BC402
Klausurfragen / Klausurlösungen

1. Ein ausführbares Programm mit einem Selektionsbild enthält die folgenden Ereignisblöcke:

1. Start-of-selection	**1. load-of-program**
2. at selection-screen	**2. initialisation**
3. at selection-screen output	**3. at selection-screen output**
4. initialisation	**4. at selection-screen**
5. load-of-program	**5. Start-of-selection**
6. at line-selection	**6. at line-selection**

in welcher Reihenfolge werden die Ereignisblöcke abgearbeitet?

- ☒ **5 4 3 2 1 6**
- ☐ 5 4 2 3 1 6
- ☐ 4 5 3 2 1 6
- ☐ 1 2 3 4 5 6
- ☐ 5 4 1 3 2 6

2. Gegeben ist der folgende Report:

```
10  REPORT zxy.
20  DATA:  field a TYPE c VALUE 'A',
30         field b TYPE c VALUE 'B'.
40  WRITE: / field a.
50  NEW-PAGE.
60  WRITE: / field b.
70  TOP-OF-PAGE.
80  WRITE: / 'Das ist der Titel'.
```

Geben Sie die Reihenfolge an, in der die Anweisungen ausgeführt werden:

- ☐ 40 80 50 80 60
- ☐ 40 50 60 70 80
- ☒ **80 40 50 80 60**

3. Welche Aussagen hinsichtlich der Lebensdauer und der Sichtbarkeit von Datenobjekten treffen zu?

- ☒ **Datenobjekte, die im Hauptprogramm angelegt werden (DATA...) sind global, d.h. sie sind im gesamten Programm, in dem sie deklariert wurden, sichtbar.**
- ☒ **Der Speicherplatz für globale Variablen eines Programmes wird erst nach Beenden des Programms freigegeben.**
- ☒ **Datenobjekte, die in einem Unterprogramm deklariert wurden, sind nur im jeweiligen Unterprogramm sichtbar.**
- ☒ **Wird ein Unterprogramm beendet, werden alle darin mit DATA deklarierten Variablen gelöscht.**
- ☒ **Mit TABLES angelegte Strukturen sind immer globale Datenobjekte, auch wenn sie in einem Unterprogramm deklariert wurden.**
- ☐ Die Lebensdauer der mit TABLES angelegten Strukturen entspricht der Lebensdauer globalen Objekten.
- ☒ **Mit STATICS im Unterprogramm angelegte Variablen sind nur im Unterprogramm sichtbar, werden aber beim beenden des Unterprogrammes nicht gelöscht.**
- ☒ **In Modulen der Dynproablauflogik deklarierte Variablen sind immer global**

4. Welche Aussagen treffen auf Zahlenliterale zu?

- ☒ **Zahlenliterale, die in den Wertebereich der Ganzzahlen (Integer) fallen, werden als Ganzzahl interpretiert.**
- ☐ Ein Zahlenliteral, das nicht in den Wertebereich der Ganzzahlen fällt, wird als Fließpunktzahl (Type f) interpretiert.
- ☒ **Die Genauigkeit von Berechnungen mit Zahlenliteralen ist abhängig von der Größe des Zahlenliterals.**
- ☐ Die Berechnungen mit Zahlenliteralen im Intervall von im Wertebereich der Ganzzahlen sind genauer als die, die nicht in diesem Bereich liegen.
- ☒ **Es ist besser, mit Konstanten zu arbeiten als mit Zahlenliteralen.**

5. Welche Aussagen treffen hinsichtlich der in ABAP verwendeten Arithmetiken zu?

- ☒ **Festpunktarithmetik ist die Standardarithmetik**
- ☒ **Sind alle an einer Berechnung beteiligten Komponenten ganzzahlig, wird die Ganzzahlarithmetik angewendet.**
- ☐ Ist mindestens eine Komponente nicht ganzzahlig, wird die Gleitpunktarithmetik angewendet.
- ☐ Fließpunktzahlen werden intern als Dualbruchsummen gespeichert. Zur Speicherung stehen 53 bit zur Verfügung. Auf Grund der hohen Genauigkeit von Gleitpunktzahlen sind diese besonders für betriebswirtschaftliche Berechnungen geeignet.
- ☐ Bei gepackten Zahlen werden immer 2 Ziffern in ein Bate geschrieben.

