Tutorial 3

BMS und CICS-Anwendungsprogrammierung (Cobol)

Dieses Tutorial vertieft die in den Tutorien 3 (C/C^{++}, Assembler, Cobol, PL/I) behandelten Themen. Insbesondere wird BMS (**B**asic **M**apping **S**upport) untermauert. Es wird demonstriert, wie sich in BMS mehrere Maps (maximal 7) in einem Mapset integrieren lassen, wobei jede Map das Layout eines separaten Screens beschreiben soll. Des weiteren wird ein neues Kommando der CICS-Anwendungsprogrammierung vorgestellt. Zu dem aus den anderen Tutorien 3 bekannten Kommando "EXEC CICS SEND MAP ..." wird hier das Kommando "EXEC CICS RECEIVE MAP ..." eingeführt. Ersteres sendet eine Map an ein Terminal, letzteres empfängt eine Map von einem Terminal.

Das hier vorgestellte Beispiel sendet eine Map an ein Terminal (EXEC CICS SEND MAP), welche als Input-Screen für zwei Summanden erscheint. Der User gibt in zwei Felder die beiden Summanden ein. Anschließend wird der Inhalt der Felder vom Terminal zum Mainframe transferiert (EXEC CICS RECEIVE MAP). Nun addiert das CICS-Anwendungsprogramm in Cobol beide Summanden. Abschließend sendet ein weiteres EXEC CICS SEND MAP-Kommando das Additionsergebnis an das Terminal.

Die Business-Logik, also die Komponente der CICS-Anwendung, die die beiden Summanden addiert, wird hier in Cobol erstellt. Diese, heute an den Universitäten vernachlässigte Sprache, ist aber in der CICS-Anwendungsentwicklung eine der üblichsten Sprachen überhaupt, die für das Erstellen der Business-Logik benutzt wird.

Informationen über CICS finden Sie auch im Buch "Einführung in z/OS und OS/390" (P. Herrmann / U. Kebschull / W. G. Spruth), das im Jahr 2004 im Oldenbourg-Verlag in der zweiten, korrigierten Auflage erschienen ist.

Aufgabe: *Beschäftigen Sie sich mit diesem Tutorial. Vollziehen Sie das hier vorgestellte Beispiel nach.*

Anschließend (Kapitel 1 bis 5) erfolgt die Vorstellung der aus didaktischen Gründen sehr einfach gehaltenen CICS-Anwendung. Das Kapitel "Syntax und Semantik von BMS-Programmen" auf der Seite 8 enthält eine vertiefende Beschreibung der Sprache BMS. Am Ende dieses Tutorials befindet sich ein Aufgabenpaket, welches zum Entwickeln einer eigenen CICS-Anwendung mit Schwerpunkt auf BMS genutzt werden kann.

1. Einrichten der CICS-Entwicklungsumgebung

Unter TSO bzw. ISPF müssen im Vorfeld einige Datasets angelegt werden. Wie Datasets angelegt werden (allocate), wird im Tutorial 1 beschrieben.

Aufgabe: *Für die Übung wird ein Dataset PRAKT14.CICS.BMS benötigt. Legen Sie diesen mit den in Abbildung 1 angegebenen Parametern an.*

Wenn noch nicht vorhanden, legen Sie ebenfalls einen Dataset PRAKT14.LIB an, ebenfalls mit den Parametern aus der Abbildung 1. Ist dieser Dataset schon vorhanden, sollten Sie aus diesem alle Member löschen und auf diesen anschließend einen Compress anwenden.

```
  Menu  RefList  Utilities  Help
------------------------------------------------------------------------------
                           Allocate New Data Set
                                                                More:     +
Data Set Name  . . . : PRAKT14.CICS.TEST2

Management class . . .                (Blank for default management class)
Storage class  . . . .                (Blank for default storage class)
 Volume serial . . . .                (Blank for system default volume) **
 Device type . . . . .                (Generic unit or device address) **
Data class . . . . . .                (Blank for default data class)
 Space units . . . . . KILOBYTE       (BLKS, TRKS, CYLS, KB, MB, BYTES
                                       or RECORDS)
 Average record unit                  (M, K, or U)
 Primary quantity  . . 16             (In above units)
 Secondary quantity    1              (In above units)
 Directory blocks  . . 2              (Zero for sequential data set) *
 Record format . . . . FB
 Record length . . . . 80
 Block size  . . . . . 320
 Data set name type  : PDS            (LIBRARY, HFS, PDS, or blank)  *
Command ===>
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F10=Actions  F12=Cancel
```

Abbildung 1: Allocate-Parameter

2. Definition des Layouts der zwei benötigten Screens unter BMS

Die zu erstellende CICS-Anwendung soll aus zwei Screens bestehen. Aus einem Eingabe- (Abbildung 2) und einem Ausgabe-Screen (Abbildung 3).

