| Prüfungsteilnehmer | Prüfungstermin | Einzelprüfungsnummer |
|---|---|---|

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

**Herbst**

**2004**

**66113**

# Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

## - Prüfungsaufgaben -

Fach: **Informatik (vertieft studiert)**

Einzelprüfung: **Rechnerarchitektur, Datenb., Betriebssys.**

Anzahl der gestellten Themen (Aufgaben): 2

Anzahl der Druckseiten dieser Vorlage: 15

Bitte wenden!

## Thema Nr. 1

**Sämtliche Teilaufgaben sind zu bearbeiten!**

### Aufgabe 1: Normalformen

Gegeben sei die folgende relationale Datenbank der Mietwagenfirma „MobilRent", Station „München-Mitte":

| *Mobil-Rent* | *KdNr* | *Name* | *Wohnort* | *Buchungs-datum* | *Aktion* | *Fahr-zeug* | *Typ* | *Tarif-gruppe* | *Tage* | *Rückgabe-station* | *Stations-leiter* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 123 | Chomsky | Nürnberg | 23.01.04 | 0 | Micra | Klein | 1 | 2 | Nürnberg-Süd | Backus |
| | 123 | Chomsky | Nürnberg | 07.10.03 | - 25% | Sprinter | Transp | 5 | 1 | Nürnberg-Nord | Hoare |
| | 220 | Neumann | München | 02.04.04 | 0 | Lupo | Klein | 1 | 2 | München-Mitte | Zuse |
| | 710 | Turing | München | 20.02.04 | - 10% | Micra | Klein | 1 | 2 | München-Mitte | Zuse |
| | 888 | Neumann | Passau | 07.10.03 | - 25% | A3 | Mittel-klasse | 3 | 5 | München-Mitte | Zuse |

„KdNr" steht für die Kundennummer der Kunden. An bestimmten Tagen gewährt die Firma Rabatt. Folgende funktionale Abhängigkeiten seien vorgegeben:

KdNr → Name, Wohnort
KdNr, Buchungsdatum → Fahrzeug, Typ, Tarifgruppe, Tage, Rückgabestation, Stationsleiter
Buchungsdatum → Aktion
Fahrzeug → Typ, Tarifgruppe
Typ → Tarifgruppe
Rückgabestation → Stationsleiter

1. Erklären Sie kurz den Unterschied zwischen „Schlüsselkandidat" und „Superschlüssel".

2. Erklären Sie, warum nur (KdNr, Buchungsdatum) als Schlüsselkandidat in Frage kommt.

3. Begründen Sie, warum nur Relationen mit einem zusammengesetzten Schlüsselkandidaten die 2. Normalform verletzen können.

4. Erläutern Sie, inwiefern obiges Schema die zweite bzw. dritte Normalform verletzt. Zeigen Sie anhand obiger Relation „MobilRent" mögliche Anomalien auf, die bei fehlender Normalisierung auftreten können.

5. Begründen Sie, dass obiges Schema in erster Normalform ist und überführen Sie es in die zweite, aber nicht in die dritte Normalform. Erläutern Sie die dazu durchzuführenden Schritte jeweils kurz. Zeigen Sie, inwiefern die entstandene Relation die dritte Normalform verletzt.

6. Überführen Sie das Schema, ebenfalls mit kurzer Erläuterung, nun in die dritte Normalform.

Fortsetzung nächste Seite!

## Aufgabe 2: E-R-Modellierung

Für ein Transportunternehmen soll eine Datenbank für folgendes Szenario entwickelt werden:

Die Firma verfügt über mehrere Abteilungen, die von jeweils einem Mitarbeiter geleitet werden, und jeder Mitarbeiter, von dem Name, Wohnort und Gehaltsstufe abgespeichert werden sollen, ist einer dieser Abteilungen eindeutig zugeordnet. Der Betrieb besitzt mehrere Transporter verschiedener Typen. Diese unterscheiden sich durch Größe, maximale Nutzlast und benötigte Führerscheinklasse. Die einzelnen Fahrzeuge werden fortlaufend nummeriert und es soll das Baujahr sowie der Termin zur nächsten Inspektion abgespeichert werden. Neben den Verwaltungsangestellten (für Buchhaltung, Personal, Kundenbetreuung etc.) und den hauseigenen KFZ-Meistern (zur LKW-Wartung) beschäftigt das Logistikunternehmen natürlich insbesondere Fahrer. Bucht ein Kunde eine Tour, so werden für diese wenigstens ein Fahrer und ein Transporter eingesetzt. Von der gebuchten Tour müssen folgende Daten abrufbar sein: Buchungsdatum, Termin und die Anzahl der benötigten LKWs. Von den Kunden müssen Name, Wohnort und die zugehörige Firma vorliegen.

Erstellen Sie ein E-R-Diagramm. Verarbeiten Sie dabei nur die unbedingt notwendigen Informationen, geben Sie aber an, wie die nicht im E-R-Modell auftauchenden Informationen bestimmt werden können. Legen Sie auch die Primärschlüssel fest und begründen Sie Ihre Entscheidung, falls Sie zusätzliche künstliche Schlüssel einfügen. Geben Sie die Funktionalitäten an.

## Aufgabe 3: Threads

1. Besitzt jeder Thread einen eigenen Kellerspeicher? Erläutern Sie Ihre Antwort.

2. Besitzt jeder Thread einen eigenen Adressraum? Erläutern Sie Ihre Antwort.

3. Erläutern Sie die Realisierung von Threads im Benutzer-Adressraum.

4. Erläutern Sie die Realisierung von Threads im System-Adressraum.

5. Threads in Java:
   Im Folgenden soll der Betrieb eines Parkhauses teilweise simuliert werden. Ein Objekt der Klasse `Parkhaus` repräsentiert das Parkhaus. Jedes Fahrzeug wird als eigener Thread simuliert.

Folgende Klassen seien gegeben:

```
public class ParkhausTest {
      public static void main(String[] args) {
            Parkhaus p = new Parkhaus(50);

            /* Hier 100 Threads mit Hilfe der Klasse Fahrzeug und des
            Objekts p erzeugen und starten (siehe Teilaufgabe a) */
      }
}
```

Fortsetzung nächste Seite!

```
public class Parkhaus {
      private int kapazitaet;       // Kapazitaet des Parkhauses
      private int anzFahrzeuge;     // Anzahl der parkenden Fahrzeuge

      public Parkhaus(int kapazitaet) {
            this.kapazitaet = kapazitaet;
            anzFahrzeuge = 0;
      }

      // Einfahrt (Methode noch zu ergaenzen, siehe Teilaufgabe b)
      public void einfahrt() {
            anzFahrzeuge++;

      }

      // Ausfahrt (Methode noch zu ergaenzen, siehe Teilaufgabe b)
      public void ausfahrt() {
            anzFahrzeuge--;

      }
}

public class Fahrzeug implements Runnable {
      private Parkhaus p;

      public Fahrzeug(Parkhaus p) {
            this.p = p;
      }

      public void run() {

            // In Parkhaus einfahren
            p.einfahrt();

            // Aufenthalt
            try {
                  Thread.sleep((int)(Math.random() * 10000));
            } catch (InterruptedException e) {}

            // Aus Parkhaus ausfahren
            p.ausfahrt();
      }
}
```

a) Ergänzen Sie die Klasse `ParkhausTest` so, dass mit Hilfe der Klasse `Fahrzeug` und des Objekts `p` einhundert Threads erzeugt und gestartet werden.

b) Ergänzen Sie die Methoden `einfahrt ()` und `ausfahrt ()` der Klasse `Parkhaus` so, dass der Zugriff auf `anzFahrzeuge` unter wechselseitigem Ausschluss geschieht! Ferner soll folgendes Szenario simuliert werden: Wenn die Kapazität des Parkhauses erschöpft ist, warten einfahrbereite Fahrzeuge in einer Wartezone. Immer wenn ein Fahrzeug das Parkhaus verlassen hat, verlässt ein beliebiges wartendes Fahrzeug die Wartezone.

Fortsetzung nächste Seite!

