```
******************************************************************
* Author: Björn Lüpschen
******************************************************************
 IDENTIFICATION DIVISION.
 PROGRAM-ID. GAUSSEINGABE.
 ENVIRONMENT DIVISION.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT FD-MATRIX ASSIGN TO 'eingabe.txt'
     ORGANIZATION IS LINE SEQUENTIAL.

 DATA DIVISION.
 FILE SECTION.
      FD FD-MATRIX.
         01 D-N                      PIC +999.
         01 D-MATRIX-ROW.
            05 D-MATRIX-VALUE        PIC +999.99
               OCCURS 1 TO 100 DEPENDING ON NUMBER-OF-COLUMNS.

 WORKING-STORAGE SECTION.
     01 INPUT-DATA-EOF               PIC X.
     01 NUMBER-OF-COLUMNS            PIC 99 COMP-3.

     01 E-MATRIX.
        05 E-MATRIX-ROW              OCCURS 100.
           10 E-MATRIX-CLM           OCCURS 100.
              15 E-MATRIX-VALUE      PIC -ZZ9.99.

     01 R-MATRIX.
        COPY "MATRIX.CPY" REPLACING ==#== BY ==R==.


     01 ERRORS                       PIC 9.
        88 OUT-OF-MEMORY             VALUE 0.
        88 NOT-SPARSE-MATRIX         VALUE 1.

     01 ERRORS-FOUND                 PIC 9.
        88 ERRORS-FOUND-NO           VALUE 0.
        88 ERRORS-FOUND-YES          VALUE 1.

     01 MAX-ROWS                     PIC 99 COMP-3 VALUE ZERO.
     01 ROW                          PIC 99 COMP-3.
     01 CLM                          PIC 99 COMP-3.
     01 NOT-ZERO-COUNTER             PIC 99999 VALUE ZERO.
     01 NUMBER-OF-ELEMENTS           PIC 99999 VALUE ZERO.
     01 NUMBER-OF-ROWS               PIC 99999 VALUE ZERO.
     01 MAX-NUMBER-OF-ELEMENTS       PIC 99999 VALUE ZERO.
     01 ZAHL                         PIC S999V99 COMP-3.

 PROCEDURE DIVISION.
 MAIN-PROCEDURE.
         PERFORM FORERUN
         PERFORM MAINRUN
         PERFORM LASTRUN

         IF ERRORS-FOUND-YES
            EVALUATE TRUE
             WHEN OUT-OF-MEMORY
                     DISPLAY "ERROR: NICHT GENUG SPEICHER"
             WHEN NOT-SPARSE-MATRIX
                     DISPLAY "ERROR: KEINE DUENN BESETZTE MATRIX"
             WHEN OTHER
                     DISPLAY "ERROR: UNBEKANNTER FEHLER"
            END-EVALUATE
         ELSE

*     Verschieben der Eingabe-Matrix in die Übergabe-Matrix
             PERFORM VARYING ROW
                     FROM 1 BY 1
                     UNTIL ROW > MAX-ROWS
                     AFTER CLM
                     FROM 1 BY 1
                     UNTIL CLM > MAX-ROWS + 1
```

```
                               MOVE E-MATRIX-VALUE(ROW,CLM) TO
                                    R-MATRIX-VALUE(ROW, CLM)
                   END-PERFORM

                   CALL "GAUSSALGO"
                       USING R-MATRIX, MAX-ROWS
               END-IF
               STOP RUN.
       FORERUN.
           OPEN INPUT FD-MATRIX
           MOVE SPACES TO INPUT-DATA-EOF

      *    Lese Spaltenanzahl
           READ FD-MATRIX INTO D-N
               AT END MOVE "C" TO INPUT-DATA-EOF
           END-READ

           MOVE D-N TO NUMBER-OF-COLUMNS

           COMPUTE NUMBER-OF-ROWS = NUMBER-OF-COLUMNS - 1
           COMPUTE NUMBER-OF-ELEMENTS =
                   NUMBER-OF-COLUMNS * NUMBER-OF-ROWS
           COMPUTE MAX-NUMBER-OF-ELEMENTS = NUMBER-OF-ELEMENTS * 0.3

           PERFORM SINGLE-PROCESSING.
       MAINRUN.
               PERFORM SINGLE-PROCESSING until INPUT-DATA-EOF ="C".

       LASTRUN.
               CLOSE FD-MATRIX.

       SINGLE-PROCESSING.
