Klausur Informatik II

29. März 2010

Name .....

Matrikelnr. .....

Hinweis

Versehen Sie zunächst das Titelblatt und alle Antwortblätter mit Ihrem Namen und Ihrer Matrikelnummer. Wo angegeben, verwenden Sie bitte die bereits vorgegebenen Skizzen. Sie können die Antwortblätter auf beiden Seiten beschreiben. Sollte Ihnen wider Erwarten der Platz auf einem Blatt nicht ausreichen, so können Sie von der Klausuraufsicht weitere Blätter erhalten, die Sie wiederum mit Ihrem Namen und Ihrer Matrikelnummer kennzeichnen. Geben Sie alle Blätter dieser Klausur wieder ab, da nachgereichte Blätter nicht korrigiert werden können. Heften Sie vor der Abgabe die Blätter mit der Büroklammer zusammen.

Die Notenskala wird erst nach Korrektur der Klausur festgelegt. Sie können daher *nicht* davon ausgehen, dass 20 Punkte zum Bestehen ausreichen.

Aufgabe	**1**	**2**	**3**	**4**	**5**	**6**	$\sum$
Punkte							
Note							

Name: Matrikelnummer:

Aufgabe 1: Wissensfragen *(10 Punkte)*

a) Erläutern Sie in wenigen Sätzen den Zusammenhang zwischen Software-Komponenten und dem Begriff „Kohäsion“! (1 Punkt)

Musterlösung

Eine Komponente soll Aufgaben vereinen, die in einem inneren Zusammenhang stehen (Kohäsion), Aufgaben mit geringer Kohäsion sollen durch unterschiedliche Komponenten realisiert werden.

b) Nennen Sie vier wichtige Dienstqualitäten! (1 Punkt)

Musterlösung

Korrektheit, Robustheit, Aufwand, Skalierbarkeit, Dauerhaftigkeit, Portierbarkeit

c) Erklären Sie den Unterschied zwischen Aggregation und Komposition bei UML in eigenen Worten! (1 Punkt)

Musterlösung

Die Aggregation ist eine Assoziation, die eine „Ist-Teil-von“-Beziehung ausdrückt. Die Komposition schränkt dies weiter auf ein „Kann nicht ohne existieren“ ein.

d) Wie lässt sich die Fehlerwahrscheinlichkeit eines Monte-Carlo-Algorithmus verringern? (0,5 Punkte)

Musterlösung

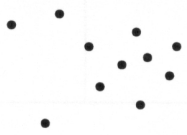
Durch Wiederholungen.

e) Was ist ein Wurzelgraph? (0,5 Punkte)

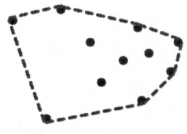
Musterlösung

Ein gerichteter azyklischer Graph mit genau einem Wurzelknoten (Knoten, der keine eingehenden Kanten hat).

f) Zeichnen Sie die konvexe Hülle in folgende Punktmenge ein! (0,5 Punkte)



Musterlösung



Σ

g) Gegeben sei eine Menge von Gegenständen mit den folgenden Gewichten (in Kg): {2, 13, 9, 27, 15, 12}. Ein Rucksack fasse maximal 70 Kilo. Wie schwer wird er, wenn Sie ihn nach dem klassischen Greedy-Algorithmus packen? (1,5 Punkte)

Musterlösung

69 Kg.

h) Nennen Sie die **formale** Definition des o-Kalküls (Klein-o) mit Hilfe der Funktionen $g(n)$ und $f(n)$ und der Konstante c! (2 Punkte)

Musterlösung

$o(g(n)) = \{f(n) \mid \forall c > 0 \; \exists \; n_0 \; > 0 \; \forall n \geq n_0 : 0 \leq \; f(n) < c \cdot g(n)\}$

i) Vervollständigen Sie die folgende Tabelle in Bezug auf die entstehenden Aufwände für die Berechnung von „n!“ (n Fakultät)! (2 Punkte)

