Tutorial 3

CICS (COBOL)

ph v/2010/03

Ziel dieser Aufgabe ist es, ein "Hello World"-Programm in COBOL zu schreiben und Daten mittels CICS auf dem Bildschirm auszugeben.

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

Aufgabe: *Beschäftigen Sie sich mit diesem Tutorial und lösen Sie gewissenhaft alle kursiv geschriebenen und eingerahmten Aufgaben.*

TSO ist ein z/OS-Subsystem. CICS ist ein weiteres Subsystem. Jedes der beiden Subsysteme hat eine eigene Benutzerschnittstelle (eine eigene Shell). Um eine CICS-Anwendung zu erstellen, müssen wir mit beiden Subsystemen arbeiten: Mit TSO, um die Anwendung zu erzeugen, und mit CICS, um die Anwendung (unter dem CICS-Subsystem) auszuführen. Da z/OS ein Multi-User-Betriebssystem ist (multisession-fähig), können wir gleichzeitig eine TSO-Session und eine CICS-Session auf unserem Arbeitsplatzrechner laufen lassen. Jede Session läuft in einem eigenen Fenster.

Wir starten einen 3270-Emulator zunächst für eine TSO-Session und loggen uns ein.

Wir öffnen den "Data Set Utility"-Screen und erzeugen (Allocate) einen neuen Partitioned Dataset: "PRAKT20.CICS.COBOL". Dabei verwenden wir die in der nachstehenden Aufgabe angegebenen Parameter.

Außerdem brauchen wir noch einen Partitioned Dataset mit dem vorgegebenen Namen "PRAKT20.LIB", dessen Members von der Entwicklungsumgebung während der CICS-Programmentwicklung mit Daten gefüllt werden. Verwenden Sie bei dessen Anlage ebenfalls die unten angegebenen Parameter.

Dieser Name besteht aus einem "Project"-Begriff und einem "Group"-Begriff. Es fehlt der "Type"-Begriff. Wenn wir das Typ-Feld leer lassen, wird TSO dies nicht akzeptieren. Deshalb tragen wir den Namen 'PRAKT20.LIB' (mit Hochkommas!) in die Zeile "Data Set Name" ein. Damit wird auch dieser Dataset angelegt.

Aufgabe: *Legen Sie die Datasets "PRAKT20.CICS.COBOL" , "PRAKT20.LIB" an. Verwenden Sie dazu folgende Parameter:*

Space units . . . . . KILOBYTE	*Record format . . . . FB*
Primary quantity . . 16	*Record length 80*
Secondary quantity 1	*Block size 320*
Directory blocks . . 2	*Data set name type : PDS*

Unsere Anwendung besteht aus zwei Programmteilen und einem JCL-Script für die Übersetzung. Wir erstellen diese als Members in dem neuen Partitioned Dataset "PRAKT20.CICS.COBOL".

Ein sauber strukturiertes CICS-Programm besteht aus zwei Teilen: Business Logic und Presentation Logic. Business Logic ist der Teil, in dem Berechnungen erfolgen und Daten in einer Datenbank gelesen/geschrieben werden. Presentation Logic ist der Teil, in dem die Ergebnisse der Berechnungen so aufgearbeitet werden, dass sie dem Benutzer in einer ansprechenden Art auf dem Bildschirm dargestellt werden.

Business Logic wird in Sprachen wie C, C++, COBOL, PL/I, ASSEMBLER usw. geschrieben. Für die Presentation Logic gibt es viele Alternativen. Die modernste Alternative benutzt Java Server Pages und einen Web Application Server, um den Bildschirminhalt innerhalb eines Web-Browsers darzustellen. Die älteste (und einfachste) Alternative verwendet das CICS-BMS (**B**asic **M**apping **S**upport)-Subsystem. BMS-Programme werden in der BMS-Sprache geschrieben. In unserem Beispiel wird die Business Logic in COBOL und die Presentation Logic in BMS geschrieben.

Wir fangen mit dem letzteren an, rufen den "Edit Entry Panel Screen" auf und editieren ein Member "MAPCO01" für den neu angelegten Partitioned Dataset "PRAKT20.CICS.COBOL" (s. Abbildung 1). Man beachte den Buchstaben "O" und die Ziffer "0" in diesem Membernamen (!)

```
  File  Edit  Confirm  Menu  Utilities  Compilers Help
--------------------------------------------------------------------------------
EDIT       PRAKT20.CICS.COBOL(MAPCO01) - 01.05            Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000100 //PRAKT20M JOB (),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
000200 //         REGION=4M
000300 //ASSEM EXEC DFHMAPS,MAPNAME='MSET020',RMODE=24
000400 //COPY.SYSUT1  DD  *
000500 MSET020 DFHMSD TYPE=MAP,MODE=INOUT,LANG=COBOL2,STORAGE=AUTO,           *
000600                TIOAPFX=YES
000700 *       MENU MAP
000800 MAP020  DFHMDI SIZE=(24,80),CTRL=(PRINT,FREEKB)
000900         DFHMDF POS=(9,23),ATTRB=(ASKIP,NORM),LENGTH=34,               *
001000                INITIAL='WELCOME TO THE MAGIC WORLD OF CICS'
001100         DFHMDF POS=(12,33),ATTRB=(ASKIP,NORM),LENGTH=16,              *
001200                INITIAL='MEMBER PRAKT20 !'
001300         DFHMSD TYPE=FINAL
001400         END
001500 /*
001600 //
Command ===>                                                 Scroll ===> PAGE
 F1=Help      F3=Exit      F5=Rfind     F6=Rchange  F12=Cancel
```

Abbildung 1: Das BMS-Programm

Unser BMS-Programm verwendet 3 Befehlstypen: DFHMSD, DFHMDI und DFHMDF. Ein BMS-Bildschirm verwendet das 24 Zeilen x 80 Zeichen / Zeile 3270-Bildschirmformat. Die Ein- und Ausgabemasken werden als Felder innerhalb der 24x80-Matrix dargestellt, jeweils mit der Angabe: Zeilenadresse, Spaltenadresse und Feldlänge. Dies geschieht mit Hilfe des DFHMDF-Befehls. Der DFHMDF-Befehl in den Zeilen 000900 und 001000 definiert ein Feld, welches in Zeile 9, Spalte 23 beginnt, 34 Zeichen lang ist, und mit dem Wert "WELCOME TO THE MAGIC WORLD OF CICS" initialisiert wird. Unser Beispiel-BMS-Programm enthält 2 derartige DFHMDF-Befehle.

