付録1　FDLの文法

’X’ は終端記号Xを示す。
[X]は、Xがひとつ存在するか、省略して空であることを示す。

```
〈class definition〉::=
    'class' 〈class name〉
        ['super' 〈super names〉 ';']
        ['part' 〈part definition〉 ';']
        ['parameter' 〈parameter definition〉 ';']
        ['database'  〈item definition〉';' 〈retrieval formula〉';']
        ['where' 〈constraint definition〉';']
        ['method'  〈method description〉]
        ['local'  〈local method description〉]
    'end' '.'


〈super names〉::=  〈class name〉 | 〈class name〉','〈super names〉
〈class name〉::= 〈atom〉
〈part definition〉::=
            〈part definition〉','〈part definition〉
          | 〈part slot name〉 ':=' '#'〈part class〉
〈parameter definition〉::=
            〈parameter definition〉','〈parameter definition〉
          | 〈parameter slot name〉 〈type〉 〈single attribute constraint〉
〈part slot name〉::= 〈atom〉
〈part class〉::= 〈atom〉
〈parameter slot name〉::= 〈atom〉
〈type〉::= 'real'|'integer'|'number'|'string'|'list'
〈single attribute constraint〉:==  '[]' | '[' 〈cons〉']'
〈cons〉:== [  'unit('〈atom〉')'',']〈cons1〉
〈cons1〉:== 'member('〈member list〉')' | 〈cons2〉
〈cons2〉:==
             'module('〈module〉','〈residual〉')'
          | ['module('〈module〉','〈residual〉')'',']〈cons3〉
〈cons3〉:==
            '>'〈number〉| '>='〈number〉
          | ['>'〈number〉]〈cons4〉| ['>='〈number〉]〈cons4〉
〈cons4〉:== '<'〈number〉 | '=<'〈number〉
          |
〈module〉::= 〈integer〉
〈residual〉::= 〈integer〉
〈member list〉::=
           '['〈members〉']'
〈members〉::=  〈integers〉|〈reals〉|〈numbers〉|〈strings〉
〈integers〉::= 〈integer〉|〈integer〉','〈integers〉
〈reals〉::= 〈real〉|〈real〉','〈reals〉
〈numbers〉::= 〈number〉|〈number〉','〈numbers〉
〈strings〉::= 〈string〉|〈string〉','〈strings〉
〈item definition〉::=
            〈item definition〉','〈item defition〉
          | 〈item name〉 〈item type〉 〈unit〉
〈item name〉::= 〈atom〉
〈item type〉::= 'real'|'integer'|'string'
〈unit〉::= 〈string〉
〈retrieval formula〉::=
             <retrieval formula>','<retrieval formula>
           | <formula> <arithmatic operator> <formula>
<arithmatic operator>::=
             '<' | '=<' | '>' | '>=' | '='
<formula>::=
              <number>
           |  <parameter slot>
           |  'd!'<item name>
           |  <variable>
           |  'sqrt('<formula>')'
           |  'floating_point_to_integer('<formula>')'
           |  'floor('<formula>')'
           |  'ceiling('<formula>')'
           |  '**('<formula>','<number>')'
           |  'bigger('<formula>','<formula>')'
           |  'smaller('<formula>','<formula>')'
           |  'sin('<formula>')'
           |  'cos('<formula>')'
           |  'tan('<formula>')'
           |  <formula>  '+' <formula>
           |  <formula>  '-' <formula>
           |  <formula>  '*' <formula>
           |  <formula>  '/' <formula>
<constraint definition>::=
             <constraint definition>','<constraint definition>
           | <identity constraint>
           | <multiple attribute constraint>
<identity constraint>:==
             <slot definition>  '==' <slot definition>';'
<slot definition>::=
             <part slot name>'!'<slot definition>
           | <part slot name>
           | <parameter slot name>
<multiple attribute constraint>::=
             <constraint>';'
           | <precondition>  '==>' <constraint>';'
<precondition>::= <constraint>
<constraint>::=
             <constraint>','<constraint>
           | <constraint equation>
           | <constraint inequality>
           | <prolog predicate>
<method description>::=
             <method description>, <method description>
           | ':'<prolog head>  ':-' <prolog body>';'
           | ':'<prolog head>';'
<local method description>::=
             <local method description>,<local method description>
           | <prolog head>  ':-' <prolog body>';'
           | <prolog head>';'

<constraint inequality>::=
              <polynomial>  '<'  <polynomial>
           |  <polynomial>  '=<' <polynomial>
           |  <polynomial>  '>'  <polynomial>
           |  <polynomial>  '>=' <polynomial>
<constraint equation>::=
              <polynomial>  '=' <polynomial>
<polynomial>::=
              <number>
```

```
            | <parameter slot>
            | <prefix><parameter slot>
            | <variable>
            | 'sqrt ('<polynomial>') '
            | 'floating_point_to_integer ('<polynomial>') '
            | 'floor ('<polynomial>') '
            | 'ceiling ('<polynomial>') '
            | '** ('<polynomial>','<number>') '
            | 'bigger ('<polynomial>','<polynomial>') '
            | 'smaller ('<polynomial>','<polynomial>') '
            | 'sin ('<polynomial>') '
            | 'cos ('<polynomial>') '
            | 'tan ('<polynomial>') '
            | 'cot ('<polynomial>') '
            | 'sinh ('<polynomial>') '
            | 'cosh ('<polynomial>') '
            | 'tanh ('<polynomial>') '
            | 'coth ('<polynomial>') '
            | 'asin ('<polynomial>') '
            | 'acos ('<polynomial>') '
            | 'acot ('<polynomial>') '
            | 'asinh ('<polynomial>') '
            | 'acosh ('<polynomial>') '
            | 'atanh ('<polynomial>') '
            | 'acoth ('<polynomial>') '
            | 'log ('<polynomial>') '
            | 'log ('<polynomial>','<polynomial>') '
            | 'exp ('<polynomial>') '
            | <polynomial>  '+' <polynomial>
            | <polynomial>  '-' <polynomial>
            | <polynomial>  '*' <polynomial>
            | <polynomial>  '/' <polynomial>
<parameter slot>::=
              <parameter slot name>
            | <part slot name>'!'<parameter slot>
<prefix>::= '@' | '^'
<number>::= <integer>|<real>
```