Art der Berechnung	Speicher	Zeit
Rekursive: $n! = n \cdot (n-1)!$	$O(\ )$	$O(\ )$
Iterativ: $n! = 1 \cdot 2 \cdot \cdots \cdot n$	$O(\ )$	$O(\ )$

Musterlösung

Art der Berechnung	Speicher	Zeit
Rekursive: $n! = n \cdot (n-1)!$	$O(\ n\ )$	$O(\ n\ )$
Iterativ: $n! = 1 \cdot 2 \cdot \cdots \cdot n$	$O(\ 1\ )$	$O(\ n\ )$

$\sum$

Name: Matrikelnummer:

Aufgabe 2: Rekursion und Iteration *(8 Punkte)*

Die Fibonacci-Folge f ist definiert wie folgt:

$$f_0 = 0 \quad (1)$$
$$f_1 = 1 \quad (2)$$
$$f_n = f_{n-1} + f_{n-2} \quad (3)$$

a) Schreiben Sie eine rekursive Funktion **int** fibo(**int** n), die die n-te Fibonacci-Zahl zurückgibt. *(3 Punkte)*

Musterlösung

```
int fibo(int n) {
  if (n<=1) {
    return n;
  } else {
    return (fibo(n-1) + fibo(n-2));
  }
}
```

b) Führen Sie nun die rekursive Funktion in eine iterative Funktion über. Verwenden Sie dazu *nicht* die geschlossene Darstellung der Fibonacci-Folge, auch wenn sie Ihnen bekannt ist. *(5 Punkte)*

Musterlösung

```
int fibo2(int n) {
  if (n <= 1) return n;

  int i = 2;
  int a = 0;
  int b = 1;
  int neu = 0;

  while (i <= n) {
    neu = a + b;
    a = b;
    b = neu;
    i += 1;
  }
  return neu;
}
```


$\sum$

Aufgabe 3: Rot-Schwarz Baum *(10 Punkte)*

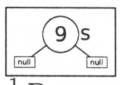
a) Welche fünf Eigenschaften gelten für jeden Rot-Schwarz-Baum? *(2,5 Punkte)*

Musterlösung

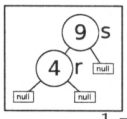
1. Jeder Knoten ist entweder rot oder schwarz
2. Die Wurzel ist schwarz
3. Jedes Blatt (Null) ist schwarz
4. Wenn ein Knoten rot ist, sind beide Kinder schwarz
5. Für jeden Knoten gilt, dass alle Pfade vom Knoten zu einem Blatt dieselbe Anzahl schwarzer Knoten beinhalten

b) Fügen Sie in einen leeren Rot-Schwarz-Baum die Zahlen 9, 4, 7, 6, 5 in dieser Reihenfolge ein. Beschriften Sie die Knoten entsprechend ihrer Farben: Ein r neben einem Knoten bezeichnet einen roten, ein s einen schwarzen Knoten. Zeichnen Sie den Baum auf diese Weise nach jedem Einfügeschritt. *(4 Punkte)*

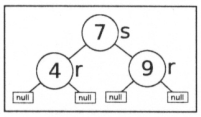
Musterlösung



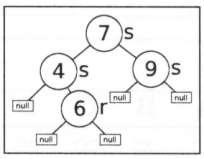
$\frac{1}{2}P$,



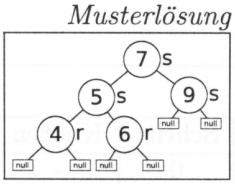
$\frac{1}{2}P$,



$1P$,



$1P$,



$1P$

c) Beweisen Sie: Jeder durch einen Knoten x aufgespannte Teilbaum eines Rot-Schwarz-Baums hat wenigstens $2^{bh(x)} - 1$ innere Knoten.
Hierbei bezeichnet $bh(k)$ die Black-Height eines Knotens k, also die Anzahl schwarzer Knoten auf einem beliebigen Pfad von k zu einem Blatt, k nicht mitgerechnet. *(3,5 Punkte)*

Musterlösung

Beweis per vollständiger Induktion:

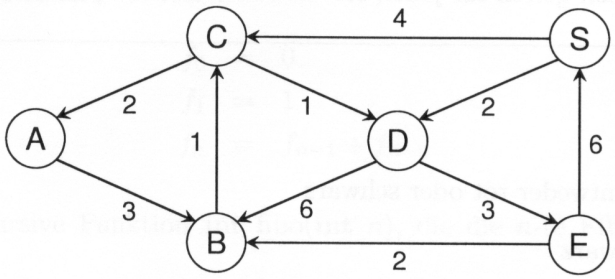
Induktions-Anfang:
$h = 0 \Rightarrow x$ ist Blatt. Teilbaum enthält $2^0 - 1 = 0$ innere Knoten.

Induktions-Schritt:
Sei x ein innerer Knoten mit Höhe $h + 1$ und zwei Kindern:
$\Rightarrow$ jedes Kind hat $bh(x)$ oder $bh(x) - 1$, je nachdem ob es rot oder schwarz ist
$\Rightarrow$ jedes Kind hat mindestens $2^{bh(x)-1} - 1$ innere Knoten (I-Annahme)
$\Rightarrow$ Der von x aufgespannte Teilbaum hat mindestens
$2^{bh(x)-1} - 1 + 2^{bh(x)-1} - 1 + 1 = 2^{bh(x)} - 1$ innere Knoten, q.e.d

$\sum$

Name: Matrikelnummer:

Aufgabe 4: Dijkstra *(12 Punkte)*



a) Verwenden Sie den Dijkstra-Algorithmus, um in dem Graphen die kürzesten Wege von Knoten S zu allen anderen Knoten zu finden. Verwenden Sie dafür die angegebene Tabelle. Geben Sie nach jeder Iteration des Algorithmus neben dem aktuell besuchten Knoten jeweils alle momentan kürzesten Wege von S zu allen Knoten an. *(5 Punkte)*

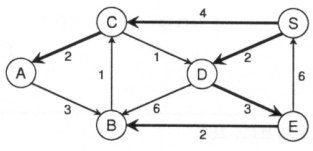
Schritt	Knoten	S	A	B	C	D	E
0							
1							
2							
3							
4							
5							
6							

Musterlösung

Schritt	Knoten	S	A	B	C	D	E
0	-	0	∞	∞	∞	∞	∞
1	S	0	∞	∞	4	2	∞
2	D	0	∞	8	4	2	5
3	C	0	6	8	4	2	5
4	E	0	6	7	4	2	5
5	A	0	6	7	4	2	5
6	B	0	6	7	4	2	5

b) Zeichnen Sie den aus der Anwendung des Dijkstra-Algorithmus resultierenden Spannbaum in den dargestellten Graphen ein. *(3 Punkte)*

Musterlösung



c) Handelt es sich bei dem durch den Dijkstra-Algorithmus ermittelten Spannbaum um den minimalen Spannbaum für diesen Graphen? Begründen Sie Ihre Antwort. *(2 Punkte)*

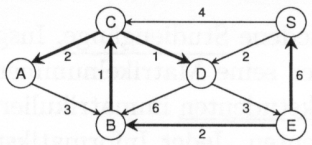
Musterlösung

Nein, der ermittelte Spannbaum ist nicht der minimale Spannbaum. Begründung durch Gegenbeispiel eines minimaleren Spannbaums als der ermittelte (nicht notwendigerweise der minimale

$\sum$

Spannbaum!). Der Spannbaum definiert sich dabei als „zusammenhängender Teilgraph, der alle Knoten enthält, und die Summe der Kantengewichte ist minimal" Böhm 2008.

Beispiel fuer einen minimaleren Spannbaum:



d) Funktioniert der Dijkstra-Algorithmus auch bei Vorhandensein von negativen Kantengewichten? Begründen Sie Ihre Antwort. *(2 Punkte)*

Musterlösung

Nein, Dijkstra funktioniert nicht mit negativen Kantengewichten. Begründung: Eine Kante mit negativem Gewicht kann die Distanz eines bereits bearbeiteten Knotens nachträglich verbessern. Dadurch wird die Invariante des Dijkstra-Algorithmus ungültig. (Weiteres Gegenbeispiel: Zyklus mit negativem Kantengewicht).