**Aufgabe 4: Semaphore**

Vor dem Zugang zu den Flugsteigen eines Flugplatzes findet in einer *Sicherheitszone* eine Sicherheitsüberprüfung der Passagiere statt. In der Sicherheitszone dürfen sich immer nur maximal 10 Passagiere aufhalten.

Die Sicherheitszone enthält einen *Kontrollbereich* mit einem *Gepäckkontrolleur* und einem *Personenkontrolleur*. Um jedes Gepäckstück eindeutig einem Passagier zuordnen zu können, ist im Kontrollbereich nur maximal ein Passagier erlaubt.

Der Gepäckkontrolleur darf erst dann mit seiner Kontrolle beginnen, wenn ein Passagier den Kontrollbereich betreten hat. Erst nachdem der Gepäckkontrolleur dem Passagier das Gepäckstück abgenommen hat, darf der Personenkontrolleur mit der Kontrolle beginnen. Der Gepäckkontrolleur darf dem Passagier das Gepäckstück erst dann wieder zurückgeben, wenn der Personenkontrolleur seine Kontrolle beendet hat.

Nur wenn eine komplette Kontrolle (Person und Gepäck) durchgeführt wurde, darf der Passagier den Kontrollbereich verlassen.

Folgende Prozesse seien definiert:

```
Passagier {
        <betritt Sicherheitszone>
        <betritt Kontrollbereich>
        <verlässt Kontrollbereich>
        <verlässt Sicherheitszone>
}


Gepäckkontrolleur {
        <nimmt Passagier Gepäckstück ab>
        <kontrolliert Gepäckstück>
        <gibt Passagier Gepäckstück zurück>
}


Personenkontrolleur {
        <kontrolliert Passagier>
}
```

Stellen Sie mit Hilfe von Semaphoren den oben skizzierten Ablauf sicher. Geben Sie dazu die Semaphore mit einem kurzen aussagekräftigen Kommentar an. Geben Sie auch zu jedem Semaphor die Art (boolesch oder ganzzahlig) sowie die Anfangsbelegung an.

Fortsetzung nächste Seite!

**Aufgabe 5: Maschinennahe Programmierung**

Gegeben sei eine einfache Maschine mit folgenden Eigenschaften:

- Es stehen 8 Register RO-R7 mit je 4 Byte Breite zur Verfügung.
- Das einzige Datenformat ist eine 4 Byte lange Ganzzahl.
- In einem Programm werden Speicheradressen nur per Marken angegeben. Als Marken können beliebige Zeichenreihen, jedoch keine Zahlen, verwendet werden.
- Befehlssatz:

| Befehl | Wirkung |
|---|---|
| MOVE R*m*, R*n* | R*n* := R*m* |
| MOVE *int*, R*n* | R*n* := *int* |
| MOVE *marke*, R*n* | R*n* := [*marke*] |
| MOVE R*m*, *marke* | [*marke*] := R*m* |
| ADD *op1*, *op2*, R*n* | R*n* := *op1* + *op2* |
| SUB *op1*, *op2*, R*n* | R*n* := *op1* – *op2* |
| MULT *op1*, *op2*, R*n* | R*n* := *op1* * *op2* |
| DIV *op1*, *op2*, R*n* | R*n* := *op1* div *op2* (Ganzzahldivision) |
| JMP *marke* | unbedingter Befehlssprung zu Adresse *marke* |
| JEQ R*m*, R*n*, *marke* | Befehlssprung zu *marke*, falls R*m* = R*n* |
| JGT R*m*, R*n*, *marke* | Befehlssprung zu *marke*, falls R*m* > R*n* |
| JLT R*m*, R*n*, *marke* | Befehlssprung zu *marke*, falls R*m* < R*n* |

Dabei gelten folgende Vereinbarungen:

- *m, n* ∈ {0,1,...,7}
- *marke* ist eine Speicheradresse, deren Inhalt mit [*marke*] angegeben wird.
- *int* ist eine ganze Zahl.
- Für *op1* und *op2* können folgende Operanden eingesetzt werden:
  - ein beliebiges Register R*m*, *m* ∈ {0,1,...,7}
  - eine ganze Zahl *int* als unmittelbarer Operand

- Jede Zeile eines Programms enthält genau einen Befehl und evtl. einen Kommentar, der durch zwei Schrägstriche (//) gekennzeichnet wird.