Der DFHMSD-Befehl (Zeile 000500) definiert einen "Mapset" mit dem Namen "MSET020". Eine Transaktion involviert in der Regel mehrere unterschiedliche Screens: Z.B. einen Screen, in dem der Benutzer zu einer Eingabe aufgefordert wird und einen weiteren Screen, welcher die Ergebnisse der Anfrage wiedergibt. Alle Maps (Screens) eines Transaktionstyps werden zu einem Mapset zusammengefaßt.

Die einzelnen Maps (Screens) eines Mapsets werden durch den Befehl DFHMDI definiert und zur Kennzeichnung mit einer Label versehen. In unserem einfachen "Hello World"-Beispiel besteht der Mapset aus einer einzigen Map, die in Zeile 000800 mit der Bezeichnung "MAP020" definiert wird.

Das Member "MAPCO01" stellt in Wirklichkeit ein JCL-Script dar. Im Gegensatz zu Tutorial 2 wird das zu verarbeitende File nicht mit INFILE='xxx.yyy.zzz' angegeben. Der JCL-Befehl in Zeile 000400 "//COPY.SYSUT1 DD *" besagt, dass das zu verarbeitende File unmittelbar danach folgt (Zeilen 000500 bis 001400).

Wir geben auf der Kommandozeile den ISPF-Befehl "SUB" ein. Es wird die Prozedur "DFHMAPS" ausgeführt.

JCL findet das Member "DFHMAPS" (Zeile 000300) in der Library "SYS1.PROCLIB(DFHMAPS)". Durch die Ausführung von "DFHMAPS" werden zwei Ausgabe-Files erzeugt. Einmal wird der übersetzte BMS-Quellcode in einen Member in eine CICS-interne Library mit dem Namen "CICSTS13.CICS.PRAKLOAD" gestellt. Hier kann ihn die BMS-Komponente des CICS-Subsystems später finden.

```
  File  Edit  Confirm  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       PRAKT20.LIB(MSET020) - 01.00                  Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
==MSG> -CAUTION- Profile is set to STATS ON. Statistics did not exist for
==MSG>           this member, but will be generated if data is saved.
000001        01  MAP020I.
000002            02  FILLER PIC X(12).
000003        01  MAP020O REDEFINES MAP020I.
000004            02  FILLER PIC X(12).
****** **************************** Bottom of Data ****************************









Command ===>                                                 Scroll ===> PAGE
 F1=Help      F3=Exit      F5=Rfind     F6=Rchange  F12=Cancel
```

Abbildung 2: Das Member "MSET020"

Das zweite Ausgabe-File wird als Member "MSET020" in den Partitioned Dataset "PRAKT20.LIB" gestellt (s. Abbildung 2). Alle Ein- und Ausgabe-Masken, die auf dem Bildschirm wiedergegeben werden sollen, werden ja bereits durch das BMS-Programm definiert. Das Business Logic-Programm bearbeitet diese Daten als COBOL-Strukturen, und diese werden von DFHMAPS während der Übersetzung von "MAPCO01" gleich miterzeugt und in "PRAKT20.LIB(MSET020)" abgespeichert.

Aufgabe: *Modifizieren Sie Ihr BMS-Programm derart, dass es den* ***Transaktionsnamen "U<Login-Nr.>", "TUTORIAL 3 IN COBOL"*** *sowie* ***den Namen oder die Namen der Autoren*** *enthält.Dieses BMS-Programm soll später einen Screen ähnlich der Abbildung 29 generieren. (Hinweis: Die Ausführung des JCL-Scriptes erfolgt fehlerfrei, wenn dieses mit der Statusmeldung "MAXCC=0" beendet wird).*

Wir rufen den Editor auf und erstellen in COBOL das Anwendungsprogramm COB020 (s. Abbildung 3). Letzteres enthält das CICS-Statement "EXEC CICS SEND MAP('MAP020') ...

```
  File  Edit  Confirm  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       PRAKT20.CICS.COBOL(COB020) - 01.03            Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000100        IDENTIFICATION DIVISION.
000200        PROGRAM-ID. COB020.
000300        ENVIRONMENT DIVISION.
000400        DATA DIVISION.
000500        WORKING-STORAGE SECTION.
000600        COPY MSET020.
000700        LINKAGE SECTION.
000800        PROCEDURE DIVISION.
000900            EXEC CICS SEND MAP('MAP020')
001000                           MAPSET('MSET020')
001100                           FROM(MAP020O)
001200                           ERASE
001300            END-EXEC.
001400            GOBACK.
****** **************************** Bottom of Data ****************************

Command ===>                                                 Scroll ===> PAGE
 F1=Help      F3=Exit      F5=Rfind     F6=Rchange  F12=Cancel
```

Abbildung 3: Das Anwendungsprogramm "COB020"

Dieses bewirkt, dass die Map mit dem im BMS-Programm definierten Mapnamen "MAP020" aus dem Mapset "MSET020" mit Hilfe des 3270-Protokolls an den Bildschirm des Endbenutzers übertragen wird.

```
  File  Edit  Confirm  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAKT20.CICS.COBOL(COBSTA03) - 01.00            Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000100 //PRAKT20C 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.COBOL(COB020)
000500 //COB.SYSLIB DD DSN=&SYSUID..LIB,DISP=SHR
000600 //LKED.SYSIN DD *
000700  NAME COB020(R)
000800 /*
****** **************************** Bottom of Data ****************************




Command ===> SUB                                          Scroll ===> PAGE
 F1=Help      F3=Exit      F5=Rfind     F6=Rchange  F12=Cancel
```

Abbildung 4: JCL-File "COBSTA03"

Dieses Programm soll nun übersetzt werden. Dazu wird für den Partitioned Dataset "PRAKT20.CICS.COBOL" ein neues Member "COBSTA03" erstellt (s. Abbildung 4). Dies ist ein JCL-File, das die Prozedur COBCICS enthält. COBCICS ruft zunächst den CICS-Precompiler auf, der alle CICS-Befehle in COBOL-Befehle übersetzt. Anschließend wird der COBOL-Compiler aufgerufen, der ein Maschinenprogramm erstellt und in eine für das CICS-Subsystem zugängliche Library stellt.