6. Gegeben ist der folgende Programmausschnitt:

```
DATA:           BEGIN OF wa,
                       sflight TYPE sflight,
                       auslastung TYPE I,
                END OF wa.
```

Welche Aussagen treffen zu?

- ☐ Der Programmausschnitt ist syntaktisch falsch.
- ☒ **Bei der Komponente SFLIGHT handelt es sich um eine Struktur**
- ☒ **Die Struktur wa ist eine flache Struktur.**
- ☐ Die Struktur wa ist eine tiefe Struktur
- ☒ **Die Struktur wa ist eine geschachtelte Struktur**

7. Schreiben Sie die Structur aus Aufgabe 6 so um, dass eine nicht geschachtelte Strukture entsteht.

```
DATA: BEGIN OF wa,
        include TYPE sflight as sflight,
        data auslastung TYPE i,
      END OF wa.
```

8. Gegeben ist die interne Tabelle IT. Vom Progamm zfield_symbols wird sie mit folgendem Inhalt geladen:

CARRID	CONNID	FLDATE	SEATSMAX	SEATSOCC
...				
LH	400	25.09.2006	380	200
LH	400	26.09.2006	380	220
LH	400	27.09.2006	380	350
LH	400	28.09.2006	380	250
...				

Ergänzen Sie das Programm zfield_symbols so, dass für den Flug LH 400 26.09.2006 das Feld Seatsocc von 220 auf 225 gesetzt wird. Verwenden Sie dazu das Feldsymbol <fs> und eine performante READ-Anweisung. Geben Sie anschließend alle Datensätze über eine LOOP-Schleife aus. Nutzen Sie auch dazu das Feldsymbol <fs>.

```
REPORT zfield_symbols.
DATA: it TYPE SORTED TABLE OF sflight
        WITH UNIQUE KEY carrid connid fldate.
FIELD-SYMBOLS: <fs> _type sflight_____________________ .
START-OF-SELECTION.
SELECT * FROM sflight INTO CORRESPONDING FIELDS OF TABLE it
WHERE carrid = 'LH' AND connid = '400' AND
      fldate >= '20060925'.
...
READ TABLE it ______
assigning <fs>
Carrid = 'LH'
Connid = '400'
With key fldate = '20060926' and seatsocc = '220'_____
IF sy-subrc = 0.
*Änderung programmieren
_____________________________________________________
<FS>-seatsocc = ,225'
ENDIF.
LOOP AT it _____________________________________________
Assigning <fs>
    WRITE: / <fs>-carrid,
<fs>-connid,
<fs>-fldate,
<fs>-seatsmax,
<fs>-seatsocc.
ENDLOOP.
```

9. Welche Aussagen treffen hinsichtlich der Performance bei internen Tabellen zu?

- ☐ Die Read-Anweisung READ TABLE ... WITH TABLE KEY ist immer performanter als die Anweisung READ TABLE ... WITH KEY... unabhängig von der verwendeten Tabellenart.
- ☒ **Wird bei der READ-Anweisung die sich auf eine Tabelle vom Typ SORTED bezieht, der Schlüssel unvollständig, aber linksbündig ohne Lücken angegeben, wird die Suche der Tabellezeile trotzdem optimiert (binäre Suche) durchgeführt.**
- ☐ Der im Punkt 2 beschriebene Sachverhalt trifft auch auf Hashed-Tabellen zu.
- ☒ **Die WHERE-Anweisung in einer LOOP-Schleife (LOOP AT it INTO wa WHERE carrid = 'LH'.) kann lediglich bei sortierten Tabellen zu einer Optimierung des Laufzeitverhaltens führen**
- ☐ Die Bearbeitung geschachtelter Tabellen mit Feldsymbolen ist immer Laufzeitgünstiger als deren Bearbeitung über den Arbeitsbereich.

