AuD: große Übung
5.11.2015

Christian Scheffer
Jan-Marc Reinhardt

Vorlesung: Dienstag, 09:45 - 11:15, SN 19.1 und Mittwoch, 11:30 - 13:00, SN 19.1
Große Übung: Donnerstag, 11:30 - 13:00, SN 19.1 (14-täglich)
Kleine Übung: siehe folgende Tabelle

- Algorithmenbegriff
- Graphen
- Suche in Graphen
- Korrektheit und Komplexität von Algorithmen
- Datenstrukturen
- Sortieren
- Rekursionen

Anmeldung bis heute 14:00!

Gruppe	Termin (14-täglich)	Raum	Tutor	Teilnehmer
1	Montag, 8:00 - 9:30	IZ 305	Sören van der Wall	TBA
2	Dienstag, 15:00 - 16:30	IZ 305	Christian Rieck	TBA
3	Mittwoch, 08:00 - 09:30	IZ 305	Dennis Luck	TBA
4	Mittwoch, 08:00 - 09:30	IZ 358	Christian Rieck	TBA
5	Mittwoch, 13:15 - 14:45	IZ 305	Sören van der Wall	TBA
6	Mittwoch, 13:15 - 14:45	IZ 160	Paul Maximilian Bittner	TBA
7	Mittwoch, 15:00 - 16:30	IZ 358	Micha Horlboge	TBA
8	Mittwoch, 15:00 - 16:30	Container 1	Alexander Hill	TBA
9	Mittwoch, 16:45 - 18:15	IZ 358	Micha Horlboge	TBA
10	Mittwoch, 16:45 - 18:15	PK 3.3	Antje Mönch	TBA
11	Donnerstag, 13:15 - 14:45	IZ 160	Lennart Behme	TBA
12	Donnerstag, 15:00 - 16:30	BW 74.5	Antje Mönch	TBA
13	Donnerstag, 16:45 - 18:15	IZ 160	Julian Troegel	TBA
14	Freitag, 11:30 - 13:00	PK 3.1	Lennart Behme	TBA
15	Freitag, 11:30 - 13:00	IZ 160	Julian Troegel	TBA
16	Freitag, 13:15 - 14:45	IZ 358	Florian Maurer	TBA
17	Freitag, 13:15 - 14:45	IZ 305	Paul Maximilian Bittner	TBA
18	Freitag, 15:00 - 16:30	IZ 358	Florian Maurer	TBA
19	Freitag, 15:00 - 16:30	IZ 305	Alexander Hill	TBA
20	Freitag, 15:00 - 16:30	PK 3.1	Dennis Luck	TBA

Anmeldungen der selben Matrikelnummer, werten wir die neueste. **Bitte markiere alle Termine an, denen du Zeit hast auch als solche, damit am Ende eine faire Gruppenaufteilung möglich ist**

Studienleistung: Mindestens 50 Prozent der Hausaufgaben
Prüfungsleistung: Klausur am Ende des Semesters

Semesterplan: AuD – WS1516

KW	VL - Nr. (Di.+Mi.)	Gr. UE (Do)	Kl. UE (Mi.+Do.+Fr.)	HA Ausgabe (Mi.abends)	HA Abgabe (Mi. bis 13:00)	HA Rückgabe (in kl. UE)
44	0 (ausn. Do.)					
45	1,2	1		HA_0a*		
46	3,4		1	HA_0b*, HA_1		HA_0a*
47	5,6	2				
48	7,8		2	HA_2	HA_1	HA_0b*
49	9,10	3				
50	11,12		3	HA_3	HA_2	HA_1
51	13,14	4				
52	Weihnachtsferien					
53						
1	15,16		4	HA_4	HA_3	HA_2
2	17,18	5				
3	19,20		5	HA_5	HA_4	HA_3
4	21,22	6				
5	23,24		6		HA_5	HA_4
6	25,26	7				HA_5+
*: Mündliche Aufgaben, Bearbeitung in ersten kleinen Übungen; keine Bewertung						
+: Besprechung in großer Übung 7						