Wir geben "SUB" ein und warten, bis der Job ausgeführt wurde (s. Abbildung 4).

Wir entwickeln unsere Anwendung unter TSO und wollen sie unter CICS ausführen. Dazu muß sie als Teil des CICS-Subsystems installiert werden. Es ist komfortabel, mit 2 z/OS-Sessions gleichzeitig zu arbeiten. Dazu kann man einfach noch ein zweites Terminal öffnen, mit dem man CICS startet.

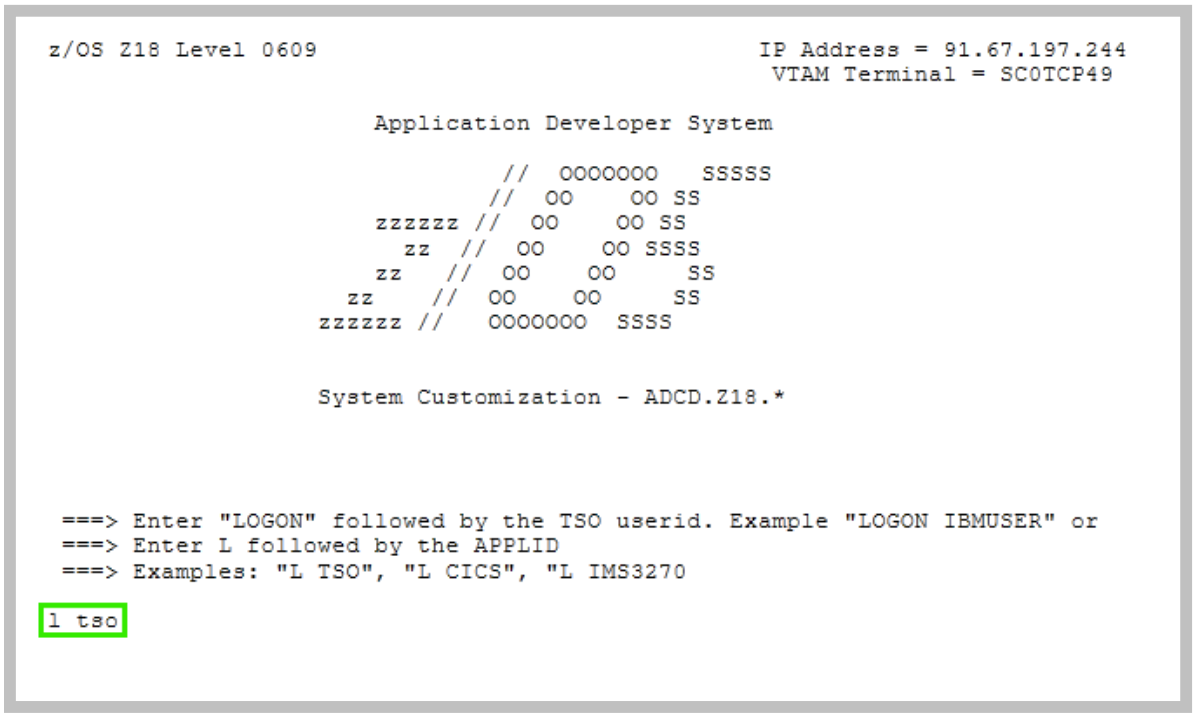

Abbildung 5: Logon-Bildschirm

Wir loggen uns anstatt "l tso" mit "L CICS" ein (einschließlich Eingabetaste) und rufen damit das CICS-Subsystem auf (s. Abbildung 5)

