```
! LINGO model for sequencing jobs in a flow shop.
  Find the order, sequence,  or permutation in which to process jobs in a Flow shop.
  Assume that same sequence is used on all machines;
! Keywords: Flow shop, LINGO, Permutation, Scheduling, Sequencing, Taillard;
SETS:
 JOB;
 JINP( JOB, JOB): Y;
 MACH;
 MXP( MACH, JOB): PT, FT;
ENDSETS
DATA:
! A small data set;
!LIL;  MACH = A..C; ! The set of machines;
!LIL;   JOB = 1..6; ! The set of jobs;
! Processing times, machine x job;
!LIL;    PT =
       9 3 4 8 2 1
       3 7 6 2 4 5
       6 5 2 4 8 3;

! Data from Taillard:
   http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/ordonnancement.html;
!Tail  MACH = MCH1..MCH5; ! The set of machines;
!Tail  JOB = JB01..JB20;  ! The set of jobs;
! Processing times, machine x job;
! 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20;
!Tail  PT =
 54 83 15 71 77 36 53 38 27 87 76 91 14 29 12 77 32 87 68 94
 79  3 11 99 56 70 99 60  5 56  3 61 73 75 47 14 21 86  5 77
 16 89 49 15 89 45 60 23 57 64  7  1 63 41 63 47 26 75 77 40
 66 58 31 68 78 91 13 59 49 85 85  9 39 41 56 40 54 77 51 31
 58 56 20 85 53 35 53 41 69 13 86 72  8 49 47 87 58 18 68 28;
ENDDATA
! Variables:
   Y( j, p) = 1 if job j is assigned to position p,
   FT( m, p) = finish time on machine m of job in position p;

 @FOR( JINP( j, p): @BIN( Y(  j,p))); ! The Y's = 0 or 1;

 nm = @SIZE(MACH);
 nj = @SIZE(JOB);
 !MIN = @SUM( JOB(p): FT(nm, p)); ! Minimize sum of finish times on last machine;
 MIN = FT( nm, nj); ! Minimize makespan;

 ! Each job must be assigned a position;
 @FOR( JOB( j):
   @SUM( JOB( p): Y( j, p)) = 1;
     );
 ! Each position gets a job;
  @FOR( JOB( p):
    @SUM( JOB( j): Y( j, p)) = 1;
      );

 ! Finish time on machine m of job in position p;
 ! Jobs on machine 1 have no predecessors;
  FT( 1, 1) >= @SUM( JOB( j): PT( 1, j)* Y( j, 1));
  @FOR( JOB( p) | p #GT# 1:
 ! Job in position p on machine 1 cannot start until job in position p-1 finishes;
    FT( 1, p) >= FT( 1, p-1) + @SUM( JOB( j): PT( 1, j)* Y( j, p));
      );

 @FOR( MACH(m) | m #GT# 1:
 ! Job in position 1 on machine m cannot start until it finishes on m-1;
  FT( m, 1) >= FT( m-1, 1) + @SUM( JOB( j): PT( m, j)* Y( j, 1));

  @FOR( JOB( p) | p #GT# 1:
 ! Job in position p cannot start on m until job position p-1
    finishes on m;
```

```
   FT( m, p) >= FT( m, p-1) + @SUM( JOB(j): PT( m, j)* Y( j, p));
! Job in position p on machine m cannot start until it finishes on m-1.
 Exploit observation that job in position p on m is also the only job
 in position p on machine m-1;
   FT( m, p) >= FT( m-1, p) + @SUM( JOB( j): PT( m, j)* Y( j, p));
     );
    );
```