Algorithmus

Lösung

Problem

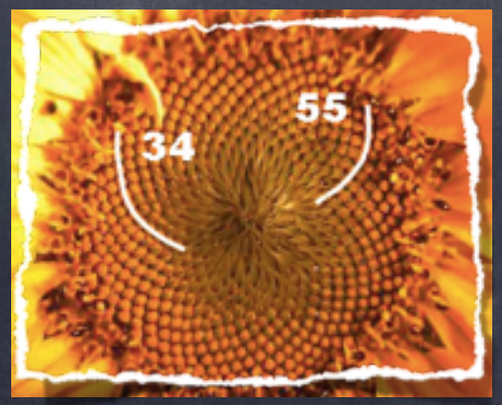
Beispiel: Fibonacci-Zahlen

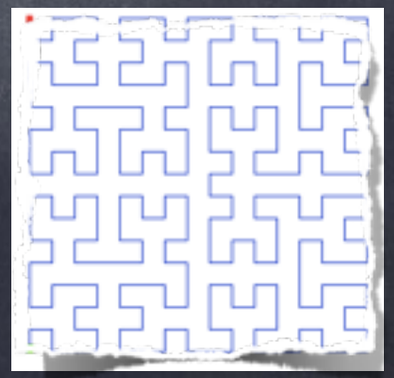
Problem ⇨ Algorithmus ⇨ Lösung

1,1,2,3,5,8,13,...

Leonardo da Pisa („figlio di Bonacci", Sohn des Bonacci)

Strebt gegen den goldenen Schnitt $\lim_{n\to\infty} f(n)/f(n-1)$





$$f(1) = 1,\ f(2) = 1,\ f(n) = f(n-1) + f(n-2) \text{ für } n > 2$$

Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne f(5)

5

Probleminstanz

$$
\begin{aligned}
f(5) &= f(4) + f(3) \\
&= f(3) + f(2) + f(2) + f(1) \\
&= f(2) + f(1) + f(2) + f(2) + f(1) \\
&= 1 + 1 + 1 + 1 + 1 = 5
\end{aligned}
$$

$$f(1) = 1,\ f(2) = 1,\ f(n) = f(n-1) + f(n-2) \text{ für } n > 2$$

Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne f(5) — 5

Probleminstanz

Algorithmus zur Berechnung von f(5):

Gib 5 zurück

$$f(1) = 1,\ f(2) = 1,\ f(n) = f(n-1) + f(n-2) \text{ für } n > 2$$

Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne eine Fibonacci-Zahl

bel. Fib.-Zahl

Algorithmus zur Berechnung bel. Fib. Zahl

~~Gib 5 oder 1 aus~~

Gib 5 aus

Eindeutigkeit

$$f(1) = 1,\ f(2) = 1,\ f(n) = f(n-1) + f(n-2) \text{ für } n > 2$$

8

Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne n-te Fibonacci-Zahl

n-te Fib.-Zahl

Algorithmus zur Berechn. der n-ten Fib.-Zahl

f(1)=1, f(2)=1, f(3)=2, f(4)=3, f(5)=5, f(6)=8, f(7)=13,
f(8)=21, f(9)=34, f(10)=55, f(11)=89, f(12)=144,
f(13)=233, f(14)=377 f(15)=610, f(16)=987, f(17)=1597
f(18)=2584, f(19)=4181, f(20)=6765, f(21)=10946
f(22)=17711, f(23)=28657, f(24)=46368, f(25)=75025,
f(26)=121393, f(27)=196418, f(28)=317811, ...