           READ FD-MATRIX INTO D-MATRIX-ROW
               AT END MOVE "C" TO INPUT-DATA-EOF
           END-READ
           PERFORM VARYING ROW FROM 1 BY 1 UNTIL ROW > NUMBER-OF-ROWS
               MOVE D-MATRIX-VALUE(ROW) TO ZAHL
               IF ZAHL NOT EQUALS 0
                    ADD 1 TO NOT-ZERO-COUNTER
                    IF NOT-ZERO-COUNTER > MAX-NUMBER-OF-ELEMENTS
                        SET ERRORS-FOUND-YES TO TRUE
                        SET NOT-SPARSE-MATRIX TO TRUE
                    END-IF
               END-IF
           END-PERFORM
           IF INPUT-DATA-EOF NOT EQUAL "C"
              ADD 1 TO MAX-ROWS
              MOVE D-MATRIX-ROW TO E-MATRIX-ROW(MAX-ROWS)
           END-IF.
```

```
*******************************************************************
* Author: Philipp Kohl
*******************************************************************
 IDENTIFICATION DIVISION.
 PROGRAM-ID. GAUSSALGO.
 DATA DIVISION.
 FILE SECTION.
 WORKING-STORAGE SECTION.

     01 MAX-COLS                             PIC 99 COMP-3.
     01 A-MATRIX.
         05 A-MATRIX-ROW OCCURS 100 INDEXED BY AR.
            10 A-MATRIX-COL OCCURS 100 INDEXED BY AC.
                 15 A-MATRIX-VALUE           PIC -ZZ9.9999.
                 15 FILLER                   PIC XX VALUE ' '.


     01 R-MATRIX.
         COPY "MATRIX.CPY" REPLACING ==#== BY ==R==.

     01 MATRIX-TEMP-ROWCHANGER.
         05 MATRIX-ROW-CHANGER.
          10 EX-MATRIX-VALUE PIC S999V9(10) COMP-3 OCCURS 1 TO 100
          DEPENDING ON MAX-COLS.

     01 EPSILON                PIC 9V9(8) COMP-3 VALUE 0.00000001.

     01 I-COLUMN               PIC 99 COMP-3.
     01 TEMP-MAX               PIC S999V9(10) COMP-3.

     01 MAX-ROW-INDEX          PIC 99 COMP-3.
     01 TEMP-MAX-INDEX         PIC 99 COMP-3.

     01 INDEX-ROW              PIC 99 COMP-3.
     01 INDEX-COL              PIC 99 COMP-3.

     01 INDEX-SWAPFIRST        PIC 99 COMP-3.
     01 INDEX-SWAPSECOND       PIC 99 COMP-3.

     01 SUBTRACT-ROW-FACTOR    PIC S99V9(10) COMP-3.
     01 DIVIDE-ROW-FACTOR      PIC S99V9(10) COMP-3.

     01 PRINT-VALUE            PIC +999.99.

     01 VALUE-NOT-ZERO-COUNTER PIC 9(10).

     77 DEBUG                  PIC 9.
         88 DEBUG-ON           VALUE 0.
         88 DEBUG-OFF          VALUE 1.

     77 RESULT-TYPE            PIC 9.
         88 UNIQUE             VALUE 0.
         88 NOT-SOLVABLE       VALUE 1.

 LINKAGE SECTION.
     01 MATRIX.
        COPY "MATRIX.CPY" REPLACING ==#== BY ==E==.

     01 MAX-ROWS               PIC 99 COMP-3.
 PROCEDURE DIVISION
             USING MATRIX, MAX-ROWS.
 MAIN-PROCEDURE.
     COMPUTE MAX-COLS = MAX-ROWS + 1
     MOVE MATRIX TO R-MATRIX

     SET DEBUG-OFF TO TRUE

     DISPLAY 'Ausgangs Matrix: '
     PERFORM PRINT

      PERFORM VARYING I-COLUMN
              FROM 1 BY 1
              UNTIL I-COLUMN > MAX-COLS - 1
              PERFORM FIND-MAX-ELEMENT-IN-COLUMN

              IF NOT NOT-SOLVABLE
              MOVE TEMP-MAX TO PRINT-VALUE

              IF DEBUG-ON
              DISPLAY 'Maximun in Spalte ' I-COLUMN
              ' betreagt ' PRINT-VALUE
              END-IF

              MOVE I-COLUMN TO INDEX-SWAPFIRST
              MOVE TEMP-MAX-INDEX TO INDEX-SWAPSECOND
              PERFORM SWAP-ROWS

              IF DEBUG-ON
                 PERFORM PRINT
                 DISPLAY ' '
              END-IF

              PERFORM SUBTRACT-ROWS

              IF DEBUG-ON
                 DISPLAY 'Zwischenergebnis'
                 PERFORM PRINT
                 DISPLAY ' '
              END-IF

             END-IF
      END-PERFORM

      PERFORM DIVIDE-ROWS
      DISPLAY 'Ergebnis:'
      PERFORM PRINT

      PERFORM INTERPRET-RESULT

      EVALUATE TRUE
         WHEN UNIQUE
             CALL "GAUSSAUSGABE"
                 USING R-MATRIX, MAX-COLS
         WHEN NOT-SOLVABLE
             DISPLAY 'Nicht (eindeutig) loesbar!'
