Technische Universität München **WS 2006/07**
Institut für Informatik
Prof. Manfred Broy **29. Januar 2007**
A. Gruler, B. Spanfelner **Übungsblatt 11**

Übungen zu „Formale Methoden“

Aufgabe 1 Referenzen

Gegeben ist das folgende Programm zur Erläuterung des Konzepts der Referenzen anhand verketteter Listen.

(a) Erklären Sie anhand einer Skizze was beim Einfügen bzw. Entfernen eines Knotens passiert.

(b) Was passiert im Speicher, wenn ein Knoten eingefügt bzw. entfernt wird ?

(c) Erläutern Sie mit Hilfe dieses Programms das Konzept der Referenzen, wie es in der Vorlesung eingeführt wurde.

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

struct NODE
{
     int nodevalue;
     struct NODE *next;
}
struct NODE *list, *target;

main()
{
     /*initialisation*/
     int delval;
     list := (struct NODE *)malloc(sizeof(NODE));
     tail := list
     /*tail is a pointer to move within list's tail*/
     ..
     /*Here some nodes are added to the List*/
     ..
     tail := list;
     ..
     /*Again some nodes are added to the List*/
     ..
     /*tail is assigned, some how, the position number 2*/
```

```
    addnode(48,tail);
    ..
    /*tail is assigned, some how, the position number 5*/
    addnode(34,tail);
    ..
    ..
    /*tail is assigned, some how, the position number 2*/
    delval := deletenode(tail);
    ..
    /*tail is assigned, some how, the position number 5*/
    delval := deletenode(tail);
    ..
    ..
    createlist(48);
    ..
    createlist(34);
    ..
    ..
}

createlist(int value) /*our algorithm starts here*/
{
    struct NODE *new;
    if(list != NULL )
    {
        list->nodevalue := value;
        list->next := NULL;
    }
    else
    {
        if(new=(struct NODE*)malloc(sizeof(NODE)))
        {
            new->nodevalue := value;
            new->next := NULL;
            list->next := new;
            list := new;
        }
        else
            printf("Memory shortage");
        }
    }

addnode(int value, struct NODE *position)
{
    struct NODE *new;
    if(new == (struct NODE*)malloc(sizeof(NODE)))
    {
        new->nodevalue := value;
```

```
                new->next := position->next;
                position->next := new;
            }
            else
                printf("Memory shortage");
        }

        int deletenode(struct NODE *position)
        {
            int deletedvalue := position->nodevalue;
            struct NODE *curr := list;
            while(current->next != position)
               current := current->next;
            current->next := position->next;
            free(position);
            return(deletedvalue);
       }
```

Aufgabe 2 Object Store

Schreiben Sie folgendes Programm in einer Notation, die den Objectstore verwendet.

```
x: var Nat = 10;
y: var Nat = 20;

if x >= y then x:= x * 10;
else y:= y*10;

r: var Ref var Nat;
s: var Ref var Nat;

New(r);
New(s);

deref(s) := 12;
deref(r) := 15;

x:= deref(s) * deref(r);
```

Aufgabe 3 (Hausaufgabe) Terminierung von Schleifen
(4 Punkte; Abgabe bis spätestens 05.02.2007 zu Beginn der Übung.)

In Aufgabe 1, Blatt 10, haben Sie bereits die partielle Korrektheit des folgenden Programmes P gezeigt.

$$
\begin{array}{l}
i,\ s\ \ \textbf{var}\ Nat; \\
\\
i,\ s := 1,\ b[0]; \\
\textbf{do}\ \ (i < 11)\ \ \textbf{then} \\
\quad i,\ s := i{+}1,\ s{+}b[i]; \\
\textbf{od}
\end{array}
$$

Beweisen Sie nun noch die Terminierung von P.

(a) Geben Sie hierzu einen passenden Terminierungsausdruck (Terminierungsfunktion) E an.

(b) Beweisen Sie mit Hilfe von E die Terminierung von P.