- ~~Gib n-te Fib. Zahl aus~~

Eindeutigkeit

Endlich

Laufzeit beschreibbar

$$f(1) = 1,\ f(2) = 1,\ f(n) = f(n-1) + f(n-2) \text{ für } n > 2$$

Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne n-te Fibonacci-Zahl

n-te Fib.-Zahl

Algorithmus zur Berechn. der n-ten Fib.-Zahl

- f(1)=1
- f(2)=1
- f(3)=f(2)+f(1)=2
- f(4)=f(3)+f(2)=2+1=3
- f(5)=f(4)+f(3)=3+2=5

...

- f(n)=f(n-1)+f(n-2)=...

n Schritte

Eindeutigkeit

Endlich bzgl. Eingabe (n)

Laufzeit beschreibbar

$$f(1) = 1,\ f(2) = 1,\ f(n) = f(n-1) + f(n-2) \text{ für } n > 2$$

Problem ⇨ Algorithmus ⇨ Lösung

ggT(a,b) | euklidischer Algorithmus | ggT(a,b)

Euklidischer Algorithmus für zwei Zahlen:

falls a gleich 0, gib b zurück

falls a ungleich 0, wiederhole Folgendes
solange b ungleich 0 ist

falls a größer als b, verringere a um b

falls a kleiner oder gleich b,
verringere b um a

gib a zurück

Problem ⇨ Algorithmus ⇨ Lösung

ggT(a,b)

euklidischer Algorithmus

ggT(a,b)

```
int EuklAlgo(int a, int b){
    if(a==0){
        return b;
    }else{
        while(b!=0){
            if(a>b){
                a=a-b;
            }else{
                b=b-a;
            }
        }
    }
}
```

Problem ⇨ Algorithmus ⇨ Lösung

ggT(a,b)

euklidischer Algorithmus

ggT(a,b)

```
EuklAlgo(a,b)
    wenn a=b
    |   return b
    sonst
    |   solange b≠0
    |   |   falls a>b
    |   |    | a=a-b;
    |   |   sonst
    |   |    | b=b-a;
```

Problem ⇨ Algorithmus ⇨ Lösung

ggT(a,b)

euklidischer Algorithmus

ggT(a,b)

```
EuklAlgo(a,b)
1--wenn a=b
2----return b
3--sonst
4----so lange b≠0
5--------falls a>b
6----------a=a-b;
7--------sonst
8----------b=b-a;
```

a=32, b=8

a=24, b=8

a=16, b=8

a=8, b=8

a=8, b=0

Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne n-te Fibonacci-Zahl

n-te Fib.-Zahl

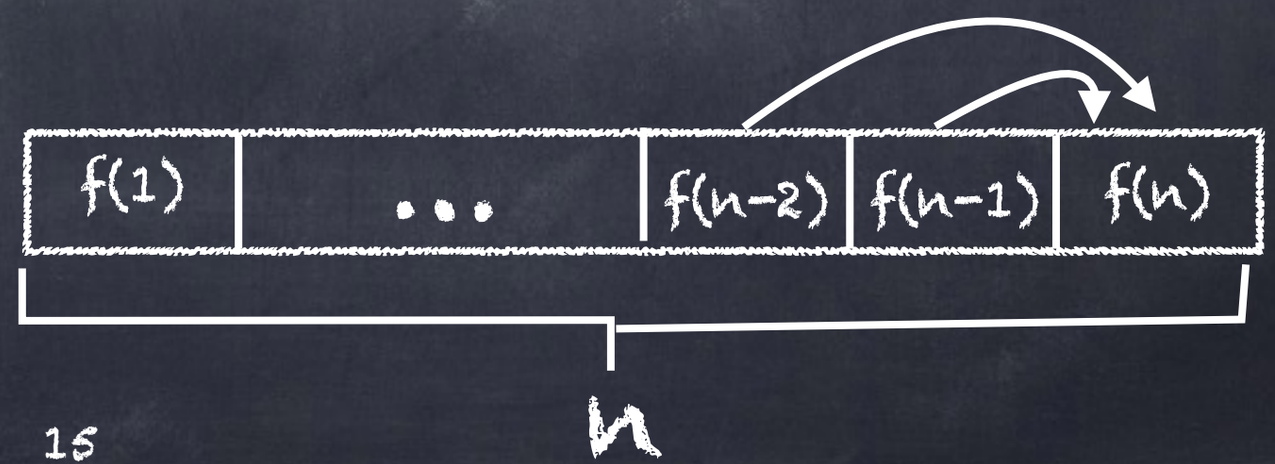
Algorithmus zur Berechn. der n-ten Fib.-Zahl

