Prof. Dr. Franz Schweiggert   Institut für Angewandte Informationsverarbeitung   10. Januar 2009
Matthias Röhm

Allgemeine Informatik III / Systemnahe Software I (WS 2008)

Abgabetermin: 20.01.2009

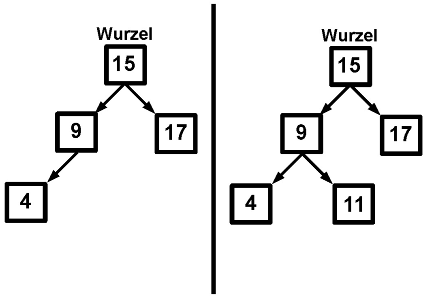
1 Theorie (5 Punkte)

- Was bedeutet „Call by reference“ und wie wird diese in C erreicht?
- Erläutern Sie die Funktionsweise des Tools „make“.
- Eine Implementierung des Binärbaums aus Aufgabe 2 besteht aus den Dateien: main_baum.c (das Hauptprogamm), baum.c und baum.h (Implementierung des Binärbaums). Schreiben Sie ein Makefile welches aus main_baum.c, baum.c und baum.h eine ausführbare Datei „bin_baum“ erzeugt.
- Die beiden folgenden Strukturen beschreiben ein Haus und seine Bewohner. Ein Haus kann beliebig viele Bewohner haben (dynamischer Vektor). Vervollständigen Sie das folgende Code-Fragment, sodass ihr Haus Platz für die beiden Bewohner bietet. Arbeiten Sie hierfür mit *calloc*. Lösen Sie die Aufgabe mit Papier und Bleistift!

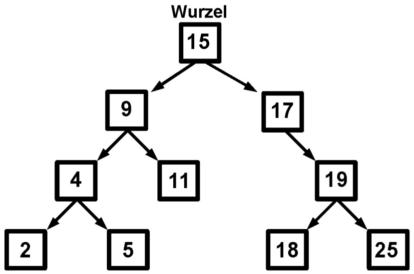
```
#include <stdlib.h>
#include <stdio.h>

struct bewohner {
 char *name; //Der Name
 int alter; //Das alter
 int geschlecht; //Das Geschlecht: 0=Mann, 1=Frau
};
struct haus {
 struct bewohner **bew; //Zeiger auf die Bewohner
 int plz; //Die Postleitzahl
 char *strasse; //Die Strasse
 int nummer; //Die Hausnummer
};

int main( ) {
 //Haus "bauen"
 struct haus *kleines_haus = ......;

 //Haus braucht Platz fuer 2 Bewohner!
 kleines_haus->bew = ......;
 kleines_haus->strasse = "Hauptstr.";
 kleines_haus->plz = 89073;
 kleines_haus->nummer = 1;

 //Beide Bewohner erzeugen
 kleines_haus->bew[0] = ......;
 kleines_haus->bew[1] = ......;

 kleines_haus->bew[0]->name = "Hans";
 kleines_haus->bew[0]->alter = 25;
 kleines_haus->bew[0]->geschlecht = 0;

 kleines_haus->bew[1]->name = "Maria";
 kleines_haus->bew[1]->alter = 23;
 kleines_haus->bew[1]->geschlecht = 1;

 //Ausgabe der Bewohner
 printf("name: %s, alter: %d, geschlecht: %d\n", kleines_haus->bew[0]->name,\
  kleines_haus->bew[0]->alter, kleines_haus->bew[0]->geschlecht );
 printf("name: %s, alter: %d, geschlecht: %d\n", kleines_haus->bew[1]->name,\
  kleines_haus->bew[1]->alter, kleines_haus->bew[1]->geschlecht );

 return 0;
}
```

2 Binärbäume (5 Punkte)

Ein Binärbaum ist eine weitere Datenstruktur zum schnellen auffinden bestimmter Elemente. In dieser Aufgabe soll ein Binärbaum zur Verwaltung von Integer-Zahlen erstellt werden. Zur Erinnerung, hier noch einmal die Eigenschaften eines Binärbaumes in aller Kürze:
Jedes Element des Baumes besteht aus einem Integer-Wert und jeweils einem Zeiger auf das linke und rechte „Kind“-Element. Beim Einfügen eines neuen Elementes wird von der Wurzel aus überprüft, ob der Wert des neuen Elementes gleich, kleiner oder größer als der Wert des aktuellen Elementes ist. Ist der Wert des neuen Elementes **gleich** dem des aktuellen Elementes, ist dieser Wert bereits vorhanden und der Vorgang wird **abgebrochen**. Ist der Wert **kleiner**, wird im **linken** Teilbaum weitergesucht, ist der Wert **größer**, im **rechten** Teilbaum. Eingefügt wird das neue Element, sobald ein Blatt des Baumes erreicht ist.



Ein Binärbaum nach dem Einfügen der Werte 15,9,17,4. Einfügen der 11.



Ein Binärbaum nach dem Einfügen der Werte 15,9,17,4,11,2,5,19,18,25.

2.1 Implementierung

Schreiben Sie ein Programm welches einen Binärbaum erzeugt. Ihr Programm soll zwei Funktionen implementieren:

- *put*. Einfügen eines neuen Wertes in den Baum.
- *print*. Ausgabe aller Werte eines Baumes. Z.B. als Liste: $15, 9, 17, 4, ...$

Füllen Sie den Baum mit den Werten $15, 9, 17, 4, 11, 2, 5, 19, 18, 25$. Anschließend geben Sie alle Werte aus.

Viel Erfolg!