$\sum$

Aufgabe 5: UML *(10 Punkte)*

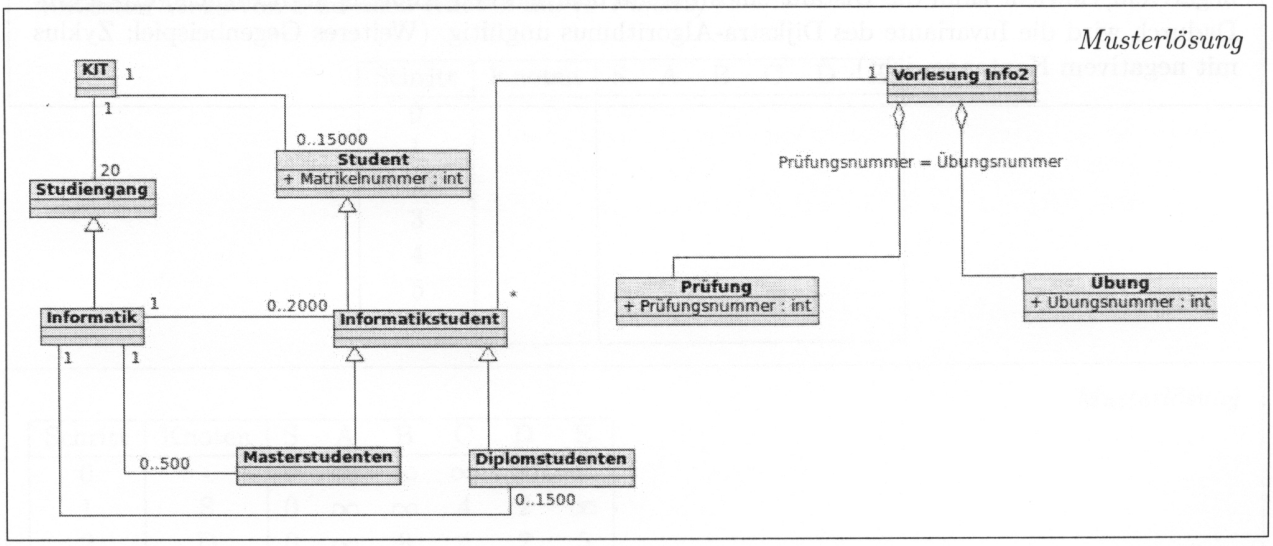
Das KIT beauftragt Sie, einen Teil des Modulhandbuchs zu modellieren. Sie stellen fest:

Am KIT existieren 20 verschiedene Studiengänge. Insgesamt studieren am KIT 15000 Studenten. Jeder Student ist eindeutig über seine Matrikelnummer zu identifizieren. Im Studiengang Informatik können bis zu 2000 Informatikstudenten immatrikuliert sein, aufgeteilt in bis zu 500 Masterstudenten und bis zu 1500 Diplomstudenten. Jeder Informatikstudent muss die Vorlesung Info2 belegen. Diese Vorlesung besteht aus einer Übung und einer Prüfung, wobei Übung und Prüfung über dieselbe Identifikationsnummer verfügen.

a) Modellieren Sie die Zusammenhänge als **Klassendiagramm**. Verwenden Sie sinnvolle Klassennamen und tragen Sie die entsprechenden Kardinalitäten und Assoziationen ein. *(10 Punkte)*

Hinweis: Erstellen Sie nur Klassen aus den unterstrichenen Substantiven, fügen Sie keine weiteren Klassen hinzu.