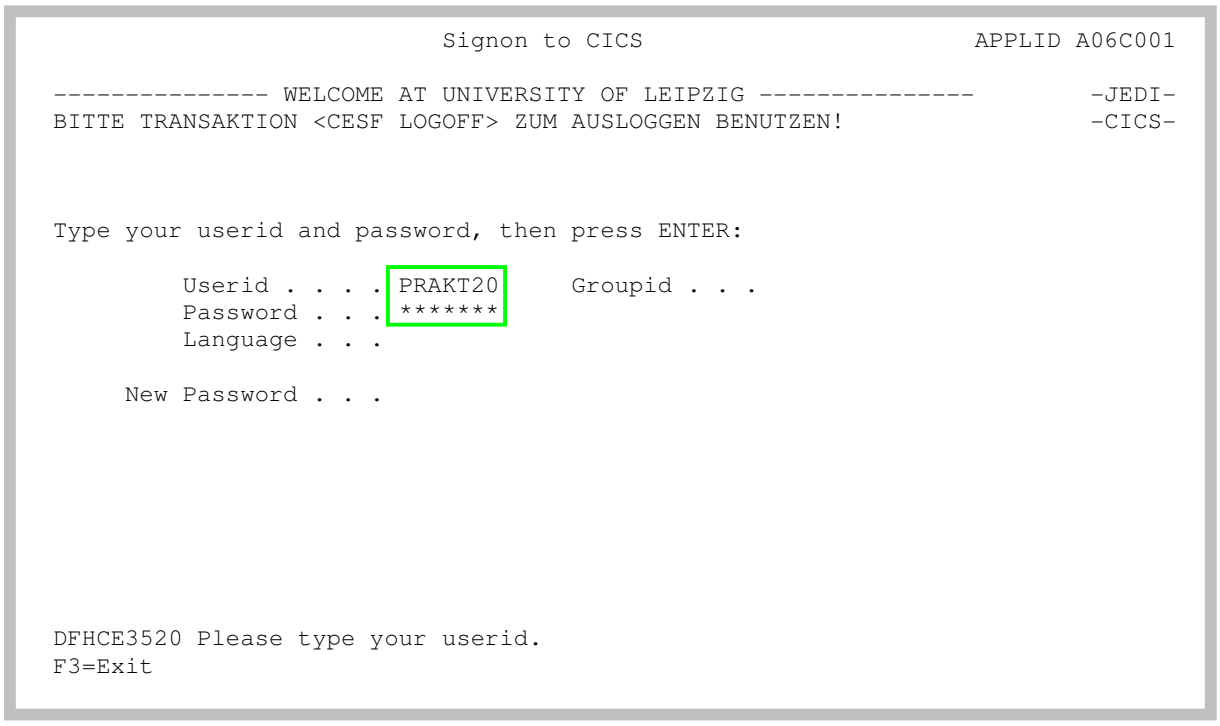

Abbildung 6: Signon to CICS-Bildschirm

Der Signon to CICS-Bildschirm erscheint (s. Abbildung 6). Hier müssen wir unseren TSO-Loginnamen sowie das entsprechende Passwort eingeben. Die Eingabetaste führt uns in den nächsten Screen (s. Abbildung 7).

Abbildung 7: Sign-on is complete

Mit der "Tab"-Taste bewegen wir den Cursor auf die unterste Zeile.

```
DFHCE3549 Sign-on is complete (Language ENU). CEDA DISPLAY GROUP(*)
                                                          10:42:29 IBM-3278-2
```

Abbildung 8: Beispielkommando "CEDA DISPLAY GROUP(*)"

CICS erwartet, dass man eine (von vielen) Transaktionen aufruft. Die unterschiedlichen Transaktionen werden normalerweise durch die Eingabe einer aus vier Zeichen bestehenden Tranaktions-ID aufgerufen.
Der CICS-Kommandointerpreter ist ebenfalls als Transaktion implementiert. Er wird mit der internen Transaktions-ID "CEDA" aufgerufen, gefolgt von einer Parameterliste, welche CICS-Kommandos sowie Eingabedaten enthält. Als Beispiel geben wir das Kommando "CEDA DISPLAY GROUP(*)" gefolgt von der Eingabetaste ein (s. Abbildung 8).

```
display group(*)
ENTER COMMANDS
  GROUP
  AOR2TOR
  ARTT
  ATC
  CBPS
  CEE
  CICREXX
  CICSJADP
  CICS0ADP
  CSQ
  CSQCKB
  CSQSAMP
  CTA1TCP
  C001EZA
  C001TCP
  DAVIN15
  DAVIN4
+ DAVIN8

                                                SYSID=C001 APPLID=A06C001
  RESULTS: 1 TO 17                          TIME:  00.00.00  DATE: 01.035
PF 1 HELP       3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 9: Auflistung der Gruppen

Wenn unter CICS Anwendungen (Transaktionen) installiert werden, dann wird für jede Transaktion eine "Group" angelegt. In der "Group" befinden sich die Komponenten: Das Anwendungsprogramm selbst, der dazugehörige Mapset sowie ein Eintrag, der die Transaktion mit einer 4-stelligen TRID (**TR**ansaktions-**ID**) verknüpft.

Es werden die ersten 17 Gruppen angezeigt (s. Abbildung 9).

```
CEDA DEFINE MAPSET(MSET020) GROUP(PRAKT20)
 ENTER COMMANDS
  GROUP
  AOR2TOR
  ARTT
  ATC
  CBPS
  CEE
  CICREXX
  CSQ
  CSQCKB
  CSQSAMP
  CTA1TCP
  C001EZA
  C001TCP
  DBA1
  DFH$ACCT
  DFH$AFFY
  DFH$AFLA
+ DFH$BABR

                                                SYSID=C001 APPLID=A06C001
  RESULTS: 1 TO 17                          TIME:  00.00.00  DATE: 01.037
PF 1 HELP       3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 10: Definition des Mapsets "MSET020" und der Gruppe "PRAKT20"

Wir definieren für unsere Transaktion eine eigene Gruppe "PRAKT20" und den dazugehörigen Mapset als "MSET020". Hierzu überschreiben wir die oberste Zeile, die als Kommandozeile dient, mit dem CEDA-Befehl "CEDA DEFINE MAPSET(MSET020) GROUP(PRAKT20)" (s. Abbildung 10, bitte Großbuchstaben benutzen!)

Um zu bestätigen drücken wir die Eingabetaste.

```
 CEDA DEFINE MAPSET(MSET020) GROUP(PRAKT20)
 OVERTYPE TO MODIFY                                        CICS RELEASE = 0530
  CEDA  DEFine Mapset( MSET020   )
   Mapset         : MSET020
   Group          : PRAKT20
   Description  ==>
   REsident     ==> No                No | Yes
   USAge        ==> Normal            Normal | Transient
   USElpacopy   ==> No                No | Yes
   Status       ==> Enabled           Enabled | Disabled
   RSl            : 00                0-24 | Public









   I New group PRAKT20 created.
                                                      SYSID=C001 APPLID=A06C001
   DEFINE SUCCESSFUL                                TIME:  00.00.00  DATE: 01.037
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 11: Bestätigung der Erstellung von Gruppe und Mapset

CICS teilt uns mit, dass die neue Gruppe "PRAKT20" erstellt sowie der Mapset erfolgreich definiert wurde (s. Abbildung 11).

Führen wir nochmals den Befehl "CEDA DISPLAY GROUP(*)" aus, so finden wir in der dargestellten Liste den Eintrag "PRAKT20" (erfordert ein Durchblättern der Liste unter Benutzung der Taste F8).

Nachdem der Mapset unter CICS definiert wurde, definieren wir jetzt das COBOL-Programm "COB020".