10. Gegeben ist folgender Programmausschnitt:

```
DATA: zk TYPE string.
zk = 'Die Klausur ist zu einfach'.
SEARCH zk FOR 'ist'.
...
```

Wie sind die Systemvariablen sy-subrc und sy-fdpos geladen?
sy-subrc : _____0
sy-fdpos: _____12

11. In einem Programm wird folgender Zeichenkettenvergleich ausgeführt:

```
DATA:           zk1 TYPE string,
                zk2 TYPE string.

zk1 = 'ABC'.
zk2 = 'AB'.
if zk1 CO zk2.
    write: 'Zeile 1'.
else.
    write: 'Zeile 2'.
endif.
```

- ☐ Es wird die Zeichenkette 'Zeile 1' ausgegeben
- ☒ **Es wird die Zeichenkette 'Zeile 2' ausgegeben**
- ☐ sy-fdpos ist mit 0 geladen
- ☐ sy-fdpos ist mit 1 geladen
- ☒ **sy-fdpos ist mit 2 geladen**
- ☐ sy-fdpos ist mit 3 geladen

12. In einem Programm wird folgender Zeichenkettenvergleich ausgeführt:

```
DATA:           zk1 TYPE string,
                zk2 TYPE string.

zk1 = 'ABC'.
zk2 = 'AB'.
if zk1 CS zk2.
    write: 'Zeile 1'.
else.
    write: 'Zeile 2'.
endif.
```

- ☒ **Es wird die Zeichenkette 'Zeile 1' ausgegeben**
- ☐ Es wird die Zeichenkette 'Zeile 2' ausgegeben
- ☒ **sy-fdpos ist mit 0 geladen**
- ☐ sy-fdpos ist mit 1 geladen
- ☐ sy-fdpos ist mit 2 geladen
- ☐ sy-fdpos ist mit 3 geladen

13. Welche Aussagen zu einer Funktionsgruppen ZFGR mit zwei Funktionsbausteinen (FB1, FB2) und globalen Datendeklarationen sind richtig?

- ☐ Im Include ZFGRTOP der FunkFunktionsgruppen sind die globalen Datenobjekte deklariert.
- ☐ Auf die globalen Daten der Funktionsgruppe kann vom aufrufenden Programm direkt zugegriffen werden.
- ☒ **Im Gegensatz zu globalen Klassen könnten in der Funktionsgruppe auch Dynpros angelegt werden.**
- ☐ Die Funktionsgruppe kann (wie globale Klassen) mehrfach instanziiert werden.
- ☒ **Auf die in der Funktionsgruppe deklarierten Daten haben nur die Funktionsbausteine der Funktionsgruppe zugriff. Es ist nicht möglich, vom rufenden Programm auf diese Daten direkt zuzugreifen.**
- ☒ **Das Rahmenprogramm der Funktionsgruppe ZFGR heißt SAPZFGR.**
- ☒ **Das Include LZFGRUXX enthält Include-Anweisungen für die Includeprogramm LZFGRU01 und LZFGRU02. Diese beiden Includes enthalten je einen der beiden Funktionsbausteine.**
- ☐ Es ist nicht möglich, dass der Funktionsbaustein FB1 die globalen Datenobjekte lädt und der Funktionsbaustein FB2 auf die von FB1 geladenen Datenobjekte zugreift. Ursache dafür ist, dass vor jedem Aufruf eines Funktionsbausteins (CALL FUNCTION 'FB1>' ... bzw. CALL FUNCTION 'FB2' ...) die globalen Daten auf ihre Initialwerte zurückgesetzt werden.
- ☒ **Importparameter der Funktionsbausteine können als optional gekennzeichnet werden.**
- ☐ Optionale Parameter müssen beim Aufruf des Funktionsbausteins nicht übergeben werden. Deshalb muss für diese Parameter ein Vorschlagswert im Funktionsbaustein angegeben werden.
- ☐ Optionale Parameter müssen immer mit der Eigenschaft „Wertübergabe" angelegt werden.
- ☒ **Exportparameter sind immer optional.**
- ☐ Um eine interne Tabelle an einen Funktionsbaustein übergeben zu können, muss der Parameter, der diese interne Tabelle entgegennimmt, immer mit einem globalen Tabellentyp typisiert sein

