Praktikum 2

Einfach nur sortieren …

Aufgabenstellung

In diesem Praktikum sollen Sie „Binary Tree Sort“ implementieren.

Dieser Algorithmus fügt Elemente der Reihe nach in einen binären Suchbaum ein und gibt sie dann mit einer Inorder-Traversierung sortiert aus.

In dieser Aufgabe bestehen diese Elemente aus je einem Vornamen, einem Nachnamen und einer 9-stelligen Zahl. Sie werden in einer Eingabedatei zur Verfügung gestellt, die in folgendem Format vorliegt:

```
<Anzahl der Einträge N>
<Vorname1>;<Nachname1>;<Nummer1>
<Vorname2>;<Nachname2>;<Nummer2>
⋮
<VornameN>;<NachnameN>;<NummerN>
```

Abbildung 1: Eingabedatei

Lesen Sie die Datei „prak2.txt“ vollständig ein. Sie können davon ausgehen, dass die Datei dem vorgegebenen Format entspricht. Die Vor- und Nachnamen bestehen ausschließlich aus den Buchstaben A-Z und die Zahlen aus dezimalen Ziffern $[0-9]$. Die Worte beginnen mit einem Großbuchstaben und bestehen sonst nur aus Kleinbuchstaben. Was hinter dem N. Eintrag kommt, ignorieren Sie.

Fügen Sie die Elemente der Reihe nach in einen binären, unbalancierten Suchbaum ein. (Sie können auch einen balancierten Baum verwenden nehmen. Dies ist jedoch deutlich aufwändiger.) Das Sortierkriterium ist der Nachname und — bei gleichen Nachnamen — der Vorname. Gleiche Paare aus Vor- und Nachname kommen nicht vor.

Als Ausgabe sollen die Elemente des Baumes sortiert in folgendem Format ausgegeben werden:

```
1. <VornameA> <NachnameA>: <NummerA>
2. <VornameB> <NachnameB>: <NummerB>
⋮
```

Abbildung 2: Ausgabe

Vorgehensweise

Überlegen Sie zunächst, wie Sie den Binärbaum im Speicher darstellen wollen. Da es auf Assemblerebene keine Speicherverwaltung (im Sinne von Befehlen wie „`new`“ o.ä.) gibt, müssen Sie dies selbst übernehmen. Sie könnten etwa einen großen, kontinuierlichen Speicherbereich auf dem Heap mithilfe des MARS-Kommandos `.space` reservieren und diesen der Reihe nach mit Daten füllen.

Implementieren Sie folgende Hilfsfunktionen:

atoi wandelt einen String in eine Ganzzahl um, also bspw. "42"→ 42 (ähnlich der Java-Funktion `Integer.parseInt(String s)`. Sie benötigen diese Funktion zum Beispiel zum Bearbeiten der erste Zeile der Eingabedatei.

strcmp vergleicht zwei Strings und gibt zurück, welcher der beiden lexikografisch größer ist.

insert fügt einen Eintrag in den Suchbaum ein.

print_inorder gibt den Bauminhalt in Inorder-Reihenfolge aus.

All diese Funktionen müssen den Aufrufkonventionen folgen, die in der Vorlesung vorgestellt wurden. Die genaue Spezifikation erstellen Sie selbst und dokumentieren die Funktionen im Code genau (insb. Eingabe und Ausgabe).

Gehen Sie sparsam mit dem Speicher um! Der MARS-Simulator stellt für das Datensegment nur 4 MB zur Verfügung. Sie sollten Eingabedateien bis zu einer Grösse von 512 KB (2^{19} Bytes) unterstützen.

Verwenden Sie den Programmrahmen `prak2.s`, der auf unserer Homepage zur Verfügung steht. Dort finden Sie auch weitere Materialien und Testfälle.

Einreichen und Testieren

Laden Sie ihren Quellcode bis zum 18.11.2007 auf folgender Webseite hoch:

`http://www.sec-tud.de/index.php?page=pages/lehre/upload/`

Sie benötigen hierfür einen RBG-Account. Bitte melden Sie sich in Gruppen von je drei Personen an. Diese Gruppen sind unabhängig von den Teams/Gruppen der Übungen.

Die Testierung findet in der Woche ab dem 19.11.2007 statt. Sie können sich wie gewohnt über das Uploadsystem für einen Testatstermin anmelden.

Nutzen Sie für weitere Fragen das Fachschaftsforum und die Sprechstunden.