```
! A model in the LINGO optimization modeling language
 of the multi-dimensional knapsack problem;
! The one dimension knapsack problem is:
   Given
     a container (the knapsack) of given size or capacity, and
     a set of candidate items, each with a given value and size,
   Choose which items to place in the knapsack so as to:
     Maximize the sum of the values of the items placed in the knapsack,
      subject to
     the sum of the sizes of the placed items does not exceed knapsack size.
Variations of the knapsack items appear in many applications.
 The loading or packing of a truck, cutting required smaller lengths of steel rods
 or cables from a longer stock length,
 cutting wide stock rolls of paper into smaller customer requested sizes,
 which songs to place on the side of a musical record of limited size, etc.;
! Keywords: Bin packing, Cable cutting, Cutting stock, Knapsack, LINGO, Loading, Multi-
dimensional knapsack,
  Music, Packing, Paper, Steel, Truck loading;
SETS:
   ITEM : VAL, Z, NUMAVAIL;
   DIM: CAP;
   IXD( ITEM, DIM): DSIZE;
ENDSETS
DATA:
 ! The items and their values;
  ITEM  = I1  I2  I3  I4  I5  I6  I7  I8;
  VAL   =  8   4   5   3   6   8   9   3;
! and number available to load of each item;
 NUMAVAIL= 1   2   4   2   1   1   3   1;
 ! The names of the dimensions of capacity and
   the knapsack capacity for each;
   DIM  = WEIGHT   VOLUME;
    CAP =  10000      36;
  ! Sizes of items on each dimension;
       DSIZE =
           2100       4
           1500       5
           1700       6
            900       8
           2200       3
           1100       6
           1600       9
           1400       2;
ENDDATA
! Variables:
     Z(j) = 1 if item j is loaded on truck;

! Maximize the value of items loaded;
MAX = @SUM( ITEM( j) : VAL( j) * Z( j));

! For each capacity type k, cannot exceed knapsack capacity;
 @FOR( DIM( k):
     @SUM( ITEM( j): DSIZE( j, k) * Z( j)) <= CAP( k);
     );

@FOR( ITEM( j):
! The Z( j) must be general integers: 0, 1, 2, ...;
     @GIN( Z( j));
! Cannot load more of each item than number available;
     Z(j) <= NUMAVAIL( j);
  );
```