```
CEDA DEFINE PROGRAM(COB020) GROUP(PRAKT20)
 OVERTYPE TO MODIFY
  CEDA  Install
   All
   Connection   ==>
   DB2Conn      ==>
   DB2Entry     ==>
   DB2Tran      ==>
   DOctemplate  ==>
   Enqmodel     ==>
   File         ==>
   Journalmodel ==>
   LSrpool      ==>
   Mapset       ==>
   PARTItionset ==>
   PARTNer      ==>
   PROCesstype  ==>
   PROFile      ==>
   PROGram      ==>
+  Requestmodel ==>

                                                      SYSID=C001 APPLID=A06C001
  INSTALL SUCCESSFUL                                TIME:  00.00.00  DATE: 01.037
PF 1 HELP       3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 12: Definition des Programms "COB020"

Wir definieren für die Gruppe "PRAKT20" unser Anwendungsprogramm "COB020", indem wir den entsprechenden CEDA-Befehl

"CEDA DEFINE PROGRAM(COB020) GROUP(PRAKT20)"

in die oberste Zeile schreiben und anschließend die Eingabetaste betätigen (s. Abbildung 12).

```
CEDA DEFINE PROGRAM(COB020) GROUP(PRAKT20)
OVERTYPE TO MODIFY                                        CICS RELEASE = 0530
  CEDA  DEFine PROGram( PROG01   )
   PROGram        : COB020
   Group          : PRAKT20
   DEscription  ==>
   Language     ==>                  CObol | Assembler | Le370 | C | Pli
   RELoad       ==> No               No | Yes
   RESident     ==> No               No | Yes
   USAge        ==> Normal           Normal | Transient
   USElpacopy   ==> No               No | Yes
   Status       ==> Enabled          Enabled | Disabled
   RSl            : 00               0-24 | Public
   CEdf         ==> Yes              Yes | No
   DAtalocation ==> Below            Below | Any
   EXECKey      ==> User             User | Cics
   COncurrency  ==> Quasirent        Quasirent | Threadsafe
  REMOTE ATTRIBUTES
   DYnamic      ==> No               No | Yes
+  REMOTESystem ==>

                                                  SYSID=C001 APPLID=A06C001
  DEFINE SUCCESSFUL                            TIME:  00.00.00  DATE: 01.037
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 13: Definition von "COB020"

CEDA will einiges von uns wissen (s. Abbildung 13). Wir übernehmen alle Default-Werte und geben als Sprache "Le370" an (s. Abbildung 14).

```
CEDA DEFINE PROGRAM(COB020) GROUP(PRAKT20)
  OVERTYPE TO MODIFY                                        CICS RELEASE = 0530
   CEDA  DEFine PROGram( COB020   )
    PROGram        : COB020
    Group          : PRAKT20
    DEscription  ==>
    Language     ==> Le370              CObol | Assembler | Le370 | C | Pli
    RELoad       ==> No                 No | Yes
    RESident     ==> No                 No | Yes
    USAge        ==> Normal             Normal | Transient
    USElpacopy   ==> No                 No | Yes
    Status       ==> Enabled            Enabled | Disabled
    RSl            : 00                 0-24 | Public
    CEdf         ==> Yes                Yes | No
    DAtalocation ==> Below              Below | Any
    EXECKey      ==> User               User | Cics
    COncurrency  ==> Quasirent          Quasirent | Threadsafe
   REMOTE ATTRIBUTES
    DYnamic      ==> No                 No | Yes
 +  REMOTESystem ==>

                                                    SYSID=C001 APPLID=A06C001
   DEFINE SUCCESSFUL                            TIME:  00.00.00  DATE: 01.037
 PF 1 HELP 2 COM 3 END             6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 14: Auswahl der Sprache

"Le370" ist keine Sprache sondern eine Laufzeitumgebung. CICS braucht an dieser Stelle in Wirklichkeit nicht die Angabe der Quellsprache unseres Anwendungsprogramms (wir haben es ja bereits übersetzt), sondern die Angabe der Laufzeitumgebung des von uns verwendeten Compilers. Alle modernen z/OS-Compiler verwenden eine gemeinsame Laufzeitumgebung, die den Namen "Le370" trägt. Mit Betätigung der Eingabetaste erscheint der nächste Screen.

```
 OVERTYPE TO MODIFY                                          CICS RELEASE = 0530
  CEDA  DEFine PROGram( COB020   )
   PROGram        : COB020
   Group          : PRAKT20
   DEscription  ==>
   Language     ==> Le370             CObol | Assembler | Le370 | C | Pli
   RELoad       ==> No                No | Yes
   RESident     ==> No                No | Yes
   USAge        ==> Normal            Normal | Transient
   USElpacopy   ==> No                No | Yes
   Status       ==> Enabled           Enabled | Disabled
   RSl            : 00                0-24 | Public
   CEdf         ==> Yes               Yes | No
   DAtalocation ==> Below             Below | Any
   EXECKey      ==> User              User | Cics
   COncurrency  ==> Quasirent         Quasirent | Threadsafe
  REMOTE ATTRIBUTES
   DYnamic      ==> No                No | Yes
+  REMOTESystem ==>

                                                    SYSID=C001 APPLID=A06C001
  DEFINE SUCCESSFUL                              TIME:  00.00.00  DATE: 01.037
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 15: Erfolgreiche Definition

Nach dem Betätigen der Eingabetaste erscheint der Bildschirm in Abbildung 15. Wir verlassen die Definition mit der Eingabe von F3, als Ergebnis davon erscheint "SESSION ENDED" (s. Abbildung 16).

```
CEDA DEFINE PROGRAM(COB020) GROUP(PRAKT20)
 STATUS:  SESSION ENDED
```

Abbildung 16: Definition wurde beendet

In diesen Bildschirm geben wir in die oberste Zeile den nächsten CEDA-Befehl, gefolgt von der Eingabetaste, ein (s. Abbildung 17).

```
CEDA DEFINE TRANS(U020) GROUP(PRAKT20)
 STATUS:  SESSION ENDED
```

Abbildung 17: Definition der Transaktion U020

Unsere Transaktion soll wie alle anderen Transaktionen vom Bildschirm über eine 4-stellige Transaktions-ID aufgerufen werden. Wir wählen hierfür die ID "U020" und teilen diese Wahl mit Hilfe des "CEDA DEFINE"-Befehls mit (s. Abbildung 17). Genauso wie der Mapset "MSET020" und das Programm "COB020" wird dies Bestandteil der Gruppe "PRAKT20". Abschließend betätigen wir die Eingabetaste.