付録２　FDL 演算子順位表

ＦＤＬのクラス記述は、ここに示される演算子順位により、文法解釈される。

```
:- op(1180, fx, (class)).
:- op(1159, fx, (super)).
:- op(1159, xfy, (super)).
:- op(1158, fx, (part)).
:- op(1158, xfx, (part)).
:- op(1157, fx, (parameter)).
:- op(1157, xfx, (parameter)).
:- op(1156, fx, (database)).
:- op(1156, xfx, (database)).
:- op(1155, fx, (where)).
:- op(1155, xfx, (where)).
:- op(1154, fx, (method)).
:- op(1154, xfx, (method)).
:- op(1153, fx, (local)).
:- op(1153, xfy, (local)).
:- op(1100, xf, (;)).
:- op(1050, xf, (:-)).
:- op(1050, xfx, (:-)).
:- op(1050, fx, (?-)).
:- op(1040, xfx, (==>)).
:- op(700, xfx, (:=)).
:- op(700, xfx, (from)).
:- op(700, fx, (=)).
:- op(700, fx, (<)).
:- op(700, fx, (>)).
:- op(700, fx, (=<)).
:- op(700, fx, (>=)).
:- op(500, xfx, =>).
:- op(500, xfx, <=).
:- op(300, xfy, (**)).
:- op(200, fx, (^)).
:- op(200, xfy, (!)).
:- op(100, fy, (:)).
:- op(100, xfx, (:)).
:- op(100, fx, (not)).
:- op(80, xf, (real)).
:- op(80, xfx, (real)).
:- op(80, xf, (integer)).
:- op(80, xfy, (integer)).
:- op(80, xf, (number)).
:- op(80, xfy, (number)).
:- op(80, xf, (string)).
:- op(80, xfy, (string)).
:- op(80, xf, (list)).
:- op(80, xfy, (list)).
:- op(80, xf, (internal)).
:- op(80, xfy, (internal)).
:- op(80, fy, (@)).
:- op(80, fy, (#)).
:- op(80, fy, (##)).
:- op(80, xfx, (#)).
:- op(80, xfx, (##)).
```

付録３　FDLの内部表現

ＦＤＬのクラス、インスタンスオブジェクトは、ＦＤＬインタプリタ内では Prolog 述語の集合により表現されている。これを、ＦＤＬの内部表現と呼ぶことにする。ここでは、(1)で FDL によるクラス記述例とそこから生成されたインスタンスの例を示し、(2)でクラスの内部表現、(3)でインスタンスの内部表現を、簡単な説明と共に示す。