14. Welche der Select-Anweisungen ist am performantesten? Gehen Sie bei der Bewertung davon aus, dass in der Tabelle scarr nur die Fluggesellschaften AA, AT, BA, LH eingetragen sind. Alle Select-Anweisungen selektieren die Datensätze zu den Fluggesellschaften AA und BA.

```
1. Anweisung
SELECT * FROM scarr INTO TABLE it_scar
WHERE carrid IN ( 'AA','BA' ).
2. Anweisung
SELECT * FROM scarr INTO TABLE it_scar
WHERE carrid NOT IN ( 'AT','LH' ).
3. Anweisung
SELECT * FROM scarr INTO TABLE it_scar
WHERE carrid = 'AA' OR
      carrid = 'BA'.
```

- ☒ **Die Anweisung 1 ist am performantesten**
- ☐ Die Anweisung 2 ist am performantesten
- ☐ Die Anweisung 3 ist am performantesten

15. Welche Aussagen sind richtig

- ☒ **Die ORDER BY-Klausel sollte durch eine SORT-Anweisung im ABAP-Programm ersetzt warden, um die Datenbank zu entlasten.**
- ☒ **Die Übertragung von Daten von der Datenbank an die Applikationsebene erfolgt in 32 KB Blöcken.**
- ☐ Der Datenbankoptimizer legt die Suchstrategie ausschließlich durch die Angaben der WHERE-Klausel in der Select- Anweisung fest.
- ☒ **Wird in der WHERE-Klausel eine Negation zu einem Feld benutzt (WHERE carrid='LH and connid not in (,400', ,401') kann das betreffende Feld vom Optimizer nicht für die Suche nach einem geeigneten Index berücksichtigt werden.**
- ☐ Für das Lesen von Daten aus mehreren Datenbanktabellen sollten möglichst verschachtelte Select-Anweisungen eingesetzt werden.
- ☒ **Zugriffe auf Daten im Puffer des Applikationsservers benötigen ca. 0,1 ms, Zugriffe auf Daten im Datenbankpuffer ca. 1 ms, Zugriffe auf die Festplatte der Datenbank ca. 10 ms.**
- ☒ **Ziele der Programmierung sind (neben einem lauffähigem Programm):**
  - Laufzeit auf dem Applikationsserver minimieren (CPU-Last verringern)
  - Datenbanklast reduzieren
  - Netzlast verringern

16. Ein ausführbares Programm mit einem Selektionsbild enthält die folgenden Ereignisblöcke:

1. Start-of-selection	1. **load-of-program**
2. at selection-screen	2. **initialisation**
3. at selection-screen output	3. **at selection-screen output**
4. initialisation	4. **at selection-screen**
5. load-of-program	5. **Start-of-selection**
6. at line-selection	6. **at line-selection**

in welcher Reihenfolge werden die Ereignisblöcke abgearbeitet?
- ☐ 5 4 3 2 1 6
- ☐ 5 4 2 3 1 6
- ☐ 4 5 3 2 1 6
- ☐ 1 2 3 4 5 6
- ☐ 5 4 1 3 2 6

17. Gegeben ist der folgende Report:

```
10 REPORT zxy.
20 DATA: field a TYPE c VALUE 'A',
30       field b TYPE c VALUE 'B'.
40 WRITE: / field a.
50 NEW-PAGE.
60 WRITE: / field b.
70 TOP-OF-PAGE.
80 WRITE: / 'Das ist der Titel'.

Geben Sie die Reihenfolge an, in der die
Anweisungen ausgeführt werden:
```

- ☐ 40 80 50 80 60
- ☐ 40 50 60 70 80
- ☐ **80 40 50 80 60**

18. Welche Aussagen hinsichtlich der Lebensdauer und der Sichtbarkeit von Datenobjekten treffen zu?
- ☐ Datenobjekte, die im Hauptprogramm angelegt werden (DATA...) sind global, d.h. sie sind im gesamten Programm, in dem sie deklariert wurden, sichtbar.
- ☐ Der Speicherplatz für globale Variablen eines Programmes wird erst nach Beenden des Programms freigegeben.
- ☐ Datenobjekte, die in einem Unterprogramm deklariert wurden, sind nur im jeweiligen Unterprogramm sichtbar.
- ☐ Wird ein Unterprogramm beendet, werden alle darin mit DATA deklarierten Variablen gelöscht.
- ☐ Mit TABLES angelegte Strukturen sind immer globale Datenobjekte, auch wenn sie in einem Unterprogramm deklariert wurden.
- ☐ Die Lebensdauer der mit TABLES angelegten Strukturen entspricht der Lebensdauer globalen Objekten.
- ☐ Mit STATICS im Unterprogramm angelegte Variablen sind nur im Unterprogramm sichtbar, werden aber beim beenden des Unterprogrammes nicht gelöscht.
- ☐ In Modulen der Dynproablauflogik deklarierte Variablen sind immer global

