5. Übung – Informatik I für VIW

Fakultät Verkehrswissenschaften
Fachrichtung Verkehrsingenieurwesen

Zeitraum: 13.12.2010 bis 07.01.2011 (WS 2010/11)

Aufgabe 1

Gegeben sei folgendes *C*-Programm:

```
 1  #include <stdio.h>
 2
 3  void g (int *x, int *y, int z) {
 4    *y = *x + z;
 5    /*label 1*/
 6  }
 7
 8  void f (int d, int *e) {
 9    /*label 2*/
10    if (d >= 0) {
11      f(d - 1, e); /* $1 */
12      /*label 3*/
13      g(&d, e, d); /* $2 */
14    }
15    else {
16      *e = 0;
17      /*label 4*/
18    }
19    *e = d + 2;
20  }
21
22  int main () {
23    int a=1, c;
24    /*label 5*/
25    f(a, &c);      /* $3 */
26    /*label 6*/
27    return 0;
28  }
```

Stellen Sie ein Speicherbelegungsprotokoll für den Ablauf des Programms dar, wobei die aktuelle Situation bei jedem Passieren der Marken `label1` bis `label6` gezeigt werden soll. Dokumentieren Sie jeweils alle benutzten Speicherzellen mit ihrer Wertebelegung (bei unbekannten Werten geben Sie ein Fragezeichen an; bei Pointern geben Sie die verwiesene Speicherzelle durch einen Pfeil an) und zeigen Sie jeweils an, welche Variablen mit welchen Speicherzellen assoziiert sind.

Beachten Sie: `$1$` bis `$3$` seien die bereits festgelegten Rücksprungmarken; diese können Sie nutzen, um sich zu merken, an welche Positionen Sie beim Beenden einer Funktion zurückspringen müssen.

Aufgabe 2

Gegeben sei folgendes *C*-Programm:

```
 1  #include <stdio.h>
 2
 3  void g (int a, int *b)
 4  { *b = a;
 5      /*label 1*/
 6  }
 7
 8  void f (int x, int y, int *z)
 9  { *z = y;
10    while (x < y)
11    {
12      /*label 2*/
13      if (x == *z)
14        f(x / 3, *z - 2, z); /*$1*/
15      else
16      { x = x + 1;
17        g(x, z); /*$2*/
18      }
19    }
20      /*label 3*/
21  }
22
23  int main ()
24  { int e=2, a;
25      /*label 4*/
26    f(e, e * 2, &a); /*$3*/
27      /*label 5*/
28    return 0;
29  }
```

Stellen Sie ein Speicherbelegungsprotokoll für den Ablauf des Programms dar, wobei die aktuelle Situation bei jedem Passieren der Marken `label1` bis `label5` gezeigt werden soll. Dokumentieren Sie jeweils alle benutzten Speicherzellen mit ihrer Wertebelegung (bei unbekannten Werten geben Sie ein Fragezeichen an; bei Pointern geben Sie die verwiesene Speicherzelle durch einen Pfeil an) und zeigen Sie jeweils an, welche Variablen mit welchen Speicherzellen assoziiert sind.

Beachten Sie: `$1$` bis `$3$` seien die bereits festgelegten Rücksprungmarken; diese können Sie nutzen, um sich zu merken, an welche Positionen Sie beim Beenden einer Funktion zurückspringen müssen.

Aufgabe 3

(a) Geben Sie für jede der folgenden Laufzeiten die O-Notation an:

- $1000 \cdot n^4 + 0.0001 \cdot 2^n$,
- $1/2 \cdot n \cdot (1 + \log n)$,
- $2 + 4 + 6 + 8 + \cdots + (n-2) + n$,
- $\log(n^2 * 2^n)$

(b) Bestimmen Sie die Laufzeitkomplexitäten der folgenden Funktionen.

```
void f1(int feld[], int n) {
   int i;
   for (i=0; i<n; i+=2) {
     feld[i] ++;
   }
}

void f2(int feld[], int n) {
   int i;
   for (i = 1; i<n; i*=2) {
     feld[i] ++;
   }
}

void f3(int feld[], int n) {
   int i, j;
   for (i = 1; i<n; i*=2) {
     for (j = 0; j < i; j++) {
       feld[j] ++;
     }
   }
}
```

Aufgabe 4 (Zusatz)

In der achten Vorlesung wurde ein Programm zur Berechnung der zehnten Silhouette der Drachenkurve vorgestellt, welches die Nachfolgersilhouetten jeweils mit Hilfe der Funktion `calcNext` berechnet hatte (siehe Folien 32 und 33).

In dem Programm werden zehn Arrays `sil1` bis `sil10` benutzt. Stattdessen kann man jedoch zwei Arrays `feld1` und `feld2` mit jeweils 1023 Einträgen deklarieren und wie folgt vorgehen:

1. Man speichert die erste Silhouette in `feld1` (dabei nutzt man nur den ersten Eintrag des Arrays)

2. Man berechnet die zweite Silhouette in speichert sie in `feld2` (wobei man nur die ersten 3 Einträge benutzt).

3. Die erste Silhouette braucht man jetzt nicht mehr. Man berechnet die dritte Silhouette aus der zweiten und speichert sie in `feld1` (dabei nutzt man die ersten 7 Einträge), überschreibt also die erste Silhouette.

4. Nun braucht man auch die zweite Silhouette nicht mehr. Man berechnet die vierte Silhouette aus der dritten und speichert sie in `feld2` (dabei nutzt man die ersten 15 Einträge), überschreibt also die zweite Silhouette.

5. So fährt man fort, man berechnet jeweils eine neue Silhouette und speichert diese abwechselnd in `feld1` und `feld2`

Schreiben Sie ein C-Programm, welches die zehnte Silhouette nach dieser Idee berechnet und für jeden Berechnungsvorgang die Funktion `calcNext` verwendet.