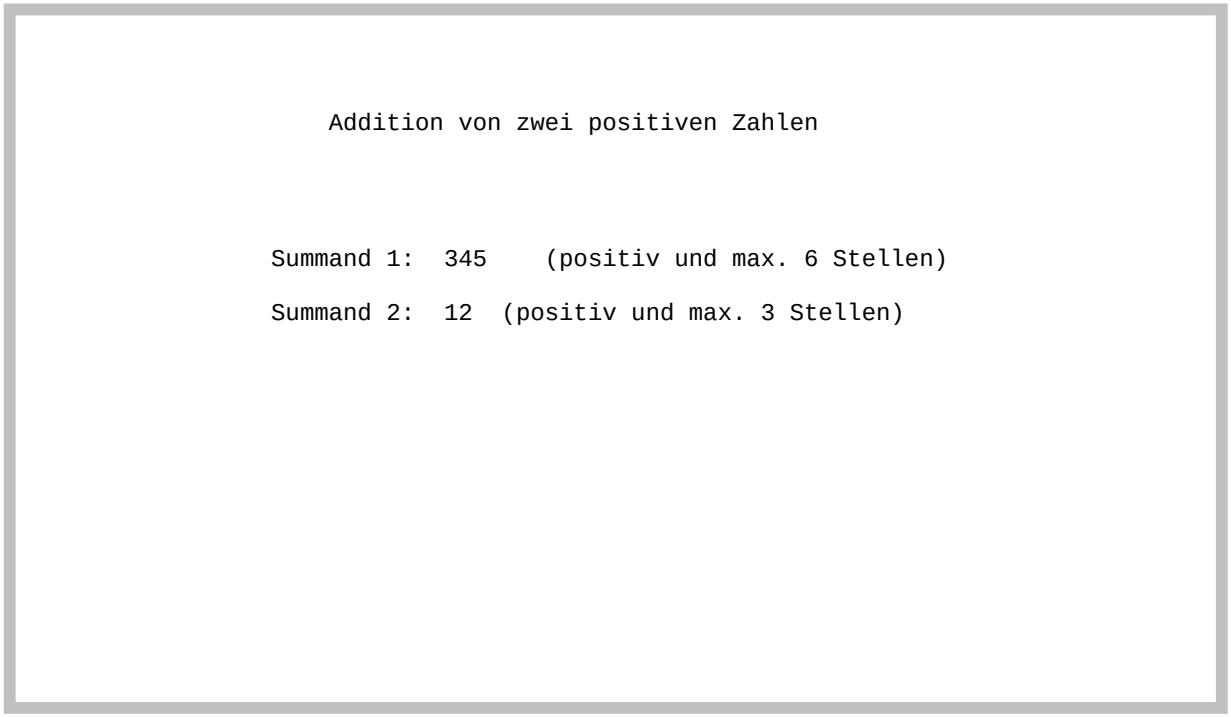

Abbildung 2: Eingabescreen

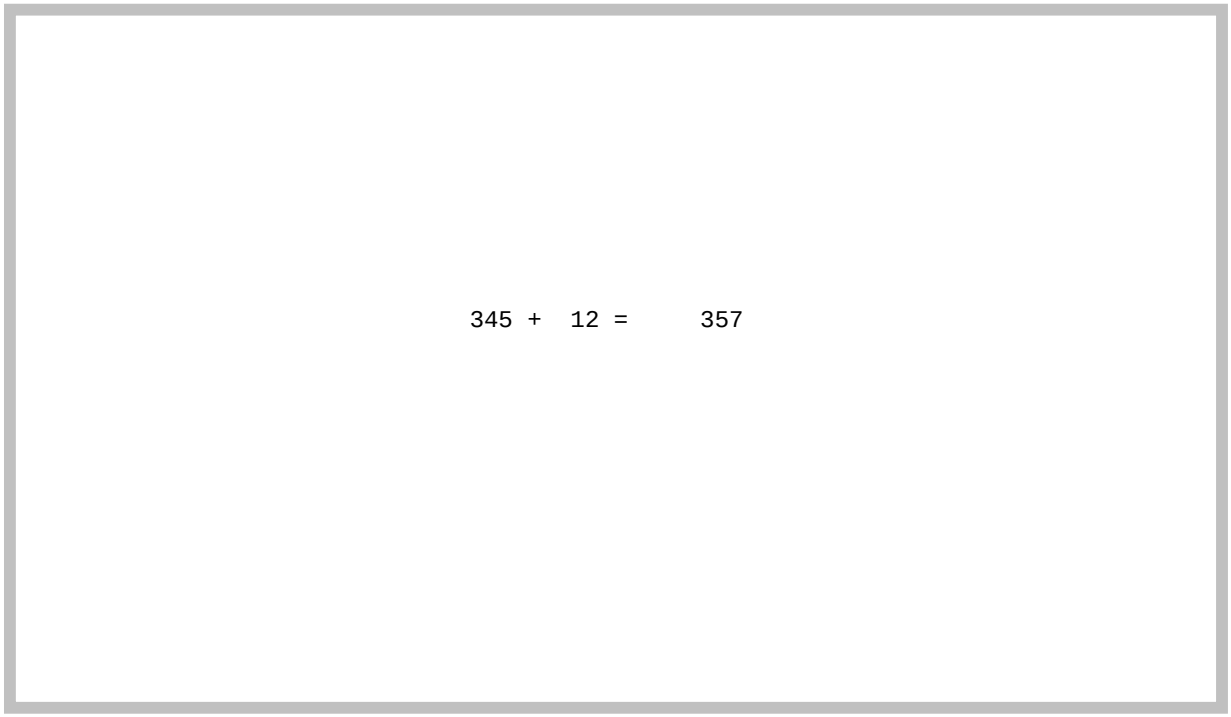

Abbildung 3: Ausgabescreen

Der Mapset besteht aus genau zwei Maps. Eine Map (EINGMAP) definiert das Layout des ersten Screens, in den die beiden Summanden eingegeben werden sollen (Abbildung 2). Die zweite Map (AUSGMAP) definiert den Screen, der die Ausgabe enthält (Abbildung 3). Abbildung 4 und Abbildung 5 zeigen ein JCL-Script, das ein BMS-Programm enthält, welches einen Mapset beinhaltet, dessen Name "M3BM014" ist (Abbildung 4, Zeile 000200).

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAKT14.CICS.BMS(MAPSET) - 01.01               Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -CAUTION- Profile changed to CAPS OFF (from CAPS ON) because data
==MSG>           contains lower case characters.
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000100 //PRAKT14M JOB (),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID
000200 //ASSEM    EXEC DFHMAPS,MAPNAME='M3BM014',RMODE=24
000300 //SYSUT1   DD *
000400 M3BM014 DFHMSD TYPE=MAP,MODE=INOUT,LANG=COBOL2,STORAGE=AUTO,TIOAPFX=YES
000500 *       Der Eingabemap des Mapsets.
000600 EINGMAP DFHMDI SIZE=(24,80),LINE=1,COLUMN=1,CTRL=FREEKB
000700         DFHMDF POS=(5,22),LENGTH=34,ATTRB=(ASKIP,NORM),                X
000800                INITIAL='Addition von zwei positiven Zahlen'
000900         DFHMDF POS=(10,18),LENGTH=10,ATTRB=(ASKIP,NORM),               X
