefrist beliebig oft neue Versionen abgeben. Lies Dir die Hinweise der Tests genau durch, denn diese helfen Dir Deine Abgabe zu korrigieren.
   Bitte beachte, dass ausschließlich die letzte Abgabe gewertet wird.
4. Die Abgabe erfolgt, sofern nicht anders angegeben, in folgendem Branch: `iprg-b<xx>-a<yy>`, wobei `<xx>` durch die zweistellige Nummer des Aufgabenblattes und `<yy>` durch die entsprechende Nummer der Aufgabe zu ersetzen sind.
5. Gib für jede Aufgabe die Quellcodedatei(en) gemäß der Vorgabe ab. Im ISIS-Kurs werden zum Teil Vorgabedateien bereitgestellt. Nutze diese zur Lösung der Aufgaben.
6. Die Abgabefristen werden vom Server überwacht. Versuche Deine Abgabe so früh wie möglich zu bearbeiten. Du minimierst so auch das Risiko, die Abgabefrist auf Grund von „technischen Schwierigkeiten“ zu versäumen. Eine Programmieraufgabe gilt als bestanden, wenn alle bewerteten Teilaufgaben bestanden sind.

Aufgabe 1 Telefonbuch implementieren (bewertet)

In dieser Aufgabe soll ein Telefonbuch als binärer Suchbaum implementiert werden. Die Daten müssen eingelesen und eingefügt werden, und danach durchsuchbar sein und in Gesamtheit ausgegeben werden. Die Telefonnummern sind dabei natürliche Zahlen mit maximal neun Ziffern (also 1 und 999999999, aber nicht -2, 3.5 oder 1234567890) und sollen so angeordnet werden, dass kleinere Zahlen immer links und größere Zahlen immer rechts platziert werden (siehe Abbildung 1). Es dürfen keine Zahlen mehrfach vorkommen.

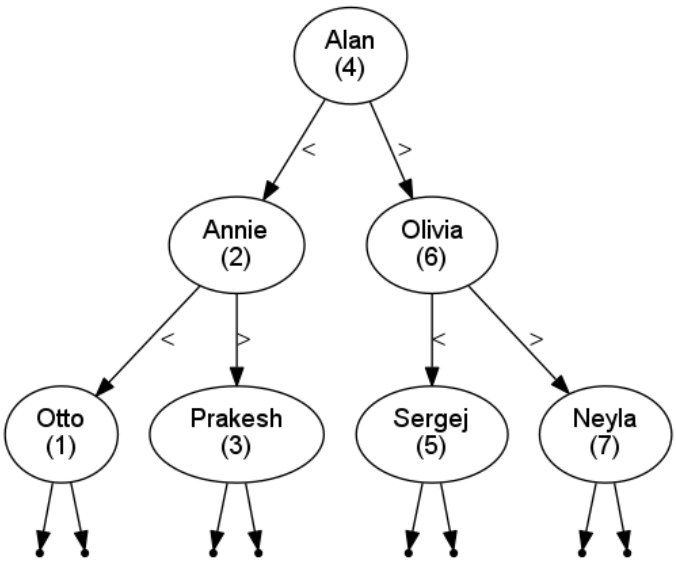

Abbildung 1: Beispiel-Telefonbuch als binärer Suchbaum

Du kannst Dich auf die Implementierung der folgenden Funktionen beschränken:

- `bst_insert_node(bstree* bst, unsigned long phone, char *name)`
  - Platziere einen neuen Knoten in den Baum `bst`.
  - Halte die Ordnung wie oben beschrieben ein.
  - Verändere, wenn nötig, den Pointer auf den Wurzelknoten `bst->root`.
  - Gebe eine Fehlernachricht aus und verlasse die Funktion mit `return`, wenn eine Telefonnummer übergeben wird, die schon im Baum existiert.
- `bst_node* find_node(bstree* bst, unsigned long phone)`
  - Finde den Knoten im Baum, der die angegebene Telefonnummer besitzt, und gebe ihn zurück.
  - Gebe `NULL` zurück, wenn es keinen entsprechenden Knoten gibt.
- `bst_in_order_walk_node(bst_node* node)`
  - Gebe den Unterbaum von `node` in in-order Reihenfolge aus (`bst_in_order_walk(bstree* bst)` aufgerufen)
  - Benutze dafür die Funktion `print_node`. Sie benötigt als einzigen Parameter einen Pointer auf den gegenwärtigen Knoten (also im Format `bst_node*`).
- `bst_free_subtree(bst_node* node)`
  - Gebe den Speicher eines Teilbaums des binären Suchbaums frei (wird von `bst_free_tree(bstree* bst)` aufgerufen)
  - Achte dabei darauf, dass Du den Baum „post-order“ traversierst

Um die Aufgabe übersichtlicher zu gestalten, ist der Code auf vier Dateien verteilt:

introprog_input_telefonbuch.c	enthält Eingabefunktionen und kleine Hilfsfunktionen. Hier solltest Du nichts verändern.
introprog_main_telefonbuch.c	kommt dazu, um das Interface von der eigentlichen Logik zu trennen. Hier wird die `main` Funktion implementiert und auch hier solltest Du nichts verändern.
introprog_telefonbuch.c	In dieser Datei sollst Du die oben genannten Funktionen implementieren. Neu ist dabei, dass sich die `main` Funktion und die `struct` in anderen Dateien befinden.
introprog_telefonbuch.h	ist die Header Datei, die alle Dateien miteinander verknüpft. In dieser Aufgabe sind an dieser Stelle auch die `struct` und `typedef`, sowie die `includes` definiert. Auch hier solltest Du nichts verändern.

Bei der Kompilierung muss, wie gehabt die Datei `introprog_input_telefonbuch.c` und zusätzlich `introprog_main_telefonbuch.c` angegeben werden. Das Programm benötigt als Parameter die Angabe des Telefonbuchs. Es sind dafür in den Vorgaben Folgendes vorhanden:

telefonbuch_leer.txt	Ein leeres Telefonbuch
telefonbuch_einfach.txt	Ein kleines Telefonbuch-Beispiel
telefonbuch_gross.txt	Ein großes Telefonbuch-Beispiel

Listing 1: Programmbeispiel

```
> clang -std=c11 -Wall introprog_telefonbuch.c \
      introprog_input_telefonbuch.c introprog_main_telefonbuch.c \
      -o introprog_telefonbuch
> ./introprog_telefonbuch telefonbuch_gross.txt
 Fernsprech-Datensatz-System
 ===========================
  Füge in das Telefonbuch ein:  + <Nummer> <Name>
  Gebe das Telefonbuch aus:     p
  Finde den Namen:              ? <Nummer>
  Debugausgabe des Baumes:      d
  Beende das System:            q
```

Um das Debugging zu vereinfachen, haben wir eine Debugging-Funktionalität eingebaut. Mittels 'd' lässt sich beim Aufruf des Programms der binäre Suchbaum in eine PNG-Datei ausgeben. Dafür wird die Software Graphviz benötigt. Auf den Rechnern in den Terminalräumen sollte diese schon installiert sein. Auf den eigenen Rechnern muss diese ggfs. nachinstalliert[1] werden.

Nutze zur Lösung der Aufgabe die Vorgaben aus unserem ISIS-Kurs. Füge Deine Lösung als Datei `introprog_telefonbuch.c` im entsprechenden Abgabebranch in Dein persönliches Repository ein und übertrage die Lösung an die Abgabeplattform.

[1] Du findest die aktuellen Packages der verschiedenen Betriebsysteme hier: https://graphviz.org/download/.