- Jede Zeile kann am Anfang durch eine Marke gefolgt von einem Doppelpunkt gekennzeichnet werden, um als Sprungziel zu dienen.

- Beispiel für ein Programm:

```
anfang: MOVE 10, RO            // RO := 10
        MOVE RO, R1            // R1 := RO
        SUB R1, 5, R1          // R1 := R1 - 5
        JLT RO, R1, anfang     // Befehlssprung zu anfang, falls RO<R1
```

Fortsetzung nächste Seite!

1. Setzen Sie folgende hochsprachliche Anweisungen in ein Programm der gegebenen Maschine um. Sie können dabei davon ausgehen, dass die hochsprachlichen Variablen an gleichnamigen Marken zur Verfügung stehen.

   a) `i = i * a`
   b) `a = m MODULO n` (d.h. `a` wird der Rest der ganzzahligen Division von `m` durch `n` zugewiesen; es gilt: `m ≥ 0, n > 0`)

2. Setzen Sie folgende hochsprachliche Routine zur Berechnung der ganzzahligen Wurzel einer ganzen Zahl `a > 1` in ein Programm der gegebenen Maschine um. Der Wert der Variablen `a` steht an der Marke `a` zur Verfügung. Der Wert der Variablen `erg` soll an der Marke `erg` abgespeichert werden. Verwenden Sie Register zur Umsetzung der übrigen Variablen.

```
ug = 0; og = a;                 // Intervall initialisieren

while (ug < og-1) {             // solange bis Intervalllänge 1 ist
  m = ug + ((og-ug) div 2);     // Mitte des Intervalls bilden
  mm = m*m;                     // Quadrat der Mitte bilden

  if (mm == a) {                // Prüfen, ob Quadrat gleich a
      ug = m;                   // Intervall so setzen, dass Abbruch-
      og = m+1;                 // bedingung erfüllt ist
  } else if (mm < a) {          // Wenn Quadrat kleiner als a,
      ug = m;                   // dann Intervall unten anpassen
  } else {                      // Wenn Quadrat größer als a,
      og = m;                   // dann Intervall oben anpassen
  }
}

erg = ug;                       // Intervalluntergrenze ist Ergebnis
```

## Aufgabe 6: Rechnernetze

1. MAC-Teilschicht:
   Beschreiben Sie das CSMA/CD-Verfahren.

2. Transportschicht:
   Nennen Sie Unterschiede zwischen TCP und UDP.

## Thema Nr. 2

**Sämtliche Teilaufgaben sind zu bearbeiten!**

### Aufgabe 1: Routing

Im Internet werden die beiden Interior Gateway Protokolle (IGP) Distance Vector und Link State verwendet. In dieser Aufgabe sollen Sie die unterschiedlichen Vorgehensweisen der beiden Routingprotokolle anhand eines Beispiels nachvollziehen und erklären. Gegeben sei das in Abbildung 1 dargestellte IP Core Netzwerk. Es besteht aus einer Reihe von Routern, die miteinander verbunden sind. Die Verbindungskanten geben als Parameter jeweils die Kosten der einzelnen Verbindungen an. Diese gelten in beide Richtungen.

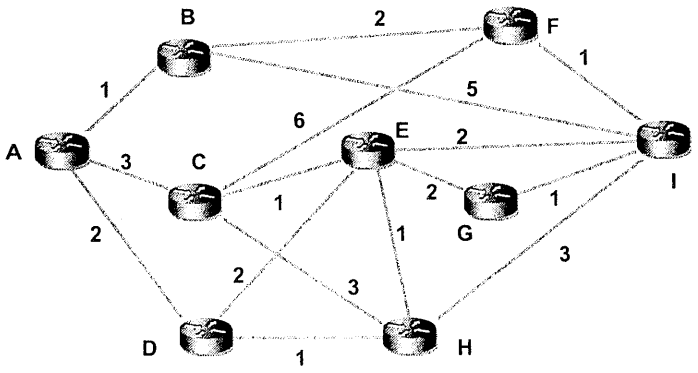



Abbildung 1: Beispielnetz

1. Erstellen Sie für das in Abbildung l dargestellte Beispielnetz die Distanzvektor-Tabelle im Router C für die ersten Austauschphasen, bis sich die Distanzvektor-Tabelle des Routers C nicht mehr ändert. Erläutern Sie dabei Ihre Vorgehensweise und erklären Sie weshalb sich die Tabellen nach einer bestimmten Anzahl von Informationabgleichen nicht mehr ändern.