001000                INITIAL='Summand 1:'
001100 A       DFHMDF POS=(10,30),LENGTH=6,ATTRB=(UNPROT,NUM,IC)
001200         DFHMDF POS=(10,37),LENGTH=28,ATTRB=(ASKIP,NORM),               Z
001300                INITIAL='(positiv und max. 6 Stellen)'
Command ===>                                                 Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Abbildung 4: JCL-Script mit BMS-Programm Panel 1/2

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAKT14.CICS.BMS(MAPSET) - 01.01               Columns 00001 00072
001400         DFHMDF POS=(12,18),LENGTH=10,ATTRB=(ASKIP,NORM),               E
001500                INITIAL='Summand 2:'
001600 B       DFHMDF POS=(12,30),LENGTH=3,ATTRB=(UNPROT,NUM)
001700         DFHMDF POS=(12,34),LENGTH=28,ATTRB=(ASKIP,NORM),               W
001800                INITIAL='(positiv und max. 3 Stellen)'
001900 *       Der Ausgabemap des Mapsets.
002000 AUSGMAP DFHMDI SIZE=(24,80),CTRL=(PRINT,FREEKB)
002100 SUMMND1 DFHMDF POS=(11,29),ATTRB=(ASKIP,NORM),LENGTH=6
002200         DFHMDF POS=(11,36),ATTRB=(ASKIP,NORM),LENGTH=1,INITIAL='+'
002300 SUMMND2 DFHMDF POS=(11,38),ATTRB=(ASKIP,NORM),LENGTH=3
002400         DFHMDF POS=(11,42),ATTRB=(ASKIP,NORM),LENGTH=1,INITIAL='='
002500 SUMME   DFHMDF POS=(11,44),ATTRB=(ASKIP,BRT),LENGTH=7
002600         DFHMSD TYPE=FINAL
002700         END
002800 /*
002900 //
****** *************************** Bottom of Data ***************************