19. Welche Aussagen treffen auf Zahlenliterale zu?
- ☐ Zahlenliterale, die in den Wertebereich der Ganzzahlen (Integer) fallen, werden als Ganzzahl interpretiert.
- ☐ Ein Zahlenliteral, das nicht in den Wertebereich der Ganzzahlen fällt, wird als Fließpunktzahl (Type f) interpretiert.
- ☐ Die Genauigkeit von Berechnungen mit Zahlenliteralen ist abhängig von der Größe des Zahlenliterals.
- ☐ Die Berechnungen mit Zahlenliteralen im Intervall von im Wertebereich der Ganzzahlen sind genauer als die, die nicht in diesem Bereich liegen.
- ☐ Es ist besser, mit Konstanten zu arbeiten als mit Zahlenliteralen.

20. Welche Aussagen treffen hinsichtlich der in ABAP verwendeten Arithmetiken zu?
- ☐ Festpunktarithmetik ist die Standardarithmetik
- ☐ Sind alle an einer Berechnung beteiligten Komponenten ganzzahlig, wird die Ganzzahlarithmetik angewendet.
- ☐ Ist mindestens eine Komponente nicht ganzzahlig, wird die Gleitpunktarithmetik angewendet.
- ☐ Fließpunktzahlen werden intern als Dualbruchsummen gespeichert. Zur Speicherung stehen 53 bit zur Verfügung. Auf Grund der hohen Genauigkeit von Gleitpunktzahlen sind diese besonders für betriebswirtschaftliche Berechnungen geeignet.
- ☐ Bei gepackten Zahlen werden immer 2 Ziffern in ein Bate geschrieben.

21. Gegeben ist der folgende Programmausschnitt:

```
DATA:     BEGIN OF wa,
                  sflight TYPE sflight,
                  auslastung TYPE I,
          END OF wa.
```

Welche Aussagen treffen zu?
- ☐ Der Programmausschnitt ist syntaktisch falsch.
- ☐ Bei der Komponente SFLIGHT handelt es sich um eine Struktur
- ☐ Die Struktur wa ist eine flache Struktur.
- ☐ Die Struktur wa ist eine tiefe Struktur
- ☐ Die Struktur wa ist eine geschachtelte Struktur

22. Schreiben Sie die Structur aus Aufgabe 6 so um, dass eine nicht geschachtelte Strukture entsteht.

```
DATA: BEGIN OF wa,
        include TYPE sflight as sflight,
        data auslastung TYPE i,
      END OF wa.
```

23. Gegeben ist die interne Tabelle IT. Vom Progamm zfield_symbols wird sie mit folgendem Inhalt geladen:

CARRID	CONNID	FLDATE	SEATSMAX	SEATSOCC
...				
LH	400	25.09.2006	380	200
LH	400	26.09.2006	380	220
LH	400	27.09.2006	380	350
LH	400	28.09.2006	380	250
...				

