Antwort für Übungsblatt 1

Jian Dong	Zezhi Chen	Hanyu Sun
jd81vuti	zc75diqa	hs54keri

April 25, 2019

1 P1 (Gruppendiskussion)

(a) Algorithmus
Ein Algorithmus ist eine endliche Sequenz von Handlungsvorschriften, die eine Eingabe in eine Ausgabe transformiert.

(b) Schleifeninvariante
Schleifeninvariante ist die explizite Aussage über den Schleifendurchlauf, die am Anfang und Ende jedes Schleifendurchlaufs und auch vor und nach der Schleife wahr ist.

(c) Totale Ordnung
Sei eine nicht leere Menge M und eine binänre Relation R auf gegeben, und sie erfüllen die Flogende Eigenschaften:
Reflexivität: $\forall x \in M: x \leq x$
Transivität: $\forall x,y,z \in M: x \leq y \land y \leq z \Rightarrow x \leq z$
Antisymmetrie: $\forall x,y \in M: x \leq y \land y \leq x \Rightarrow x = y$
Totalität: $\forall x \in M: x \leq x \lor y \leq x$
Das heißt, bei irgenden zwei Elementen x,y in Menge M bestehen eine Reation (x R y oder y R x).

2 P2 (Insertion Sort)

(a)

```
FOR j = 1 to A.length - 1
  key = A[j]
  i = j - 1
  WHILE I >= 0 and A[i] < key
    A[i+1] = A[i]
    i = i - 1
  A[i+1] = key
```

(b)
input : [“auf”,“Baum”,“Daten”,“Haus”,“sortieren”]
j = 1, key = ”Baum”, : [“auf”,“auf”,“Daten”,“Haus”,“sortieren”]
j = 1, key = ”Baum”, : [“Baum”,“auf”,“Daten”,“Haus”,“sortieren”]
j = 2, key = ”Daten” : [“Baum”,“auf”,“auf”,“Haus”,“sortieren”]
j = 2, key = ”Daten”, : [“Baum”,“Baum”,“auf”,“Haus”,“sortieren”]
j = 2, key = ”Daten”, : [“Daten“,“Baum”,“auf”,“Haus”,“sortieren”]
j = 3, key = ”Haus”, : [“Daten”,“Baum”,“auf”,“auf”,“sortieren”]
j = 3, key = ”Haus”, : [“Daten”,“Baum”,“Haus“,“auf”,“sortieren”]
j = 4, key = ”sortieren”, : [“Daten”,“Baum”,“Haus“,“auf”,“auf”]
j = 4, key = ”sortieren”, : [“Daten”,“Baum”,“Haus“,“Haus”,“auf”]
j = 4, key = ”sortieren”, : [“Daten”,“Baum”,“Baum“,“Haus”,“auf”]
j = 4, key = ”sortieren”, : [“Daten”,“Daten”,“Baum“,“Haus”,“auf”]
j = 4, key = ”sortieren”, : [“sortieren”,“Daten”,“Baum“,“Haus”,“auf”]
output : [“sortieren”,“Daten”,“Baum“,“Haus”,“auf”]

3 P3 (Eigenschaften von Algorithmen)

Algorithmus1:
Sortierung der Elemente einer Liste in absteigender Reihenfolge
Erfüllte Eigenschaften:
Determiniertheit, Korrektheit
Algorithmus2:
Output ist eine Aussage, ob die eingegebene Zahl n Primzahl ist.
Erfüllte Eigenschaften:
Determiniertheit, Determinismus, Terminierung, Korrektheit,

4 P4 (Laufzeiten)

	sqrt(n)	n	$n\,log_2(n)$	n^2	n^3	2^n
1 Sekunde	1000.000	1000	140	31	10	6
1 Stunde	360.000.000	60.000	4895	244	39	11
1 Tag	207.360.000.000	3.600.000	204.094	1897	153	15
1 Monat	186.624.000.000.000	108.000.000	4.861.992	10392	476	18
1 Jahr	24.862.980.000.000.000.000	1.314.000.000	51.302.995	36249	1.095	18
1 Jahrhundert	248.629.800.000.000.000.000.000	131.400.000.000	4.114.224.723	362491	5.083	25

5 P5 (Türme von Hanoi)

(a)
Es gibt n Scheiben auf dem Stab A , und das Ziel ist Versetzen dieser Scheiben auf den Stab C
Das Algorithmus ist wie folgend:

Listing 1: Java Code

```
import java.util.Scanner;
public class Honio {
  public static void move(int n,char A,char B,char C){
  // obere n Scheiben von dem Stab A auf dem Stab C versetzen
    if(n==1){
      System.out.println(A+"->"+C);
    }
    else{
      move(n-1,A,C,B);
      move(1,A,B,C);
      move(n-1,B,A,C);
    }
  }
  public static void main(String[] args){
        int Schieben;
        Scanner in = new Scanner(System.in);
        System.out.println("Wie viel Schieben:");
        Schieben = in.nextInt();
        Honio.move(Schieben,'A','B','C');
  }
}
```

(b)
$f(n) = (2^n - 1) * 1$
$f(n = 64) = 2^{64} - 1 = 1.844\,674 \times 10^{19} s = 5.849\,423 \times 10^{11} Jahre$
Aussage: Es dauert $5.849\,423 \times 10^{11}$ Jahre.
(c^*)
Logisch gesehen ist unser Algorithmus optimal. Und der Zeitaufwand aus (b) ist ganz groß, dass wir keine Sorge für den Weltuntergang brauchen.