Command ===> SUB                                             Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Abbildung 5: JCL-Script mit BMS-Programm Panel 2/2

Gibt man "SUB" ein, wird die Presentation-Logik für CICS erstellt. Eine fehlerfreie Übersetzung signalisiert "MAXCC=0".

3. Erstellen des CICS-Programms

Unter Nutzung des ISPF-Editors lässt sich die Business-Logik (Abbildung 6 bis Abbildung 8) erstellen. Sie ist hier in Cobol geschrieben.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       PRAKT14.CICS.BMS(PROGRAMM) - 01.01             Columns 00001 00072
****** ***************************** Top of Data ******************************
000100        IDENTIFICATION DIVISION.
000200        PROGRAM-ID. COB014.
000300        ENVIRONMENT DIVISION.
000400        DATA DIVISION.
000500        WORKING-STORAGE SECTION.
000600        COPY M3BM014.
000700        01  A       PIC 999999.
000800        01  A-DRUCK PIC ZZZZZ9.
000900        01  B       PIC 999.
001000        01  B-DRUCK PIC ZZ9.
001100        01  SUMME   PIC ZZZZZZ9.
001200        LINKAGE SECTION.
001300        PROCEDURE DIVISION.
001400            MOVE LOW-VALUES TO AUSGMAPO.
001500            MOVE LOW-VALUES TO EINGMAPO.
001600       * DIE EINGABEMASKE WIRD ANS TERMINAL GESCHICKT
Command ===>                                                 Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

Abbildung 6: Cobol-Programm (Panel 1/3)

Am Anfang des Programms der Business-Logik müssen alle Variablen der Struktur der symbolischen Ausgabemap gelöscht werden. Im Beispiel-Programm leisten das die Zeilen 001700 und 001800 (Abbildung 7).

Ein EXEC CICS SEND MAP-Kommando mit dem Parameter ERASE löscht das Bildschirmfenster komplett, bevor die gesendete Map auf dem Bildschirm erscheint. Man kann zwei Maps gleichzeitig auf dem Bildschirm darstellen, indem man zwei SEND-Kommandos nacheinander ausführt, wobei das erste einen ERASE-Parameter hat und das zweite nicht. So mischen sich die Inhalte beider Maps.

Der SEND MAP-Befehl mit dem Parameter "MAPONLY" sendet nur die Map, ohne die vom Programm veränderbaren Daten, an den Bildschirm. Der SEND MAP-Befehl mit dem Parameter "DATAONLY" sendet dagegen nicht die Map, sondern nur die variablen Daten vom Programm, die ungleich Null sind, an den Bildschirm. Lässt man beide Parameter weg, wird die Map einschließlich der variablen Daten gesendet. Im Beispiel-Programm hat das allererste SEND MAP-Kommando den Parameter "MAPONLY" (s. Programmzeile 001700 in Abbildung 7), weil ja mit der zu sendenden Map keine Ergebnisse der Programmverarbeitung an den Bildschirm geschickt werden sollen. Denn die zu sendende Map soll als Eingabe-Map für Daten dienen, nicht als Ausgabe-Map für Ergebnisse der Business-Logik.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAKT14.CICS.BMS(PROGRAMM) - 01.01              Columns 00001 00072
001700            EXEC CICS SEND MAP('EINGMAP') MAPSET('M3BM014') MAPONLY
001800                           ERASE END-EXEC.
001900       * DIE EINGEGEBENEN SUMMANDEN WERDEN VOM TERMINAL GEHOLT
002000            EXEC CICS RECEIVE MAP('EINGMAP') MAPSET('M3BM014')
002100                           INTO(EINGMAPI) END-EXEC.
002200       * SUMMANDEN VON EINGABEMAP-BEREICH (DATEN) IN INTEGER-VARIABLEN
002300       * KOPIEREN, DAMIT EINE ADDITION MOEGLICH WIRD
002400            MOVE AI TO A.
002500            MOVE BI TO B.
002600       * SUMME DER INTEGERZAHLEN BERECHNEN.
002700            ADD A, B   GIVING SUMME.
002800       * EVENTUELL VORANGEHENDE NULLEN DER SUMMANDEN DURCH LEERZEICHEN
002900       * ERSETZEN.
003000            MOVE A TO A-DRUCK,
003100            MOVE B TO B-DRUCK.
003200       * SUMMANDEN UND SUMME IN DEN AUSGABEMAP-BEREICH (DATEN) KOPIEREN
003300            MOVE A-DRUCK TO SUMMND1O.
Command ===>                                                Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