Ergänzen Sie das Programm zfield_symbols so, dass für den Flug LH 400 26.09.2006 das Feld Seatsocc von 220 auf 225 gesetzt wird. Verwenden Sie dazu das Feldsymbol <fs> und eine performante READ-Anweisung. Geben Sie anschließend alle Datensätze über eine LOOP-Schleife aus. Nutzen Sie auch dazu das Feldsymbol <fs>.

```
REPORT zfield_symbols.
DATA: it TYPE SORTED TABLE OF sflight
      WITH UNIQUE KEY carrid connid fldate.
FIELD-SYMBOLS: <fs> _type sflight__________________
.
START-OF-SELECTION.
SELECT * FROM sflight INTO CORRESPONDING FIELDS OF
TABLE it
WHERE carrid = 'LH' AND connid = '400' AND
      fldate >= '20060925'.
...
READ TABLE it ______
assigning <fs>
Carrid = 'LH'
Connid = '400'
With key fldate = '20060926' and seatsocc = '220'__
IF sy-subrc = 0.
*Änderung programmieren
___________________________________________________
<FS>-seatsocc = ,225'
ENDIF.
LOOP AT it
______________________________________________
Assigning <fs>
   WRITE: / <fs>-carrid,
<fs>-connid,
<fs>-fldate,
<fs>-seatsmax,
<fs>-seatsocc.
ENDLOOP.
```

24. Welche Aussagen treffen hinsichtlich der Performance bei internen Tabellen zu?
- ☐ Die Read-Anweisung READ TABLE ... WITH TABLE KEY ist immer performanter als die Anweisung READ TABLE … WITH KEY… unabhängig von der verwendeten Tabellenart.
- ☐ Wird bei der READ-Anweisung die sich auf eine Tabelle vom Typ SORTED bezieht, der Schlüssel unvollständig, aber linksbündig ohne Lücken angegeben, wird die Suche der Tabellezeile trotzdem optimiert (binäre Suche) durchgeführt.
- ☐ Der im Punkt 2 beschriebene Sachverhalt trifft auch auf Hashed-Tabellen zu.
- ☐ Die WHERE-Anweisung in einer LOOP-Schleife (LOOP AT it INTO wa WHERE carrid = 'LH'.) kann lediglich bei sortierten Tabellen zu einer Optimierung des Laufzeitverhaltens führen
- ☐ Die Bearbeitung geschachtelter Tabellen mit Feldsymbolen ist immer Laufzeitgünstiger als deren Bearbeitung über den Arbeitsbereich.

25. Gegeben ist folgender Programmausschnitt:

```
DATA: zk TYPE string.
Zk = 'Die Klausur ist zu einfach'.
SEARCH zk FOR 'ist'.
...
```

Wie sind die Systemvariablen sy-subrc und sy-fdpos geladen?
sy-subrc : ____________0
sy-fdpos: ____________12

26. In einem Programm wird folgender Zeichenkettenvergleich ausgeführt:

```
DATA:      zk1 TYPE string,
           zk2 TYPE string.
zk1 = 'ABC'.
zk2 = 'AB'.
if zk1 CO zk2.
   write: 'Zeile 1'.
else.
   write: 'Zeile 2'.
endif.
```

- ☐ Es wird die Zeichenkette 'Zeile 1' ausgegeben
- ☐ Es wird die Zeichenkette 'Zeile 2' ausgegeben
- ☐ sy-fdpos ist mit 0 geladen
- ☐ sy-fdpos ist mit 1 geladen
- ☐ sy-fdpos ist mit 2 geladen
- ☐ sy-fdpos ist mit 3 geladen

27. In einem Programm wird folgender Zeichenkettenvergleich ausgeführt:

```
DATA:      zk1 TYPE string,
           zk2 TYPE string.
zk1 = 'ABC'.
zk2 = 'AB'.
if zk1 CS zk2.
   write: 'Zeile 1'.
else.
   write: 'Zeile 2'.
endif.
```

- ☐ Es wird die Zeichenkette 'Zeile 1' ausgegeben
- ☐ Es wird die Zeichenkette 'Zeile 2' ausgegeben
- ☐ sy-fdpos ist mit 0 geladen
- ☐ sy-fdpos ist mit 1 geladen
- ☐ sy-fdpos ist mit 2 geladen
- ☐ sy-fdpos ist mit 3 geladen