（１）ＦＤＬ記述例

```
class shaft
   part   material:= #material;
   parameter
        max_rot integer [unit(rpm)],
        out_dia integer [unit(mm), >= 30],
        int_dia integer [unit(mm), >= 0],
        right_dia integer [unit(mm), >= 30],
        left_dia integer [unit(mm), >= 25];
end.

class main_shaft
   super shaft;
   where
        out_dia == right_dia;
        right_dia = left_dia + 5;
        out_dia >= int_dia + 5;
end.

class material
   parameter  name string [],
              elasticity real [unit('Kg/mm2')],
              shearing   real [unit('Kg/mm2')],
              poisson    real [],
              specific_num real [];
   where
         data(@name,^elasticity,^shearing,^poisson,^specific_num);
  local
   data(steel,21000,8200,0.3,7.9);
   data(aluminum,7200,2760,0.34,2.7);
   data(brass, 6300,2400,0.34,8.0);
end.

main_shaft#1
part
  material   material#2
parameter
  max_rot                int#8     nil   [rpm]
  out_dia                int#9    @nil   [mm]  [>=30]
  int_dia                int#10   @nil   [mm]  [>=0]
  right_dia              int#9    @nil   [mm]  [>=30]
  left_dia               int#12    nil   [mm]  [>=25]
where
  out_dia==right_dia;
  right_dia=left_dia+5;    %(1)        out_dia>=int_dia+5;      %(2)
end

material#2
owner
  main_shaft#1-material
parameter
  name                    str#3    @nil
  elasticity                real#4    ^nil  [Kg/mm2]
  shearing                  real#5    ^nil  [Kg/mm2]
  poisson                   real#6    ^nil
  specific_num              real#7    ^nil
end
```

（２） ＦＤＬクラスの内部表現

・クラス'shaft'の内部表現

```
is_a(shaft, []).
class(shaft, 'PART_ORDER_ORIGIN', [material]).
class(shaft, 'PARAMETER_ORDER_ORIGIN', [max_rot,out_dia,int_dia,right_dia,left_dia]).
class(shaft, 'SUPER_ALL', [shaft]).
class(shaft, 'PART_ORDER', [shaft-material]).
class(shaft, 'PARAMETER_ORDER', [shaft-max_rot,shaft-out_dia,shaft-int_dia,shaft-right_dia,shaft-left_dia]).
class(shaft, 'PART', material, #material).
class(shaft, 'PARAMETER', max_rot-integer, [unit(rpm)]).
class(shaft, 'PARAMETER', out_dia-integer, [unit(mm),>=30]).
class(shaft, 'PARAMETER', int_dia-integer, [unit(mm),>=0]).
class(shaft, 'PARAMETER', right_dia-integer, [unit(mm),>=30]).
class(shaft, 'PARAMETER', left_dia-integer, [unit(mm),>=25]).
:number_of_cons(shaft#_,0) :- !.
```