Abbildung 7: Cobol-Programm (Panel 2/3)

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAKT14.CICS.BMS(PROGRAMM) - 01.01              Columns 00001 00072
003400            MOVE B-DRUCK TO SUMMND2O.
003500            MOVE SUMME TO SUMMEO.
003600       * AUSGABEMAP AN TERMINAL SCHICKEN
003700            EXEC CICS SEND MAP('AUSGMAP') MAPSET('M3BM014')
003800                           FROM(AUSGMAPO) ERASE            END-EXEC.
003900       * PROGRAMMENDE
004000            GOBACK.
****** **************************** Bottom of Data ****************************
Command ===>                                                Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

Abbildung 8: Cobol-Programm (Panel 3/3)

Man beachte, dass Kommentarzeilen, die grundsätzlich mit einem Stern beginnen, immer in Spalte 7 beginnen müssen und dass Cobol-Kommandos grundsätzlich in Spalte 12 beginnen müssen.

4. Kompilieren der Business-Logik

Um die Business-Logik zu kompilieren, kann das in Abbildung 9 dargestellte JCL-Script verwendet werden. Dazu ist das Kommando "SUB" einzugeben.

```
    File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAKT14.CICS.BMS(COMPILE) - 01.00               Columns 00001 00072
****** ***************************** Top of Data ******************************
000100 //PRAKT14C JOB (),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
000200 //         REGION=4M
000300 //STEP1  EXEC COBCICS,PARM.TRN='COBOL3'
000400 //TRN.SYSIN DD DISP=SHR,DSN=&SYSUID..CICS.BMS(PROGRAMM)
000500 //COB.SYSLIB DD DSN=&SYSUID..LIB,DISP=SHR
000600 //LKED.SYSIN DD *
000700  NAME P3BM014(R)
000800 /*
****** **************************** Bottom of Data ****************************




Command ===> SUB                                            Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```

Abbildung 9: JCL-Script zum Kompilieren des Cobol-Programms

Sind Cobol-Programm und JCL-Script fehlerfrei, wird dies mit "MAXCC=0" quittiert.

5. Definitionen und Installation unter CICS

Nun kann man sich in einem separaten Fenster in CICS einloggen und die folgenden CICS-Kommandos ausführen. "014" ist jeweils durch die Nummer des PRAKT- oder PRAK-Accounts zu ersetzen. Details dazu werden im Tutorial 3, C-Version, behandelt.

CEDA DEFINE MAPSET(M3BM014) GROUP(PRAKT14)
CEDA DEFINE PROGRAM(P3BM014) GROUP(PRAKT14) LANGUAGE(LE370)
CEDA DEFINE TRANSACTION(W014) GROUP(PRAKT14) PROGRAM(P3BM014)

CEDA INSTALL GROUP(PRAKT14)

Nun sind Presentation-Logik, Business-Logik sowie die Transaktion "W014" unter CICS installiert. Alles kann nun mittels der Transaktions-ID "W014" aufgerufen werden.

6. JCL-Scripte mit BMS

Man betrachte noch einmal das JCL-Script, das Abbildung 4 und Abbildung 5 zeigen. Es fällt auf, dass einige Zeilen mit einem Schrägstrich (/) beginnen, andere nicht. Daran ist zu erkennen, dass in dieses JCL-Script ein BMS-Programm eingebettet ist. Alle durch den Schrägstrich eingeleiteten Zeilen gehören zum JCL-Script, alle anderen zum BMS-Programm, dass in dieses JCL-Script eingebettet ist.

Das BMS-Programm wird von der Prozedur DFHMAPS compiliert. Diese wird mit einem Hauptparameter, dem Mapset-Namen (hier MAPNAME='M3BM014') aufgerufen (Abbildung 4, Zeile 000200)

Syntax und Semantik von BMS-Programmen

Kommentarzeilen beginnen mit einem Stern (*) in der ersten Spalte. Im Beispiel-BMS-Programm (Abbildung 4 und Abbildung 5) gibt es zwei Kommentarzeilen, die jeweils den Beginn einer Map kennzeichnen.

Abbildung 10 zeigt die Struktur eines BMS-Programms.