- $f(1)=1$
- $f(2)=1$
- $f(3)=f(2)+f(1)=2$
- $f(4)=f(3)+f(2)=2+1=3$
- $f(5)=f(4)+f(3)=3+2=5$

...

- $f(n)=f(n-1)+f(n-2)=\ldots$

- Merken von Ergebnissen
- Pro Iteration eine Zahl
- „lineare Komplexität"



Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

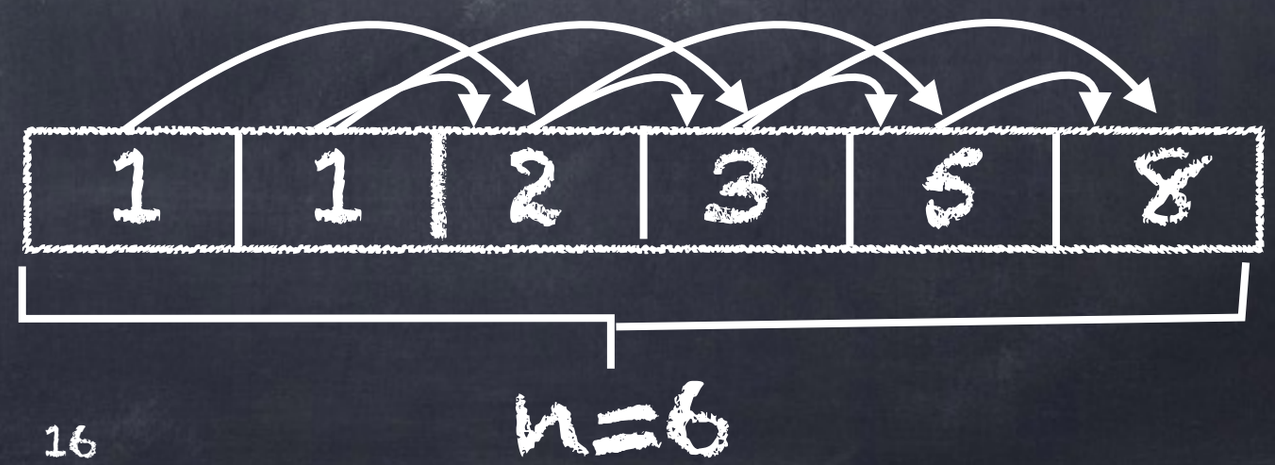
Berechne n-te Fibonacci-Zahl

n-te Fib.-Zahl

Algorithmus zur Berechn. der n-ten Fib.-Zahl

```
fib(n)
1- - - A=<0,...,0>
2- - - z=3
3- - - A[1]=1
4- - -A[2]=1
5- - - so lange z<=n
6- - - -  A[z]=A[z-1]+A[z-2]
7 - - - - z=z+1
8- - -return A[z-1]
```

- Merken von Ergebnissen
- Pro Iteration eine Zahl
- „lineare Komplexität"



Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne n-te Fibonacci-Zahl

n-te Fib.-Zahl

Algorithmus zur Berechn. der n-ten Fib.-Zahl

```
fib(n)
1- - - A=<0,...,0>
2- - - z=3
3- - - A[1]=1
4- - -A[2]=1
5- - - so lange z<=n
6- - - -  A[z]=A[z-1]+A[z-2]
7 - - - - z=z+1
8- - -return A[z-1]
```

Algorithmus zur Berechnung der n-ten Fib.-Zahl
Initialisiere Array der Größe n.
Setze initial globalen Zähler auf eins.
Setze initial ersten Wert des Arrays auf eins.
Setze initial den zweiten Wert des Arrays auf zwei.
Wiederhole die folgenden zwei Schritte, bis
der n-te Wert des Arrays berechnet ist.
Gib den n-ten Wert des Arrays zurück.

