Algorithmen und Datenstrukturen

5. Übungsblatt

Zeitraum: 16. November – 20. November 2015

Übung 1

Implementieren Sie rekursive Funktionen zur Bestimmung des n-ten Glieds der nachstehenden Folgen.[1]

(a) Die Folge der Fibonacci-Zahlen

(b) Die Folge $G\colon \mathbb{N} \to \mathbb{N}$, definiert durch

$$G(0) = 0\,, \qquad G(n) = n - G(G(n-1)) \quad \text{für } n > 0.$$

(c) Die Folgen $F, M\colon \mathbb{N} \to \mathbb{N}$, definiert durch

$$F(0) = 1\,, \qquad F(n) = n - M(F(n-1)) \quad \text{für } n > 0,$$
$$M(0) = 0\,, \qquad M(n) = n - F(M(n-1)) \quad \text{für } n > 0.$$

Übung 2 (AGS 4.22)

Gegeben sei folgendes C-Programm.

```
 1 #include <stdio.h>
 2
 3 void g(int x, int* y);
 4
 5 void f(int* x, int y){
 6   /* label1 */
 7   while (*x < y){
 8     *x = *x * 3;
 9     /* label2 */
10     g(*x, &y);    /* $1 */
11   }
12 }
13
14 void g(int x, int* y){
15   /* label3 */
16   if (*y < x){
17     *y = *y * 2;
18     /* label4 */
19     if (x > *y)
20       f(&x, *y);  /* $2 */
21   }
22   /* label5 */
23 }
```

[1] Siehe https://en.wikipedia.org/wiki/Hofstadter_sequence

```
24
25 int main(){
26   int a, b;
27   a = 3;
28   b = 6;
29   /* label6 */
30   f(&a, b);       /* $3 */
31   /* label7 */
32   printf("%d", a);
33   return 0;
34 }
```

(a) Tragen Sie den Gültigkeitsbereich jedes Objektes in die Tabelle neben dem Programm ein. Nutzen Sie dazu die Zeilennummern.

(b) Setzen Sie das folgende Speicherbelegungsprotokoll fort. Dokumentieren Sie die aktuelle Situation beim Passieren der Marken (*label1* bis *label7*). Geben Sie jeweils den Rücksprungmarkenkeller und die *sichtbaren* Variablen mit ihrer Wertebelegung an. Die Inhalte von Speicherzellen nicht-sichtbarer Variablen sollen Sie nur bei Änderungen eintragen. Beachten Sie: *$1* bis *$3* seien die bereits festgelegten Rücksprungmarken.

Label	Rücksprungmarken	1	2	3	4	5	6	7	8
label6	–	a 3	b 6						

Übung 3

Schreiben Sie eine C-Funktion `swap` mit zwei Parametern, welche die Werte der aktuellen Parameter `x` und `y` vertauscht. Ist `x` zudem ungerade, so soll der Wert von `y` um eins erhöht werden.

Zusatzaufgabe 1 (AGS 3.1.8 ⋆)

Gegeben sei die Ackermann-Funktion $\mathrm{ack}\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. Implementieren Sie diese in C.

$$\begin{aligned} \mathrm{ack}(0, y) &= y + 1 && (y \geq 0) \\ \mathrm{ack}(x, 0) &= \mathrm{ack}(x-1, 1) && (x > 0) \\ \mathrm{ack}(x, y) &= \mathrm{ack}(x-1, \mathrm{ack}(x, y-1)) && (x, y > 0) \end{aligned}$$

Zusatzaufgabe 2 (AGS 4.18)
Gegeben sei folgendes C-Programm.

```
 1  #include <stdio.h>
 2
 3  void g(int n, int *p);
 4
 5  void f(int m, int *q) {
 6    /* label1 */
 7    if (m > 0) {
 8      g(m - 1, q); /* $1 */
 9      /* label2 */
10      g(m - 2, &m); /* $2 */
11      *q = *q + m;
12    } else {
13      *q = 1;
14    }
15    /* label3 */
16  }
17
18  void g(int n, int *p) {
19    int x;
20    /* label4 */
21    if (n < 0) {
22      *p = 3;
23    } else {
24      f(n, &x); /* $3 */
25      *p = 2 * x;
26    }
27    /* label5 */
28  }
29
30  int main() {
31    int x;
32    /* label6 */
33    f(1, &x); /* $4 */
34    printf("%d\n", x);
35    /* label7 */
36    return 0;
37  }
```

(a) Geben Sie den Gültigkeitsbereich jedes Objektes des Programms an. Nutzen Sie dazu die Zeilennummern.

(b) Setzen Sie das folgende Speicherbelegungsprotokoll fort.

Haltepunkt	RM	Umgebung								
		1	2	3	4	5	6	7	8	9
label6	–	x ?								