```
Mapset-Label DFHMSD Mapset-Attribut,Mapset-Attribut,...
             Map-Name DFHMDI Map-Attribut,Map-Attribut,...
                             DFHMDF Feld-Attribut,Feld-Attribut,...
                             DFHMDF Feld-Attribut,Feld-Attribut,...
                             DFHMDF Feld-Attribut,Feld-Attribut,...
                             ...
             Map-Name DFHMDI Map-Attribut,Map-Attribut,...
                             DFHMDF Feld-Attribut,Feld-Attribut,...
                             DFHMDF Feld-Attribut,Feld-Attribut,...
                             ...
             Map-Name DFHMDI ...

             ...

             DFHMSD TYPE=FINAL
             END
```

Abbildung 10: Struktur eines BMS-Programms

Eine Mapset-Definition wird jeweils durch DFHMSD eingerahmt. Diese kann bis zu sieben Maps enthalten. Jede Map wird mittels DFHMDI definiert. Eine Map enthält viele Felder für die Datenein- sowie -ausgabe. Ein Feld wird jeweils durch DFHMDF und einige Feld-Attribute definiert.

DFHMSD

DFHMSD definiert einen Mapset. Es folgt eine Beschreibung der Mapset-Attribute sowie möglicher Werte.

TYPE sollte man, wie im Beispiel-Programm Abbildung 4, den Wert "MAP" zuweisen.

MODE kann man drei Werte zuweisen: IN, OUT oder INOUT. Dies hat Einfluss auf die zu erstellende symbolische Map (die im Programm einzusetzende Struktur). Bei IN enthält diese in PRAKT14.LIB angelegte Struktur pro Map nur eine <Map-Name>i-Variablen-Struktur, bei OUT nur eine <Map-Name>o-Variablen-Struktur sowie bei INOUT beide Variablen-Strukturen. Der Einfachheit halber kann man MODE grundsätzlich INOUT zuweisen.

LANG (LANGuage) weist man die Sprache seines CICS-Programms zu. Daraufhin bekommt die in die Sprache einzubindende symbolische Map die passende syntaktische Gestalt. In den CICS-Tutorien verwendete Werte sind C, COBOL2, ASM (Assembler) sowie PLI.

Es ist ratsam, *STORAGE* den Wert "AUTO" zuzuweisen.

DFHMDI

DFHMDI definiert eine Map. Es folgt eine Beschreibung der Map-Attribute sowie möglicher Werte.

SIZE gibt die Größe der zu definierenden Map an. Dieser Parameter ist auf JEDI unbedingt anzugeben. Üblich ist SIZE=(24,80), weil ein Screen meistens 24 Zeilen und 80 Spalten hat.

Die definierte Map überdeckt in der Regel genau den Screen. Doch das muss nicht zwingend so sein. Man kann eine Map beispielsweise so definieren, dass sich die linke obere Ecke von dieser in der Zeile 13 sowie der Spalte 15 befindet. Dies leistet zum Beispiel die folgende Definition einer Map mit dem Namen "MAP01":
000140 MAP01 DFHMDI SIZE=(12,66),*LINE*=13,*COLUMN*=15,CTRL=FREEKB
Wird nur ein rechteckiger Teilbereich des Screens von einer Map benutzt, d.h. die Map ist quasi von einem leeren Rahmen umgeben, so kann man unter Nutzung von LINE und COLUMN die Rahmenbreite links, rechts, oben und unten von der Map optimieren.

CTRL=FREEKB entriegelt die Tastatur für eine Eingabe.

DFHMDF

Da die Attribute von DFHMDF auch schon im Tutorial 3 (C-Version) behandelt wurden, wird hier nur ein kleines Beispiel zur Erinnerung angegeben.

Beispiel aus der Abbildung 4:

```
000700         DFHMDF POS=(5,22),LENGTH=34,ATTRB=(ASKIP,NORM),                X
000800                INITIAL='Addition von zwei positiven Zahlen'
```

In der Zeile 5 sowie der Spalte 22 beginnt ein Textfeld der Länge 34, welches mit den Attributen NORM (**NORM**ale Farbe) sowie ASKIP (**A**uto**SKIP**) belegt wird. Dieses Feld wird mit dem Text "Addition von zwei positiven Zahlen" initialisiert. Die Abbildung 2 zeigt schwarzweiß, wie dieses Feld auf dem Bildschirm ausgegeben wird.

Tabelle 1 zeigt die wichtigsten Attribute, die einem Feld zugewiesen werden können.

Attribut	Bedeutung
ASKIP PROT	**A**uto**SKIP**. **PROT**ect. Für die Ausgabe vorgesehene Felder. Hier ist eine Eingabe von Daten über die Tastatur nicht möglich.
UNPROT	**UNPROT**ect. Feld, über das mit der Tastatur eine Eingabe von Daten möglich ist.
NORM	**NORM**al. Feld hat eine Farbe normaler Helligkeit.
BRT	**BR**igh**T**. Feld hat eine auffallend helle Farbe.
DRK	**D**a**RK**. Die Schrift hat die gleiche Farbe wie ihr Hintergrund. Dieses Attribut bietet sich für ein Feld an, in das ein Passwort eingegeben werden soll.
IC	Nach dem Schreiben der Map steht der Cursor auf dem Beginn des Feldes mit diesem Attribut. Ausnahme: Der SEND MAP-Befehl wurde mit einer Option CURSOR gestartet, welche die genaue Cursorposition spezifiziert. Es bietet sich somit an, dass das erste Eingabefeld einer Map dieses Attribut bekommt.
NUM	**NUM**eral. Dieses Attribut lässt nur die Eingabe von Ziffern in ein Feld zu.

Tabelle 1: Feld-Attribute und ihre Bedeutung

Passen sämtliche Parameter von DFHMDF nicht auf eine Zeile, so ist die Zeile mit einem Komma sowie einem beliebigen Fortsetzungszeichen zu beenden. Als ein Fortsetzungszeichen sind die unterschiedlichsten Zeichen verwendbar, doch müssen diese unbedingt auf der letzten Spalte des Screens stehen (s. Abbildung 4 und Abbildung 5).

Zwischen zwei benachbarten Feldern muss grundsätzlich eine Lücke von mindestens einem Zeichen eingehalten werden.

Wird rechts neben einem Eingabefeld in unmittelbarer Nachbarschaft kein weiteres Feld definiert, ist es ratsam, unmittelbar rechts neben das Feld ein Stopp-Feld zu platzieren. Dieses verhindert ein Schreiben auf den Bildschirm über die rechte Feldgrenze des Eingabefeldes hinaus. Ein Stopp-Feld muss als geschütztes Feld (Ausgabefeld) definiert werden.

Lösen Sie EXAKT und VOLLSTÄNDIG alle nachfolgend gestellten Aufgaben.

Aufgabe: *Löschen Sie, wenn nötig, eine alte CICS-Gruppe, also die Gruppe, deren Namen identisch ist mit dem Namen ihres PRAKT<xx>- bzw. PRAK<xxx>-Accounts.*

Aufgabe: *Entwickeln Sie eine CICS-Anwendung, die den folgenden Anforderungen genügt:*

1) *Schreiben Sie ein individuelles BMS-Programm, das Bildschirmmasken erzeugt, die sich von anderen Übungsteilnehmern unterscheiden.*
2) *Ihr BMS-Programm muss drei Maps enthalten, von denen natürlich auch jede von Ihrem Cobol-Programm verwendet werden soll. Jede Map soll GENAU als ein Screen auf dem Bildschirm sichtbar werden. Von den drei Maps sollen zwei Eingabe-Maps der Dateneingabe sowie eine Ausgabe-Map der Datenausgabe dienen.*
3) *Kommentieren Sie Ihr BMS-Programm gut aus.*
4) *Mindestens ein Eingabefeld soll eine geheime Eingabe (z.B. ein Passwort) ermöglichen. Dazu muss dieses Feld das Attribut DRK bekommen.*
5) *Mindestens ein besonders wichtiges Eingabefeld soll das Attribut BRT erhalten. Analog dazu soll mindestens ein besonders wichtiges Ausgabefeld mittels des Attributs BRT als ein solches erkannt werden.*
6) *Mit dem jeweiligen Erscheinen einer Eingabe-Map auf dem Bildschirm soll der Cursor automatisch auf den Anfang des ersten Eingabefeldes springen.*