Beispiel: Fibonacci-Zahlen

Problem ⇨ Algorithmus ⇨ Lösung

Berechne n-te Fibonacci-Zahl

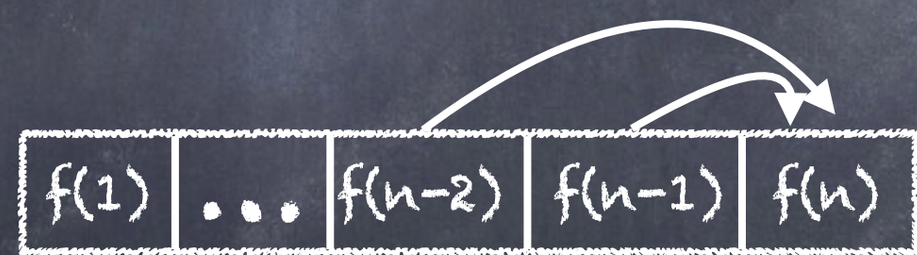
n-te Fib.-Zahl

Algorithmus zur Berechn. der n-ten Fib.-Zahl

Entwurf

$f(1)=1$
$f(2)=1$
$f(n)=F(n-2)+f(n-1)$

⇨



Skizze

⇨

```
fib(n)
1- - - A<0,...,0>
2- - - z=1
3- - - A[1]=1
4- - - A[2]=1
5- - - so lange z<=n
6- - - - - A[z]=A[z-2]+A[z-1]
7- - - - - z=z+1
8- - - return A[z]
```

⇦ ⇦

(form.) Beschreibung

„lineare Komplexität"

Analyse

Pseudo-Code

Beispiel: Fibonacci-Zahlen

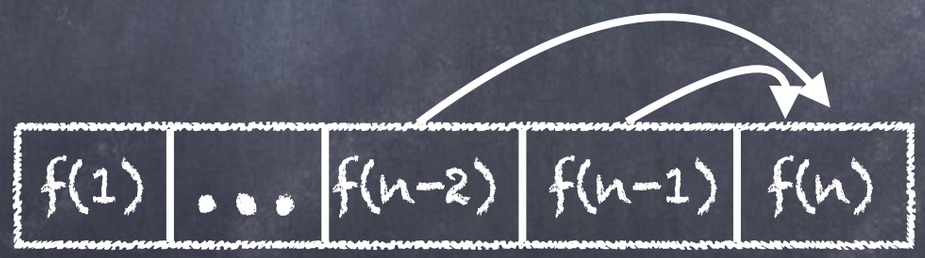
Problem → Algorithmus → Lösung

Algorithm Engineering

$f(1)=1$
$f(2)=1$
$f(n)=F(n-2)+f(n-1)$

Experiment

Implementierung



Skizze

```
fib(n)
1- - - A<0,...,0>
2- - - z=1
3- - - A[1]=1
4- - - A[2]=1
5- - - so lange z<=n
6- - - - - A[z]=A[z-2]+A[z-1]
7- - - - - z=z+1
8- - - return A[z]
```

Pseudo-Code

„lineare Komplexität"

Analyse

Zusammenfassung

- Problem, Algorithmus und Lösung
- Bsp.: Fibonacci-Zahlen, ggt(.,.)
- Eigenschaften von Algorithmen
  - Eindeutigkeit
  - Endlichkeit
- Pseudo-Code
- Entwurf von Algorithmen
- Analyse von Algorithmen

Fragen?