28. Welche Aussagen zu einer Funktionsgruppen ZFGR mit zwei Funktionsbausteinen (FB1, FB2) und globalen Datendeklarationen sind richtig?
- ☐ Im Include ZFGRTOP der FunkFunktionsgruppen sind die globalen Datenobjekte deklariert.
- ☐ Auf die globalen Daten der Funktionsgruppe kann vom aufrufenden Programm direkt zugegriffen werden.
- ☐ Im Gegensatz zu globalen Klassen könnten in der Funktionsgruppe auch Dynpros angelegt werden.
- ☐ Die Funktionsgruppe kann (wie globale Klassen) mehrfach instanziert werden.
- ☐ Auf die in der Funktionsgruppe deklarierten Daten haben nur die Funktionsbausteine der Funktionsgruppe zugriff. Es ist nicht möglich, vom rufenden Programm auf diese Daten direkt zuzugreifen.
- ☐ Das Rahmenprogramm der Funktionsgruppe ZFGR heißt SAPZFGR.
- ☐ Das Include LZFGRUXX enthält Include-Anweisungen für die Includeprogramm LZFGRU01 und LZFGRU02. Diese beiden Includes enthalten je einen der beiden Funktionsbausteine.
- ☐ Es ist nicht möglich, dass der Funktionsbaustein FB1 die globalen Datenobjekte lädt und der Funktionsbaustein FB2 auf die von FB1 geladenen Datenobjekte zugreift. Ursache dafür ist, dass vor jedem Aufruf eines Funktionsbausteins (CALL FUNCTION 'FB1>' … bzw. CALL FUNCTION 'FB2' …) die globalen Daten auf ihre Initialwerte zurückgesetzt werden.
- ☐ Importparameter der Funktionsbausteine können als optional gekennzeichnet werden.
- ☐ Optionale Parameter müssen beim Aufruf des Funktionsbausteins nicht übergeben werden. Deshalb muss für diese Parameter ein Vorschlagswert im Funktionsbaustein angegeben werden.
- ☐ Optionale Parameter müssen immer mit der Eigenschaft „Wertübergabe" angelegt werden.
- ☐ Exportparameter sind immer optional.
- ☐ Um eine interne Tabelle an einen Funktionsbaustein übergeben zu können, muss der Parameter, der diese interne Tabelle entgegennimmt, immer mit einem globalen Tabellentyp typisiert sein

29. Welche der Select-Anweisungen ist am performantesten? Gehen Sie bei der Bewertung davon aus, dass in der Tabelle scarr nur die Fluggesellschaften AA, AT, BA, LH eingetragen sind. Alle Select-Anweisungen selektieren die Datensätze zu den Fluggesellschaften AA und BA.

```
1. Anweisung
SELECT * FROM scarr INTO TABLE it_scar
WHERE carrid IN ( 'AA','BA' ).
2. Anweisung
SELECT * FROM scarr INTO TABLE it_scar
WHERE carrid NOT IN ( 'AT','LH' ).
3. Anweisung
SELECT * FROM scarr INTO TABLE it_scar
WHERE carrid = 'AA' OR
      carrid = 'BA'.
```

- ☐ Die Anweisung 1 ist am performantesten
- ☐ Die Anweisung 2 ist am performantesten
- ☐ Die Anweisung 3 ist am performantesten

30. Welche Aussagen sind richtig
1. Die ORDER BY-Klausel sollte durch eine SORT-Anweisung im ABAP-Programm ersetzt warden, um die Datenbank zu entlasten.
2. Die Übertragung von Daten von der Datenbank an die Applikationsebene erfolgt in 32 KB Blöcken.
3. .der Datenbankoptimizer legt die Suchstrategie ausschließlich durch die Angaben der WHERE-Klausel in der Select- Anweisung fest.
4. Wird in der WHERE-Klausel eine Negation zu einem Feld benutzt (WHERE carrid='LH' and connid not in (,400', ,401') kann das betreffende Feld vom Optimizer nicht für die Suche nach einem geeigneten Index berücksichtigt werden.
5. Für das Lesen von Daten aus mehreren Datenbanktabellen sollten möglichst verschachtelte Select-Anweisungen eingesetzt werden.
6. Zugriffe auf Daten im Puffer des Applikationsservers benötigen ca. 0,1 ms,
   Zugriffe auf Daten im Datenbankpuffer ca. 1 ms,
   Zugriffe auf die Festplatte der Datenbank ca. 10 ms.
7. Ziele der Programmierung sind (neben einem lauffähigem Programm):
   - Laufzeit auf dem Applikationsserver minimieren (CPU-Last verringern)
   - Datenbanklast reduzieren
   - Netzlast verringern