7) *Felder, in die ausschließlich Ziffern eingegeben werden dürfen, sollen mit einem passenden Attribut belegt werden, so dass eine Eingabe von anderen alphanumerischen Zeichen überhaupt nicht möglich ist.*
8) *Per SEND MAP soll eine Eingabemaske mit mindestens 6 Feldern an den Bildschirm geschickt werden. Von diesen 6 Feldern sollen mindestens 4 Felder Eingabefelder sein, über die die CICS-Anwendung Daten abfragt. Diese sollen vom User nun eingegeben werden. Anschließend soll die Map per RECEIVE MAP von der CICS-Anwendung empfangen werden.*
9) *Anschließend soll eine weitere Eingabemaske mit mindestens 6 Feldern an den Bildschirm geschickt werden. Von diesen sollen ebenfalls mindestens 4 Eingabefelder sein. Auch diese Map soll natürlich wieder per RECEIVE MAP von der CICS-Anwendung empfangen werden.*
10) *Nun soll die Verarbeitung der empfangenen Daten erfolgen. Wenn Sie dazu die entsprechenden Vorkenntnisse in Cobol besitzen, können Sie mit den Daten jetzt Berechnungen oder String-Operationen durchführen. Wenn nicht, reicht es hier, alle Daten aus den Datenstrukturen der Eingabe-Maps in die Datenstrukturen der Ausgabe-Map zu kopieren.*
11) *Abschließend muss nun ein EXEC CICS SEND MAP-Befehl die Ergebnisse der Datenverarbeitung durch das Cobol-Programm an den Bildschirm senden. Im Fall eines lediglichen Kopierens der Eingabedaten in den Ausgabebereich erscheinen da natürlich die Eingabedaten der Eingabe-Maps unverändert auf der Ausgabe-Map. Die Presentation-Logik der Ausgabe-Map sorgt aber für eine andere Darstellung der Daten auf dem Screen als bei der Eingabe der Daten in die Eingabe-Screens.*

Aufgabe: *Entwickeln Sie eine CICS-Anwendung ... (Fortsetzung)*

12) *BMS-Programm, Cobol-Programm und Compile-JCL-Script müssen im Dataset <Ihr Account>.CICS.BMS abgelegt werden.*

13) *Der Member mit dem BMS-Programm soll MAPSET2, der Member des Cobol-Programms soll PROGRAM2 und der Member des Compile-Scriptes soll COMPILE2 heißen.*

14) *Ihr Mapset, den Sie unter CICS installieren, muss den Namen M3BM<dreistelligeNr.IhresPrak(t)-Accounts>, Ihr Programm, dass Sie unter CICS installieren, muss P3BM<dreistelligeNr.IhresPrak(t)-Accounts> heißen.*

15) *Alle CICS-Komponenten müssen unter der CICS-Gruppe installiert werden, deren Namen identisch ist mit dem Namen ihres PRAKT<xx> bzw. PRAK<xxx>-Accounts.*

16) *Zur Abnahme Ihrer Lösungen muss Ihr Tutor Ihre CICS-Anwendung mittels der Transaktions-ID W<xxx> aufrufen können, wobei xxx wieder für die Nr. Ihres PRAK<xxx>- oder PRAKT<xx>-Accounts steht.*

Aufgabe: *Erzeugen Sie mittels Tastenkombination ALT-Druck von beiden Eingabe-Screens sowie von dem Ausgabe-Screen je einen Screenshot. Speichern Sie diesen im JPEG-Format ab. Die Dateigröße je Screenshot darf 150 KByte nicht überschreiten. Ist auf Ihrem PC keine Software installiert, die dazu in der Lage ist, darf auch das Bitmap-Format benutzt werden. Dessen Größe darf 500 KByte nicht überschreiten.*

Schicken Sie Ihrem Tutor alle drei Screenshots per E-Mail zu (genau eine Mail mit genau drei Anhängen; jeder Anhang genau eine UNVERPACKTE Datei).

Löschen Sie nichts von Ihren Daten, weder auf TSO-, noch auf CICS-Seite, damit Ihr Tutor sich Ihre Daten ansehen sowie Ihre Transaktion W<xxx> aufrufen kann.

Aufgabe: *Gehen Sie vom CUSTOMPAC MASTER APPLICATION MENU aus per "SD.ST" in eine Auflistung der Logfiles zu allen Jobs, die Sie mittels SUB ausgeführt haben. Löschen Sie alle Jobs, indem Sie links neben jeden Jobnamen "p" (purge) eingeben. Einen Job dürfen Sie natürlich nicht entfernen: Den Job in der EXECUTION-QUEUE. Denn das ist der Job, mit dem Sie jetzt gerade eingeloggt sind.*