・クラス'main_shaft'の内部表現

```
is_a(main_shaft, [shaft]).
class(main_shaft, 'PART_ORDER_ORIGIN', []).
class(main_shaft, 'PARAMETER_ORDER_ORIGIN', []).     %オリジナルのパラメータスロット情報
class(main_shaft, 'SUPER_ALL', [shaft,main_shaft]).
class(main_shaft, 'PART_ORDER', [shaft-material]).
class(main_shaft, 'PARAMETER_ORDER',                %継承も加えたパラメータスロット情報
 [shaft-max_rot,shaft-out_dia,shaft-int_dia,shaft-right_dia,shaft-left_dia]).
coupling(main_shaft, out_dia==right_dia).                       %同一性制約の内部表現
:number_of_cons(main_shaft#_,2) :- !.                %クラス内の制約数の記録。制約起動制御に利用
:prefix(main_shaft#A) :-                             %インスタンスごとにパラメータの prefix をセッ
        :set_prefix(main_shaft#A,out_dia,1),          トするために必要な情報
        :set_prefix(main_shaft#A,int_dia,1).
:constraint(main_shaft#E,F,G,2) :-
        (   F==out_dia,  G=D;   F==int_dia,   G=C),
        :get_slot(main_shaft#E,out_dia,B),
        :get_by_var(B,D),
        atomic(D),
        :get_slot(main_shaft#E,int_dia,A),
        :get_by_var(A,C),
        atomic(C),
        :occ_check(#fdl_memo,main_shaft#E,2,F),
        :set_flag(#fdl_memo,1),
        eval([D>=C+5]),
        write_equation([D>=C+5]),
        (    retract(user:constraint(main_shaft,_,2))  ;   true),
        assert(user:constraint(main_shaft#E,[D>=C+5],2)),
        :add_check(#fdl_memo,main_shaft#E,2,F),
        :set(main_shaft#E!out_dia,D),
        :set(main_shaft#E!int_dia,C),
        :owners_constraint(main_shaft#E!out_dia,D),
        :owners_constraint(main_shaft#E!int_dia,C),
        :set_flag(#fdl_memo,0).
:constraint(main_shaft#E,F,G,1) :-                       %制約内部記述の簡略な説明
        (    F==right_dia,                             ↑ ・値の変更を受けたパラメータ名が
             G=D                                       |   'right_dia,が left_dia の場合のみ、こ
        ;   F==left_dia, G=C),                         |   の制約を起動するための条件
```

```
            :occ_check (#fdl_memo,main_shaft#E,1,F) ,              ・同一制約が繰り返し起動されない条件
            :set_flag (#fdl_memo,1) ,                              ・制約は起動済みで未解決状態を記録
            :get_slot (main_shaft#E,right_dia,B) ,                 ・right_dia, left_dia について、@付き
            :get_fixed_value (B,D) ,                                または[]で囲って確定された値が
            :get_slot (main_shaft#E,left_dia,A) ,                  すでに与えられている場合、その値
            :get_fixed_value (A,C) ,                               をパラメータスロットから取り出す
            eval ( [D=C+5] ) ,                                     ・制約ソルバーの呼び出し
            write_equation ( [D=C+5] ) ,                           ・制約解のインタプリタへの表示
            (     retract (user:constraint (main_shaft,_,1) )      ・制約解を記録する。why 機能で
            ;     true ) ,                                           必要となる。
            assert (user:constraint (main_shaft#E, [D=C+5] ,1) ) ,
            :add_check (#fdl_memo,main_shaft#E,1,F) ,              ・この制約が起動されたことを記録
            :set (main_shaft#E!right_dia,D) ,                       ・制約解決により求まった値をパラメ
            :set (main_shaft#E!left_dia,C) ,                       ータスロットに格納
            :owners_constraint (main_shaft#E!right_dia,D) ,         ・制約伝播のために、関連する制約
            :owners_constraint (main_shaft#E!left_dia,C) ,            を調べ、起動する
            :set_flag (#fdl_memo,0) .                              ・この制約解決の完全な成功を記録

%クラス material の記録
is_a (material, [] ) .
class (material, 'PART_ORDER_ORIGIN', [] ) .
class (material, 'PARAMETER_ORDER_ORIGIN', [name,elasticity,shearing,poisson,specific_num] ) .
class (material, 'SUPER_ALL', [material] ) .
class (material, 'PART_ORDER', [] ) .
class (material, 'PARAMETER_ORDER',
      [material-name,material-elasticity,material-shearing,material-poisson,material-specific_num] ) .
class (material, 'PARAMETER', name-string, [] ) .
class (material, 'PARAMETER', elasticity-real, [unit ('Kg/mm2') ] ) .
class (material, 'PARAMETER', shearing-real, [unit ('Kg/mm2') ] ) .
class (material, 'PARAMETER', poisson-real, [] ) .
class (material, 'PARAMETER', specific_num-real, [] ) .
material local data (steel,21000,8200,0.3,7.9) .
material local data (aluminum,7200,2760,0.34,2.7) .
material local data (brass,6300,2400,0.34,8.0) .
:number_of_cons (material#_,1)  :- !.
:prefix (material#A)  :-
            :set_prefix (material#A,name,1) ,
            :set_prefix (material#A,elasticity,2) ,
            :set_prefix (material#A,shearing,2) ,
            :set_prefix (material#A,poisson,2) ,
            :set_prefix (material#A,specific_num,2) .
:constraint (material#G,H,I,1)  :-
            H==name,
            I=F,
            :get_slot (material#G,name,E) ,
            :get_by_var (E,F) ,
            atomic (F) ,
            :occ_check (#fdl_memo,material#G,1,H) ,
            :set_flag (#fdl_memo,1) ,
            (material local data (F,A,B,C,D) ) ,
            write_equation ( [] ) ,
            (     retract (user:constraint (material,_,1) )   ;    true) ,
            assert (user:constraint (material#G, [] ,1) ) ,
            :add_check (#fdl_memo,material#G,1,H) ,
            :set (material#G!name,F) ,
            :set (material#G!elasticity,A) ,
            :set (material#G!shearing,B) ,
            :set (material#G!poisson,C) ,
            :set (material#G!specific_num,D) ,
          :owners_constraint（material#G!name,F）,
          :owners_constraint（material#G!elasticity,A）,
          :owners_constraint（material#G!shearing,B）,
          :owners_constraint（material#G!poisson,C）,
          :owners_constraint（material#G!specific_num,D）,
          :set_flag（#fdl_memo,0）.
```