2. Nehmen Sie nun an, dass die Verbindung zwischen den beiden Routern C und E ausfällt. Berechnen Sie die neue Distanzvektor-Tabelle nach den nächsten Austauschschritten in Router C.

3. Verwenden Sie nun den Link State (Dijkstra) Algorithmus, um die Routing Tabelle des Routers C aufzubauen. Dabei ist wie in Teilaufgabe l von einem Netz ohne Ausfälle auszugehen. Dokumentieren Sie dabei jeweils die Ergebnisse nach jedem Schritt in der Form einer Tabelle der bestätigten, betrachteten und verworfenen Abstände.

Fortsetzung nächste Seite!

**Aufgabe 2: Verständnisfragen**

In dieser Aufgabe werden Verständnisfragen zu unterschiedlichen Themen gestellt. Bitte halten Sie Ihre Antworten kurz.

1. Erläutern Sie die Unterschiede zwischen Simplex-, Halbduplex-, und Vollduplex-Übertragung. Beurteilen Sie die Verfahren nach ihrem Durchsatz unter der Voraussetzung, dass alle Mechanismen die gleiche Zeichenrate zur Leitungscodierung verwenden.

   Welche Vermittlungsverfahren kennen Sie? Erläutern Sie die Unterschiede zwischen den Verfahren.

   Welche Netzstrukturen kennen Sie? Geben Sie die drei Basistopologien an und erläutern Sie die Unterschiede zwischen den Strukturen.

2. Was versteht man unter den Begriffen "(Repeating) Hub", "Switch (-ing Hub)", "Switchrouter" und "Router"?

   Zeichnen Sie das ISO/OSI Schichtenmodell und erklären Sie stichpunktartig die Aufgaben der einzelnen Schichten.

   Was versteht man unter dem Begriff CIDR? Wozu wird es eingesetzt und warum wurde es eingeführt?

   Skizzieren Sie den Aufbau eines GSM Kanals und erläutern Sie die dabei eingesetzten Multiplextechniken.

Fortsetzung nächste Seite!

**Aufgabe 3: Datenbankanfrage in SQL**

Wir betrachten die Datenbank UNIVERSITÄT mit den folgenden Beispieltabellen:

| STUDENT | | | |
|---|---|---|---|
| Matnr | Name | Semester | Hauptfach |
| 147000 | Schmidt | 1 | Informatik |
| 148000 | Maier | 2 | Informatik |
| ... | ... | ... | ... |

| VORLESUNG | | | |
|---|---|---|---|
| Vnr | Name | SWS | Fach |
| 10640 | Einführung in die Informatik | 4 | Informatik |
| 10650 | Datenstrukturen | 4 | Informatik |
| 10410 | Diskrete Mathematik | 3 | Mathematik |
| 10780 | Datenbanken | 3 | Informatik |
| ... | ... | ... | ... |

| VERANSTALTUNG | | |
|---|---|---|
| Vnr | Semester | Dozent |
| 10650 | WS 0102 | Tarjan |
| 10650 | SS 2002 | Mehlhorn |
| 10780 | WS 0102 | Ullman |
| ... | ... | ... |

| NOTENVERTEILUNG | | | |
|---|---|---|---|
| Vnr | Semester | Matnr | Note |
| 10650 | SS 2002 | 147000 | 3 |
| 10650 | SS 2002 | 148000 | 1 |
| 10780 | WS 0102 | 148000 | 2 |
| ... | ... | ... | ... |

| VORAUSSETZUNG | |
|---|---|
| hatVoraus | istVoraus |
| 10650 | 10410 |
| 10650 | 10640 |
| 10780 | 10650 |
| ... | ... |