```
DEFINE TRANS(U020) GROUP(PRAKT20)
 OVERTYPE TO MODIFY                                      CICS RELEASE = 0530
  CEDA  DEFine TRANSaction( U020 )
   TRANSaction  ==> U020
   Group        ==> PRAKT20
   DEscription  ==>
   PROGram      ==>
   TWasize      ==> 00000             0-32767
   PROFile      ==> DFHCICST
   PArtitionset ==>
   STAtus       ==> Enabled           Enabled | Disabled
   PRIMedsize     : 00000             0-65520
   TASKDATALoc  ==> Below             Below | Any
   TASKDATAKey  ==> User              User | Cics
   STOrageclear ==> No                No | Yes
   RUnaway      ==> System            System | 0 | 500-2700000
   SHutdown     ==> Disabled          Disabled | Enabled
   ISolate      ==> Yes               Yes | No
   Brexit       ==>
+ REMOTE ATTRIBUTES
  S PROGRAM OR REMOTESYSTEM MUST BE SPECIFIED.
                                                    SYSID=C001 APPLID=A06C001

PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 18: Der Definitions-Screen

CEDA will mehrere Angaben von uns und schlägt eine Reihe von Default-Werten vor (s. Abbildung 18).

```
 DEFINE TRANS(U020) GROUP(PRAKT20)
 OVERTYPE TO MODIFY                                      CICS RELEASE = 0530
  CEDA  DEFine TRANSaction( U020 )
   TRANSaction  ==> U020
   Group        ==> PRAKT20
   DEscription  ==>
   PROGram      ==> COB020
   TWasize      ==> 00000             0-32767
   PROFile      ==> DFHCICST
   PArtitionset ==>
   STAtus       ==> Enabled           Enabled | Disabled
   PRIMedsize     : 00000             0-65520
   TASKDATALoc  ==> Below             Below | Any
   TASKDATAKey  ==> User              User | Cics
   STOrageclear ==> No                No | Yes
   RUnaway      ==> System            System | 0 | 500-2700000
   SHutdown     ==> Disabled          Disabled | Enabled
   ISolate      ==> Yes               Yes | No
   Brexit       ==>
+ REMOTE ATTRIBUTES
  S PROGRAM OR REMOTESYSTEM MUST BE SPECIFIED.
                                                  SYSID=C001 APPLID=A06C001

PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 19: Eingabe der Parameter

Wir übernehmen alle Default-Werte und geben in die Zeile "PROGram" den Namen unseres Anwendungsprogramms, nämlich "COB020" ein und bestätigen mit der Eingabetaste (s. Abbildung 19).

```
 OVERTYPE TO MODIFY                                      CICS RELEASE = 0530
  CEDA  DEFine TRANSaction( U020 )
   TRANSaction    : U020
   Group          : PRAKT20
   DEscription  ==>
   PROGram      ==> COB020
   TWasize      ==> 00000             0-32767
   PROFile      ==> DFHCICST
   PArtitionset ==>
   STAtus       ==> Enabled           Enabled | Disabled
   PRIMedsize     : 00000             0-65520
   TASKDATALoc  ==> Below             Below | Any
   TASKDATAKey  ==> User              User | Cics
   STOrageclear ==> No                No | Yes
   RUnaway      ==> System            System | 0 | 500-2700000
   SHutdown     ==> Disabled          Disabled | Enabled
   ISolate      ==> Yes               Yes | No
   Brexit       ==>
+ REMOTE ATTRIBUTES

                                                  SYSID=C001 APPLID=A06C001
   DEFINE SUCCESSFUL                          TIME:  22.00.13  DATE: 01.037
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 20: Erfolgreiche Definition

Die Meldung "DEFINE SUCCESSFUL" erscheint (s. Abbildung 20).

Wir verlassen dieses Menü mit F3.

```
CEDA DEFINE TRANS(U020) GROUP(PRAKT20)
 STATUS:  SESSION ENDED
```

Abbildung 21: SESSION ENDED

Der Bildschirm in der Abbildung 21 erscheint. Wir haben CICS den Namen unseres Mapsets, Anwendungsprogramms und eine dazugehörige Transaktions-ID bekanntgegeben. Jetzt müssen diese drei Komponenten in die CICS-Programmbibliothek übernommen (installiert) werden.

```
 CEDA INSTALL GROUP(PRAKT20)
  STATUS:  SESSION ENDED
```

Abbildung 22: Aufruf der Installation

Wir geben in die oberste Zeile das CEDA-INSTALL-Kommando ein und drücken die Eingabetaste (s. Abbildung 22).

```
  INSTALL GROUP(PRAKT20)
 OVERTYPE TO MODIFY
  CEDA  Install
   All
   Connection   ==>
   DB2Conn      ==>
   DB2Entry     ==>
   DB2Tran      ==>
   DOctemplate  ==>
   Enqmodel     ==>
   File         ==>
   Journalmodel ==>
   LSrpool      ==>
   Mapset       ==>
   PARTItionset ==>
   PARTNer      ==>
   PROCesstype  ==>
   PROFile      ==>
   PROGram      ==>
+  Requestmodel ==>

                                                SYSID=C001 APPLID=A06C001
  INSTALL SUCCESSFUL                          TIME:  22.02.15  DATE: 01.037
PF 1 HELP       3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 23: Installation war erfolgreich

CEDA bestätigt, dass die Installation erfolgreich war (s. Abbildung 23). Wir verlassen mit F3 dieses Menü.