      END-EVALUATE


      EXIT PROGRAM.

 INTERPRET-RESULT.
     PERFORM VARYING R-I-COL
             FROM 1 BY 1
             UNTIL R-I-COL > MAX-COLS - 1
             IF R-MATRIX-VALUE(MAX-ROWS, R-I-COL) NOT EQUAL 0
                ADD 1 TO VALUE-NOT-ZERO-COUNTER
             END-IF
     END-PERFORM

     IF VALUE-NOT-ZERO-COUNTER NOT EQUAL 1
         SET NOT-SOLVABLE TO TRUE
     END-IF

 .
 DIVIDE-ROWS.
     PERFORM VARYING R-I-ROW FROM MAX-ROWS BY -1 UNTIL R-I-ROW < 1
             AFTER R-I-COL FROM MAX-COLS BY -1 UNTIL R-I-COL < 1
                 COMPUTE R-MATRIX-VALUE(R-I-ROW,R-I-COL) =
                 R-MATRIX-VALUE(R-I-ROW,R-I-COL) /
                 R-MATRIX-VALUE(R-I-ROW,R-I-ROW)

             IF R-MATRIX-VALUE(R-I-ROW, R-I-COL) > 0
                 IF R-MATRIX-VALUE(R-I-ROW, R-I-COL) < EPSILON
                    MOVE 0 TO R-MATRIX-VALUE(R-I-ROW, R-I-COL)
                 END-IF
             ELSE
```

```
                 IF R-MATRIX-VALUE(R-I-ROW, R-I-COL) > EPSILON*-1
                    MOVE 0 TO R-MATRIX-VALUE(R-I-ROW, R-I-COL)
                 END-IF
             END-IF
     END-PERFORM
 .

 SUBTRACT-ROWS.
     PERFORM VARYING R-I-ROW FROM 1 BY 1 UNTIL R-I-ROW > MAX-ROWS

     IF R-I-ROW NOT EQUAL I-COLUMN AND
        R-MATRIX-VALUE(R-I-ROW, I-COLUMN) NOT EQUAL 0
         COMPUTE SUBTRACT-ROW-FACTOR =
                 R-MATRIX-VALUE(R-I-ROW, I-COLUMN)
                 / R-MATRIX-VALUE(I-COLUMN, I-COLUMN)
         PERFORM VARYING R-I-COL
                 FROM 1 BY 1
                 UNTIL R-I-COL > MAX-COLS
             COMPUTE R-MATRIX-VALUE(R-I-ROW, R-I-COL)
                     = R-MATRIX-VALUE(R-I-ROW,R-I-COL)
             - (SUBTRACT-ROW-FACTOR *
             R-MATRIX-VALUE(I-COLUMN,R-I-COL))


             IF R-MATRIX-VALUE(R-I-ROW, R-I-COL) > 0
                 IF R-MATRIX-VALUE(R-I-ROW, R-I-COL) < EPSILON
                     MOVE 0 TO R-MATRIX-VALUE(R-I-ROW, R-I-COL)
                 END-IF
             ELSE
                 IF R-MATRIX-VALUE(R-I-ROW, R-I-COL) > EPSILON*-1
                     MOVE 0 TO R-MATRIX-VALUE(R-I-ROW, R-I-COL)
                 END-IF
             END-IF



         END-PERFORM
     END-IF
     END-PERFORM
     .
 FIND-MAX-ELEMENT-IN-COLUMN.