Erstellen Sie in SQL folgende Anfragen:

a) Bestimmen Sie alle Vorlesungen (Vnr, Name), die der Dozent „Ullman“ gehalten hat.

b) Bestimmen Sie alle Studierenden (Matnr, Name), die in der Vorlesung ‚10780’ eine Note erhalten haben.

c) Bestimmen Sie die Anzahl der Vorlesungen im Fach ‚Informatik’.

d) Bestimmen Sie alle Voraussetzungen (Vnr, Name) der Vorlesung ‚Datenstrukturen’.

e) Bestimmen Sie für alle Studierenden (Matnr, Name) die Durchschnittsnoten über alle ihre gehörten Vorlesungen.

Fortsetzung nächste Seite!

## Aufgabe 4: Datenmodellierung und Datenbankaufbau

Wir betrachten die Datenbank UNIVERSITÄT aus Aufgabe 3

a) Definieren Sie die Relationenschemas von UNIVERSITÄT in SQL, und geben Sie dabei geeignete Schlüssel und Fremdschlüssel an.

b) Fügen Sie in SQL in jede der Relationen

- STUDENT,
- VERANSTALTUNG und
- NOTENVERTEILUNG

aus Aufgabe 3 jeweils mindestens ein weiteres geeignetes Tupel ein, so dass die natürlichen Schlüssel- und Fremdschlüsselbedingungen aus Teil a) erfüllt sind.

c) Geben Sie ein ER-Diagramm (mit Funktionalitätsbedingungen) für das Datenbankschema aus Aufgabe 3 an, in dem möglichst viele Relationen als Relationships modelliert sind.

## Aufgabe 5: Funktionale Abhängigkeiten

Gegeben sei folgende Menge $F$ von funktionalen Abhängigkeiten:

$$F = \{BC \rightarrow A,\ AC \rightarrow DE,\ A \rightarrow D\}.$$

a) Geben Sie zwei unterschiedliche Ableitungen für die funktionale Abhängigkeit $BC \rightarrow D$ an.

b) Bestimmen Sie die Attributhülle von $BC$ unter $F$.

c) Bestimmen Sie ein minimales System $G$ für $F$. Bestimmen Sie mit dem Synthese-Algorithmus aus $G$ eine 3NF-Zerlegung des Relationenschemas $R$ mit der Attributmenge $U = ABC\,DE$ und der Menge $F$ von funktionalen Abhängigkeiten.

Fortsetzung nächste Seite!

**Aufgabe 6: Betriebsmittel-Zuteilung**

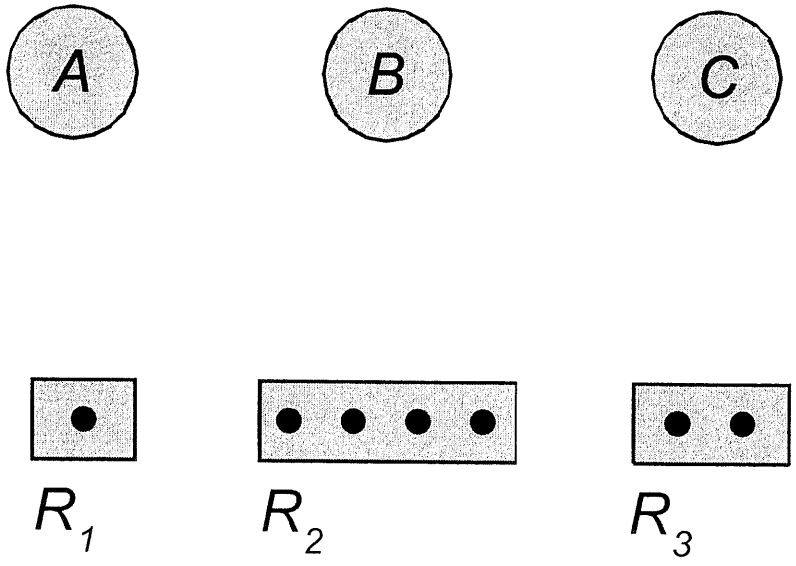
Zu einem Zeitpunkt t sei in einem System mit drei Prozessen [A, B, C] folgende Betriebsmittel-Zuteilung gegeben:

A hält **eine** Instanz von Ressource-Typ **R1** und **zwei** Instanzen vom Ressource-Typ **R2**
B hält **eine** Instanz vom Ressource-Typ **R2** und **eine** Instanz vom Ressource-Typ **R3**
C hält **eine** Instanz vom Ressource-Typ **R2.**

Dabei existieren insgesamt im System:
**Eine** Instanz vom Ressource-Typ **R1**
**Vier** Instanzen vom Ressource-Typ **R2**
**Zwei** Instanzen vom Ressource-Typ **R3.**

Folgende Anforderungen bestehen:
A benötigt **eine** Instanz von **R2,** um erfolgreich terminieren zu können
B benötigt **eine** Instanz von **R2,** um erfolgreich terminieren zu können
C benötigt **eine** Instanz von **R3,** um erfolgreich terminieren zu können.

a) Vervollständigen Sie den Betriebsmittel-Zuteilungsgraphen in folgendem Schema:




b) Gibt es Prozesse, die blockiert (= verklemmt) sind? Wenn ja: welche Prozesse sind dies und warum?

c) Was ändert sich am obigen Systemzustand bzgl. des Blockierens von Prozessen, wenn ***zusätzlich*** zu den bestehenden Anforderungen gilt:
C benötigt **eine** Instanz vom Ressource-Typ **R1,** um erfolgreich terminieren zu können?

Fortsetzung nächste Seite!

## Aufgabe 7: Sicheres Terminieren von Prozessen

Zu einem Zeitpunkt *t* sei in einem System mit fünf Prozessen [A, B, C, D, E] folgende Betriebsmittel-Zuteilung gegeben:

| *Prozess* | *Farbdrucker* | *s/w-Drucker* | *CD-Brenner* | *Scanner* |
|---|---|---|---|---|
| A | 3 | 0 | 1 | 1 |
| B | 0 | 1 | 0 | 0 |
| C | 1 | 1 | 1 | 0 |
| D | 1 | 1 | 0 | 1 |
| E | 0 | 0 | 0 | 0 |

Nachfolgende Tabelle gibt an, wie viele Ressourcen die Prozesse noch benötigen, um bis zur erfolgreichen Terminierung arbeiten zu können:

| *Prozess* | *Farbdrucker* | *s/w-Drucker* | *CD-Brenner* | *Scanner* |
|---|---|---|---|---|
| A | 1 | 1 | 0 | 0 |
| B | 1 | 1 | 1 | 2 |
| C | 3 | 1 | 0 | 0 |
| D | 0 | 0 | 1 | 0 |
| E | 2 | 1 | 1 | 0 |

Von jedem Ressourcentyp sind insgesamt vorhanden:

| *Farbdrucker* | *s/w-Drucker* | *CD-Brenner* | *Scanner* |
|---|---|---|---|
| 6 | 3 | 4 | 2 |

a) Stellen Sie zunächst die für die beteiligten Prozesse benötigten Matrizen ***Need*** *(= momentaner Restbedarf),* ***Max*** *(= Maximalbedarf bei Betreten des Systems)* und ***Allocation*** *(= momentane Belegung)* und den Vektor ***Available*** *(= momentane Verfügbarkeit)* auf.
Prüfen Sie anschließend (z. B. mit dem Banker's Algorithmus), ob das System in einem sicheren Zustand ist (d. h. es existiert eine Reihenfolge, in der alle Prozesse terminieren können). Falls der Zustand *sicher* ist, reicht die Auflistung einer möglichen Terminierungsreihenfolge mit der jeweiligen Angabe des aus einer Terminierung resultierenden ***Available*** Vektors. Falls der Zustand *nicht sicher* ist, geben Sie einen Systemzustand an, in dem Prozesse nicht mehr erfolgreich terminieren können.

b) Ausgehend von obiger Situation benötige Prozess E statt einem nun zwei CD-Brenner. Kann diese Anforderung erfüllt werden?

c) Erläutern Sie knapp, warum der Lösungsansatz aus a) in real existierenden Systemen schwierig umzusetzen ist!