Musterlösung



Σ

Name: Matrikelnummer:

Aufgabe 6: Schleifeninvariante, wp-Kalkül *(10 Punkte)*

Gegeben sei folgendes Code-Fragment:

```
1  public int nueber2(int n) {
2      // Q : n ≥ 0
3      int a = 0;
4      int result = 0;
5      int i = 1;
6      while (i ≤ n) {
7          a = i − 1;
8          result = result + a;
9          i = i + 1;
10     }
11     return result;
12     // R : result = (n)·(n−1) / 2
13 }
```

a) Bestimmen Sie eine Schleifen-Invariante P für die **while**-Schleife, die geeignet ist, unter der Vorbedingung Q die Nachbedingung R herzuleiten. *(2 Punkte)*

Musterlösung

$P : result = \frac{(i)\cdot(i-1)}{2} \wedge (i \leq n+1)$ oder
$P' : result = \sum_{k=0}^{i-1} k \wedge (i \leq n+1)$ oder
$P'' : result = \binom{i}{2} \wedge (i \leq n+1)$

Gegeben sei folgendes Pseudocode-Fragment C:

```
1 if ((x ≠ 0) ∧ (y ≠ 0)) {
2     if (x < y) {
3         r = y/x · (y − x);
4     } else {
5         r = x/y · (x − y);
6     }
7 } else {
8     r = 0;
9 }
```

b) Bestimmen Sie $\mathbf{wp}(C, r < 0)$ und vereinfachen Sie das Ergebnis. Gehen Sie davon aus, dass es sich bei x, y und r um Gleitkommazahlen mit unendlicher Genauigkeit handelt. *(8 Punkte)*

Musterlösung

$\mathbf{wp}(C, r < 0)$
$\Leftrightarrow ((x \neq 0) \wedge (y \neq 0) \wedge$
$[\underbrace{(x < y \wedge \mathbf{wp}(\text{“}r = \frac{y}{x} \cdot (y-x)\text{”}, r < 0))}_{(1)} \vee \underbrace{(x \geq y \wedge \mathbf{wp}(\text{“}r = \frac{x}{y} \cdot (x-y)\text{”}, r < 0))}_{(2)}]$
$\vee (\neg((x \neq 0) \wedge (y \neq 0)) \wedge \underbrace{\mathbf{wp}(\text{“}r = 0\text{”}, r < 0)}_{=false})$

zu(1) :
$\Leftrightarrow (x < y) \wedge \frac{y}{x} \cdot \underbrace{(y-x)}_{>0} < 0$

$\sum$

$\Leftrightarrow (x < y) \wedge \frac{y}{x} < 0$

$\Leftrightarrow \underbrace{\cancel{(x<y)}}_{\text{in (1b) enthalten}} \wedge [\underbrace{(x > 0 \wedge y < 0)}_{=false,\text{ da } (x<y)} \vee \underbrace{(x < 0 \wedge y > 0)}_{=:(1b)}]$

$\Leftrightarrow (y > 0 \wedge x < 0)$

zu(2) : (analog)

$\Leftrightarrow (x \geq y) \wedge \frac{x}{y} \cdot \underbrace{(x - y)}_{>0} < 0$

$\Leftrightarrow (x \geq y) \wedge \frac{x}{y} < 0$

$\Leftrightarrow \underbrace{\cancel{(x \geq y)}}_{\text{in (2b) enthalten}} \wedge [\underbrace{(x > 0 \wedge y < 0)}_{=:(2b)} \vee \underbrace{(x < 0 \wedge y > 0)}_{=false,\text{ da } (x \geq y)}]$

$\Leftrightarrow (y < 0 \wedge x > 0)$

mit (1) und (2):

$\Leftrightarrow (x \neq 0) \wedge (y \neq 0) \wedge [(x < 0 \wedge y > 0) \vee (x > 0 \wedge y < 0)]$

$\Leftrightarrow (x < 0 \wedge y > 0) \vee (x > 0 \wedge y < 0) =: \mathbf{wp}(C, r < 0)$

Korrekturhinweise:

3 Punkte für die richtige Anwendung der if-then-else-Regel

2 Punkte für die richtige Auswertung von (1) bzw. (2) (jeweils 1 Punkt)

1 Punkt für die Erkenntnis, dass der äußere else-Zweig false ist

2 Punkt für das richtige, vereinfachte Ergebnis (Rechenweg irrelevant!), falls nicht vereinfacht, nur 1 Punkt!

$\sum$