NAME

PHASE0 – Pivot in a linear programming tableau to get identity columns with zero costs.
That's a zero in the name of the routine.

SYNOPSIS

CALL PHASE0(T,LDT,BASIS,MP,N,ROWS,MR, RC)

T(LDT,*)	is the REAL*8 input LP tableau
LDT	is the INTEGER*4 leading dimension of T
BASIS(N)	is the INTEGER*4 output vector of basic column row indices, 0 for nonbasic
MP	is the INTEGER*4 total number of rows in the LP including the objective
N	is the INTEGER*4 number of variables in the LP
ROWS(MR)	is the INTEGER*4 vector of row indices in the problem
MR	is the INTEGER*4 input number of rows in use including the objective
RC	is the INTEGER*4 return code; 0 => ok, 1 => infeasible form 1

DESCRIPTION

The routine assumes there is no basis to begin with. It examines the constraint rows in order going down the tableau. For each row it finds the leftmost nonzero entry and calls PIVOT to pivot on it. Each pivot updates T and BASIS. If all of the entries in the row are smaller in absolute value than 1.D-06 but the corresponding constant column entry is not, the routine considers the problem to be in infeasible form 1 and returns with RC=1. If every entry in a row is smaller in absolute value than 1.D-06, it considers the row to be all zeros and removes it from the problem.

DIAGNOSTICS

On output these are the possible RC values:

0 all went well
1 the problem is in infeasible form 1

LINKAGE

gfortran source.f –L${HOME}/lib –lmisc

AUTHOR

Michael Kupferschmid

EXAMPLE

```
      PARAMETER(LDT=3,MP=3,N=5,MR=3)
      INTEGER*4 ROWS(MR)/1,2,3/,BASIS(5),RC
      REAL*8 T(LDT,1+N)/0.D0, 6.D0, 5.D0,
     ;                  1.D0, 1.D0,-1.D0,
     ;                  0.D0, 0.D0, 1.D0,
     ;                  2.D0,-1.D0, 0.D0,
     ;                 -1.D0,-3.D0, 3.D0,
     ;                  4.D0, 1.D0,-3.D0/
      DO 1 I=1,MP
           WRITE(6,901) (T(I,J),J=1,1+N)
901        FORMAT(6(1X,F4.1))
  1   CONTINUE
      CALL PHASE0(T,LDT,BASIS,MP,N,ROWS,MR, RC)
      WRITE(6,900)
900   FORMAT(' ')
      DO 2 I=1,MP
           WRITE(6,901) (T(I,J),J=1,1+N)
  2   CONTINUE
      STOP
      END
```

This example (which is from [1] Exercise 2.10.68) produced the following output:

```
unix[1] a.out
  0.0  1.0  0.0  2.0 -1.0  4.0
  6.0  1.0  0.0 -1.0 -3.0  1.0
  5.0 -1.0  1.0  0.0  3.0 -3.0

 -6.0  0.0  0.0  3.0  2.0  3.0
  6.0  1.0  0.0 -1.0 -3.0  1.0
 11.0  0.0  1.0 -1.0  0.0 -2.0
unix[2]
```

REFERENCE

[1] Kupferschmid, Michael, "Introduction to Mathematical Programming"