```
CEDA INSTALL GROUP(PRAKT20)
 STATUS:  SESSION ENDED
```

Abbildung 24: Beendete Installation

Der obige Bildschirm erscheint (s. Abbildung 24). Unsere Transaktion ist als Teil der CICS-Anwendungsbibliothek installiert worden und kann nun aufgerufen und damit ausgeführt werden. Hierzu löschen wir die oberste Zeile (die CEDA-Kommandozeile) ganz und rufen unsere Anwendung auf, indem wir dort unsere Transaktions-ID, nämlich "U020", eingeben.

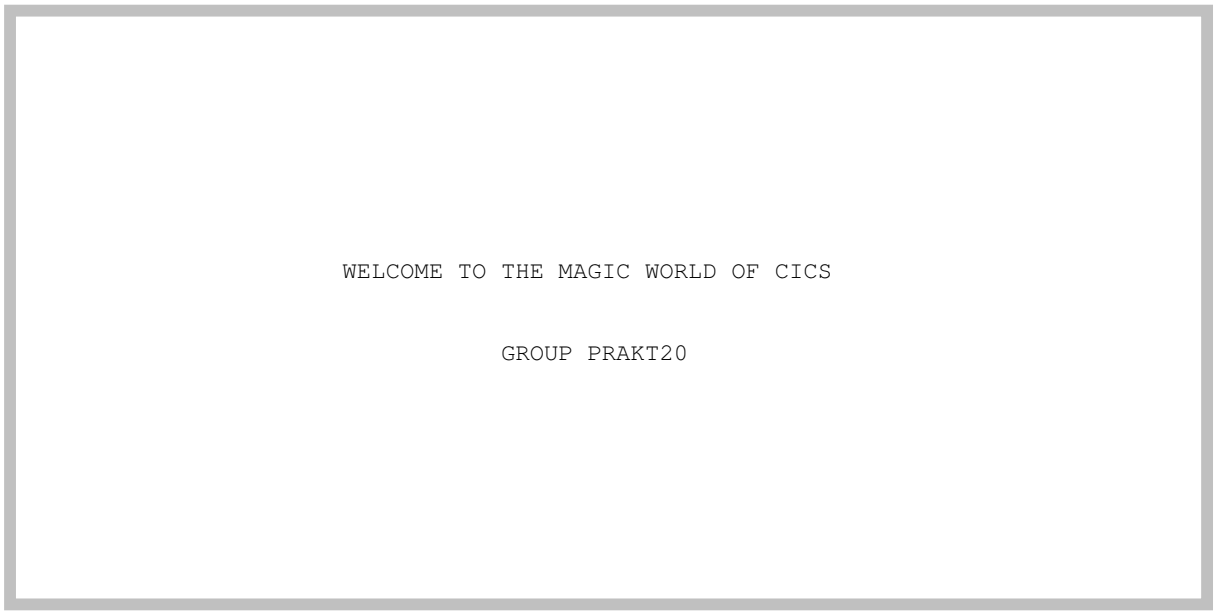

Abbildung 25: Ausgabe der Transaktion "U020" auf dem Bildschirm

Nach dem Betätigen der Eingabetaste erscheint unsere CICS-Transaktion auf dem Monitor (s. Abbildung 25).

Wir betätigen die Eingabetaste, um zum nächsten Screen zu gelangen.

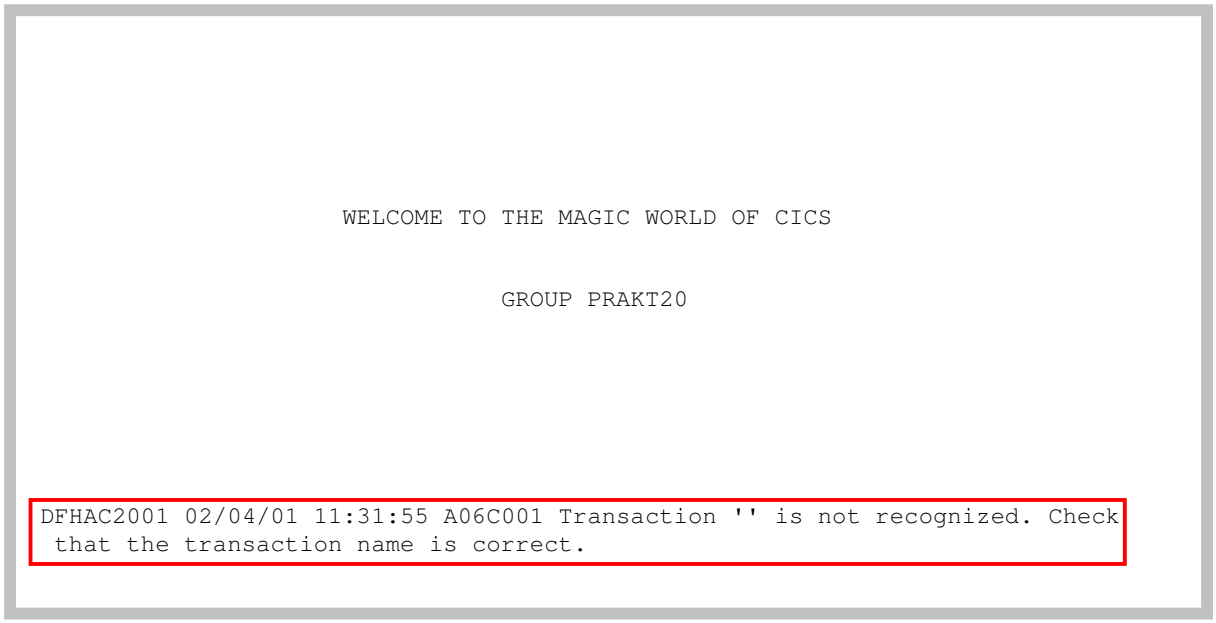

Abbildung 26: Fehlermeldung

Dies terminiert die Bildschirmausgabe unserer Transaktion und erzeugt wieder eine (belanglose) Fehlermeldung (s. Abbildung 26).

```
                    WELCOME TO THE MAGIC WORLD OF CICS

                              GROUP PRAKT20




DFHAC2001 02/04/01 11:31:55 A06C001 Transaction '' is not recognized. Check
 that the transaction name is correct. CEDA DISPLAY GROUP(PRAKT20)
```

Abbildung 27: Ansicht aller gespeicherter Daten in Group "PRAKT20"

Alle Bestandteile unserer Transaktion sind in der Gruppe PRAKT20 gespeichert. Wir schauen sie uns an, indem wir den Befehl "CEDA DISPLAY GROUP(PRAKT20)" eingeben und anschließend die Eingabetaste drücken (s. Abbildung 27).