（3）　ＦＤＬインスタンスの内部表現

```
%インスタンス基本構造記述
%第１項：インスタンス名、第２項、スロット名、第３項：スロット格納物
instance（material#2, name, string#3）.
instance（material#2, elasticity, real#4）.
instance（material#2, shearing, real#5）.
instance（material#2, poisson, real#6）.
instance（material#2, specific_num, real#7）.
instance（main_shaft#1, material, material#2）.
instance（main_shaft#1, max_rot, integer#8）.
instance（main_shaft#1, out_dia, integer#9）.
instance（main_shaft#1, int_dia, integer#10）.
instance（main_shaft#1, left_dia, integer#12）.
instance（main_shaft#1, right_dia, integer#9）.

%パラメータの値記述
%第１項：パラメータ名、第２項：値、第３項：free（値変更可能）または fix（値固定）
type（string#3, nil, free）.
type（real#4, nil, free）.
type（real#5, nil, free）.
type（real#6, nil, free）.
type（real#7, nil, free）.
type（integer#8, nil, free）.
type（integer#10, nil, free）.
type（integer#12, nil, free）.
type（integer#9, nil, free）.

%パラメータの単一属性制約の記述
%第１項：パラメータ名、第２項：１（入出力両方向）、２（入力専用）、３（出力専用）、
  第３項：単位記述、第４項：デフォルト値記述、第５項：値存在範囲の記述
type（string#3, 1, [], [], []）.
type（real#4, 2, unit（'Kg/mm2'）, [], []）.
type（real#5, 2, unit（'Kg/mm2'）, [], []）.
type（real#6, 2, [], [], []）.
type（real#7, 2, [], [], []）.
type（integer#8, 0, unit（rpm）, [], []）.
type（integer#12, 0, unit（mm）, [], [>=25]）.
type（integer#9, 1, unit（mm）, [], [>=30]）.
type（integer#10, 1, unit（mm）, [], [>=0]）.

%パラメータの所属場所（持ち主）
owner（string#3, ［material#2-name］）.
owner（real#4, ［material#2-elasticity］）.
owner（real#5, ［material#2-shearing］）.
owner（real#6, ［material#2-poisson］）.
owner（real#7, ［material#2-specific_num］）.
owner（material#2, ［main_shaft#1-material］）.
owner（integer#8, ［main_shaft#1-max_rot］）.
owner（integer#10, ［main_shaft#1-int_dia］）.
owner（integer#12, ［main_shaft#1-left_dia］）.
owner（integer#9, ［main_shaft#1-out_dia,main_shaft#1-right_dia］）.
```