     MOVE 0 TO TEMP-MAX
     PERFORM VARYING R-I-ROW
             FROM I-COLUMN BY 1
             UNTIL R-I-ROW > MAX-ROWS
*    Betragsmaessig groesstes Element
             IF R-MATRIX-VALUE(R-I-ROW, I-COLUMN)  < 0
                IF R-MATRIX-VALUE(R-I-ROW, I-COLUMN)*-1 > TEMP-MAX
                     COMPUTE TEMP-MAX =
                             R-MATRIX-VALUE(R-I-ROW, I-COLUMN)
                     MOVE R-I-ROW TO TEMP-MAX-INDEX
                END-IF
             ELSE
             IF R-MATRIX-VALUE(R-I-ROW, I-COLUMN) > TEMP-MAX
                MOVE R-MATRIX-VALUE(R-I-ROW, I-COLUMN) TO TEMP-MAX
                MOVE R-I-ROW TO TEMP-MAX-INDEX
             END-IF
     END-PERFORM
     IF TEMP-MAX EQUAL 0
         SET NOT-SOLVABLE TO TRUE
     END-IF
     .

 PRINT.
     PERFORM VARYING INDEX-ROW
             FROM 1 BY 1 UNTIL INDEX-ROW > MAX-ROWS
             AFTER INDEX-COL
             FROM 1 BY 1 UNTIL INDEX-COL > MAX-COLS
             MOVE R-MATRIX-VALUE(INDEX-ROW, INDEX-COL) TO
                  A-MATRIX-VALUE(INDEX-ROW, INDEX-COL)
     END-PERFORM

     PERFORM VARYING INDEX-ROW
```

```
            FROM 1 BY 1 UNTIL INDEX-ROW > MAX-ROWS
            DISPLAY A-MATRIX-ROW(INDEX-ROW)
    END-PERFORM
    .

SWAP-ROWS.
    IF TEMP-MAX NOT EQUAL 0
        IF DEBUG-ON
            DISPLAY 'Vertausche Zeile ' INDEX-SWAPFIRST ' mit '
                INDEX-SWAPSECOND
        END-IF
        MOVE R-MATRIX-ROW(INDEX-SWAPFIRST)
        TO   MATRIX-ROW-CHANGER
        MOVE R-MATRIX-ROW(INDEX-SWAPSECOND)
        TO   R-MATRIX-ROW(INDEX-SWAPFIRST)
        MOVE MATRIX-ROW-CHANGER
        TO   R-MATRIX-ROW(INDEX-SWAPSECOND)
    END-IF
    .
```

```
******************************************************************
* Author: Bjoern Luepschen
******************************************************************
 IDENTIFICATION DIVISION.
 PROGRAM-ID. GAUSSAUSGABE.
 ENVIRONMENT DIVISION.

 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT MATRIX-ROW ASSIGN TO "ausgabe.txt"
     ORGANIZATION IS LINE SEQUENTIAL
     ACCESS IS SEQUENTIAL.

 DATA DIVISION.

 FILE SECTION.
     FD MATRIX-ROW.
     01 MATRIX.
         05 D-MATRIX-ROW
             OCCURS 100.
             10 D-MATRIX-VALUE           PIC -ZZ9.999.
         05 FILL                         PIC XX VALUE X'0D0A'.


 WORKING-STORAGE SECTION.
     01 INDEX-ROW                        PIC 999.
     01 INDEX-COL                        PIC 999.
     01 MAX-ROWS                         PIC 99.
 LINKAGE SECTION.
     01 R-MATRIX.
      COPY "MATRIX.CPY" REPLACING ==#== BY ==R==.
     01 NUMBER-OF-COLUMNS                PIC 99 COMP-3.

 PROCEDURE DIVISION
     USING R-MATRIX, NUMBER-OF-COLUMNS.

     MAIN-PROCEDURE.
     OPEN OUTPUT MATRIX-ROW

     COMPUTE MAX-ROWS = NUMBER-OF-COLUMNS - 1

     PERFORM VARYING INDEX-ROW
             FROM 1 BY 1
             UNTIL INDEX-ROW > MAX-ROWS

             PERFORM VARYING INDEX-COL
             FROM 1 BY 1
             UNTIL INDEX-COL > MAX-ROWS + 1
                 MOVE R-MATRIX-VALUE(INDEX-ROW, INDEX-COL)
                 TO D-MATRIX-VALUE(INDEX-COL)

             END-PERFORM

             MOVE X'0D0A' TO FILL

             WRITE MATRIX
             END-WRITE
     END-PERFORM



     CLOSE MATRIX-ROW
     .
     END PROGRAM GAUSSAUSGABE.
```