```
 DISPLAY GROUP(PRAKT20)
 ENTER COMMANDS
  NAME     TYPE         GROUP                                    DATE   TIME
  MSET020  MAPSET       PRAKT20                                  01.034 24.00.00
  COB020   PROGRAM      PRAKT20                                  01.034 24.00.00
  U020     TRANSACTION  PRAKT20                                  01.034 24.00.00









                                                  SYSID=C001 APPLID=A06C001
  RESULTS: 1 TO 3 OF 3                        TIME:  00.00.00  DATE: 01.035
PF 1 HELP       3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 28: Ausgabe aller Komponenten der Gruppe "PRAKT20"

Die Gruppe "PRAKT20" besteht aus den drei Komponenten "MSET020", "COB020" und "U020", die wir unter CICS definiert und anschließend installiert haben.

Aufgabe: *Installieren Sie eine CICS-Transaktion "U<Login-Nr.>, die eine Ausgabe ähnlich der Abbildung 29 liefert und führen Sie diese dann aus. (Hinweis: Die Ausführung der beiden JCL-Scripte erfolgt fehlerfrei, wenn diese mit der Statusmeldung "MAXCC=0" beendet werden). Die Ausgabe Ihrer Transaktion muß unbedingt den* ***Transaktionsnamen****,* ***"TUTORIAL 3 IN COBOL"*** *sowie* ***den Namen oder die Namen der Autoren*** *enthalten. Benutzen Sie als CICS-Gruppennamen Ihren z/OS-Login-Namen (z.B. "PRAKT20" oder "PRAK100").*
Ersetzen Sie in den Bezeichnern für Ihren Mapset, Ihren Map sowie Ihr Cobol-Programm die Ziffern "020" durch die Nummer Ihres Prakt<xx> oder PRAK<xxx>-Accounts. Wenn Sie das nicht tun und mehrere Teilnehmer dieses Tutorial gleichzeitig bearbeiten, kommt es zu sehr unschönen Effekten, wie z.B. dass Sie den Screen von jemand anderem anstelle Ihres eigenen als Ergebnisscreen (Abbildung 29) erhalten.

Aufgabe: *Erzeugen Sie einen Screenshot Ihres Fensters, welches die Bildschirmausgabe von CICS enthält (Ihre Version der Abbildung 29). Achten Sie darauf, dass das Bild nicht mehr als 250 KByte Speicherplatz belegt. Sehr gut ist das JPEG-Format, das mit weniger als 90 KByte auskommt. Schicken Sie Ihrem Tutor dieses Bild im Bitmap- oder JPEG-Format zu. Die Daten Ihrer Arbeit dürfen nicht gelöscht werden, damit der Tutor Ihre Transaktion aufrufen kann.*

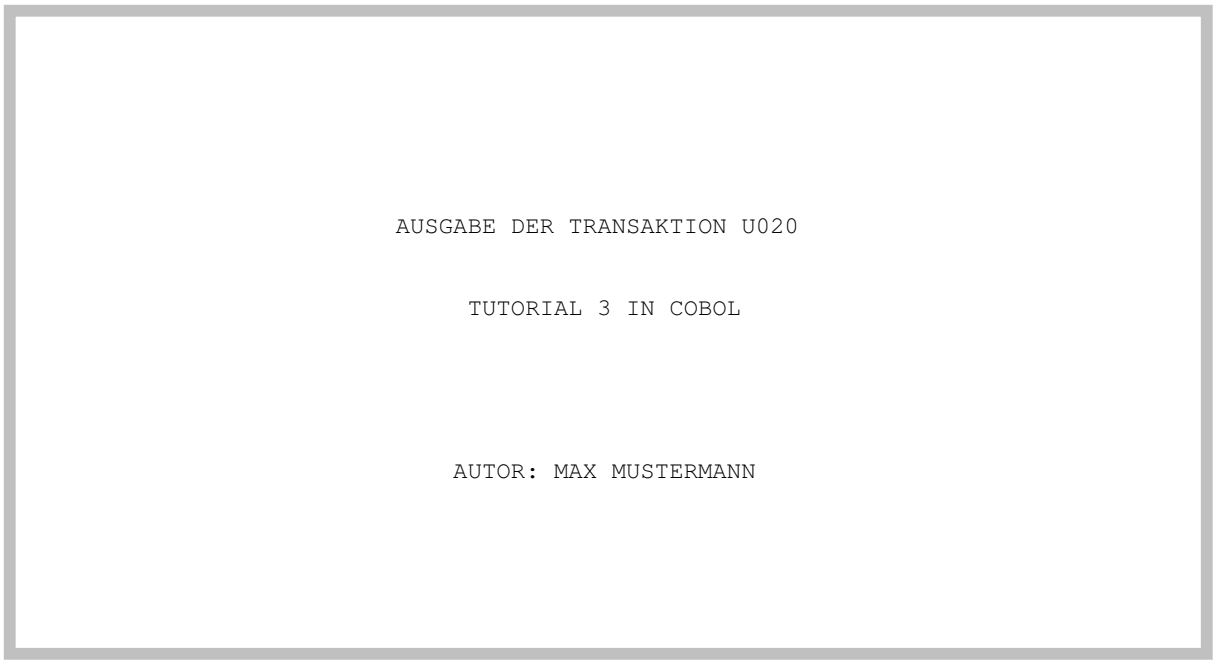

Abbildung 29: Ausgabe der Transaktion U020 der obigen Aufgabe

Um uns aus CICS auszuloggen, betätigen wir so oft die Taste F3, bis die Meldung "STATUS: SESSION ENDED" ausgegeben wird. In die Zeile darüber geben wir die Logoff-Transaktion "CESF LOGOFF", gefolgt von der Eingabetaste, ein (s. Abbildung 30).

Abbildung 30: Ausloggen aus CICS