Fortsetzung nächste Seite!

**Aufgabe 8: Paging-Strategien**

Bei der Verwaltung von virtuellem Speicher werden Seitenverdrängungsalgorithmen benötigt, da der virtuelle Adressraum i. A. erheblich größer ist als der physikalisch tatsächlich vorhandene Speicher.

Gegeben sei nun ein Prozess, der auf die logischen Seiten 1 bis 4 in der folgenden Reihenfolge zugreift: **1, 3, 2, 4, 3, 3, 4, 3, 1**

Wie viele Seitenfehler werden produziert, wenn dem Prozess jeweils eine, zwei, drei oder vier Kacheln zur Verfügung stehen? Beantworten Sie diese Fragen für die Seitenverdrängungsalgorithmen:

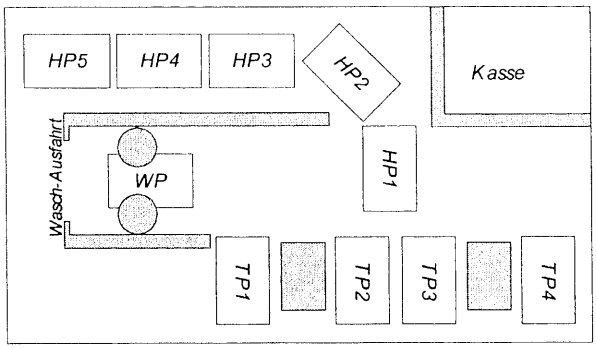
a) **OPT** (Belady's optimale Strategie)
b) **FIFO** (First-In-First-Out)
c) **LRU** (Least-Recently-Used)

Hinweis:
Beachten Sie dabei, dass die Kachel/n zu Beginn leer ist/sind; d. h. der erste Zugriff auf eine logische Seite erzeugt auf jeden Fall einen Seitenfehler.

Fortsetzung nächste Seite!

## Aufgabe 9: Prozesskoordination

Eine kleine PKW-Tankstelle besitze 4 Tankplätze (TP1, ..., TP4), wobei man an jeder Säule alle gängigen Treibstoff-Sorten zapfen kann. Ferner existiert ein Waschplatz zur Autowäsche, in dem zu einem Zeitpunkt genau ein PKW gewaschen werden kann (Waschplatz WP). Von der Straße her kommend können bis zu 5 PKWs in der Zufahrt zur Tankstelle halten und warten (Halteplätze HP1, ..., HP5). Ein Kunde will immer entweder tanken oder waschen - niemals beides. Weitere Warteplätze auf dem Tankstellen-Gelände sind nicht nutzbar. Realisieren Sie nun den reibungslosen Ablauf der Tankstelle mit Zähl-Semaphoren (Counting Semaphores).




a) Definieren Sie in einer beliebigen objektorientierten Sprache oder in Pseudo-Code die benötigten *Schnittstellen* (Konstruktoren und Methoden) und die internen Datenfelder einer Zähl-Semaphore `Semaphore`. Eine konkrete Implementierung der Konstruktoren- und Methodenrümpfe ist nicht nötig.

b) Benutzen Sie Ihre Zähl-Semaphore aus Teil a), um den unten stehenden Programmcode zu vervollständigen. Es können in Ihrer Lösung einer, keiner oder mehrere Semaphore-Aufrufe auf einer ... -Linie stehen.

```
class Tankstelle {
  Semaphore Tankplatz     = new Semaphore(.......);
  Semaphore Waschplatz    = new Semaphore(.......);
  Semaphore HaltePlatz    = new Semaphore(.......);
  public void benutzeTankstelle (Auto einAuto) {

    ........................................
    fahreWartestreifen(einAuto);
    if (einAuto.willTanken()) {

      ........................................
      belegeTankplatz(einAuto);

      ........................................
    }
    else {

      ........................................
      belegeWaschplatz(einAuto);

      ........................................
    }
    ........................................
    fahreAusfahrt(einAuto);

    ........................................
  }
  // [...]
}
```