付録4　FDLによる旋盤設計モデル記述

```
class work_cutting
parameter
      max_work_dia integer 'mm',
      max_drill_dia integer [unit('mm')],
      drill_cut_spd integer [unit('m/min'),default(30)],
      max_cut_spd integer [unit('m/min'),default(150)],
      rel_cut_friction integer [unit('kg/mm2'),default(200)],
      max_cut_area real 'mm2',
      max_cut_force integer 'kg',
      cut_power real 'KW',
      max_cut_trq real 'kg.m',
      max_drill_trq real 'kg.m',
      req_trq real 'kg.m';
where
      @rel_cut_friction * @max_cut_area = ^max_cut_force;
      @max_cut_force * @max_cut_spd / 6120 = ^cut_power;
      @max_cut_force * @max_work_dia/4000.0 = ^max_cut_trq;
      @drill_cut_spd / @max_drill_dia*1000.0/3.1416 = Drill_rot,
      0.6*0.75*0.01*(@max_drill_dia**2) *0.01 * @drill_cut_spd*2.2= Drill_power,
      974.0*Drill_power/Drill_rot = ^max_drill_trq;
      bigger(@max_cut_trq,@max_drill_trq) = ^req_trq;
end.

class material
parameter
      name string [],
      elasticity real 'kg/mm2',
      shearing real 'kg/mm2',
      poisson real [],
      spc_grabity real [];
where
data(name, ^elasticity, ^shearing, ^poisson, ^spc_grabity);
local
data(steel,21000,8200,0.3,7.9);
data(iron,21500,3300,0.28,7.9);
data(alminum,7200,2760,0.34,2.7);
data(cupper,12500,4700,0.34,8.9);
data(brass,6300,2400,0.34,8.0);
end.


class shaft
part
material:= #material;
parameter
     ext_dia integer 'mm',
     int_dia integer 'mm',
     length integer 'mm',
     bearing_dist integer 'mm',
     max_rot integer 'rpm',
     base_rot integer 'rpm',
     req_trq real 'kg.m',
     max_trq real 'kg.m',
     avg_trq real 'kg.m',
     req_power real 'KW',
     shearing real 'kg/mm2',
     right_dia   integer 'mm',
     left_dia   integer 'mm',
     max_rot_trq real 'rpm.kg.m',
     max_base_ratio real [];
end.

class main_shaft
super shaft;
part
     specification:= #work_cutting;
parameter
     max_cut_trq real 'kg.m';
where
   ext_dia == right_dia;
   shearing == material!shearing;
   req_power == specification!cut_power;
   max_cut_trq == specification!max_cut_trq;
   req_trq == specification!req_trq;
   left_dia+5 = right_dia;
   @req_trq * @max_rot = ^max_rot_trq;
%tortion equation -- to calculate the diameter
     @int_dia**4.0+2.334E10*max_cut_trq/ @shearing = X,
     sqrt(sqrt(X))/5.0 = Y,
     (floating_point_to_integer(Y)+1)*5 = ^ext_dia;
%Schenk'sequation -- to calculate the bearing distance
     @int_dia**2 = Dint2,
     ext_dia**2 = Dr2,
     0.53*(Dr2+Dint2)*(Dr2-Dint2)  = L1,
     L1**0.33333/10.0 = L2,
     (floating_point_to_integer(L2)+1)*10 = ^bearing_dist;
end.

class rot_main_shaft
part
main_shaft:=  #main_shaft,
left_bearing:= #bearing,
right_bearing:= #bearing;
where
   main_shaft!ext_dia == left_bearing!int_dia;
   main_shaft!ext_dia == right_bearing!int_dia;
   main_shaft!max_rot == left_bearing!max_rot;
   main_shaft!max_rot == right_bearing!max_rot;
end.

class counter_shaft
super shaft;
where
   ext_dia == right_dia;
   ext_dia == left_dia;
   shearing == material!shearing;
%Torsion Equation
     1.945E6*max_trq*1000.0/ @shearing = X,
     sqrt(sqrt(X))/5.0 = Y,
     (floating_point_to_integer(Y)+2)*5 = ^right_dia;
end.
```

```
class rot_counter_shaft
part
counter_shaft:= #counter_shaft,
left_bearing:= #bearing,
right_bearing:= #bearing;
where
   counter_shaft!ext_dia == left_bearing!int_dia;
   counter_shaft!ext_dia == right_bearing!int_dia;
   counter_shaft!max_rot == left_bearing!max_rot;
   counter_shaft!max_rot == right_bearing!max_rot;
end.

class key
parameter
      shaft_dia integer 'mm',
      width       integer 'mm',
      height      integer 'mm',
      min_length   integer 'mm',
      max_length integer 'mm';
where                %to select a key
      data (X,Y,^width,^height,^min_length,^max_length) ,
      shaft_dia >= X,shaft_dia < Y;
local
data (13,20,5,5,10,56) ;
data (20,30,7,7,14,90) ;
data (30,40,10,8,18,112) ;
data (40,50,12,8,22.4,140) ;
data (50,60,15,10,28,160) ;
data (60,70,18,12,35.5,200) ;
data (70,80,20,13,45,224) ;
data (80,95,24,16,56,250) ;
data (95,110,28,18,63,315) ;
data (110,125,32,20,80,355) ;
data (125,140,35,22,100,400) ;
data (140,160,38,24,112,400) ;
end.

class spur_gear
part
key:= #key;
parameter
module integer,
pitch_dia integer 'mm',
int_dia integer 'mm',
min_teeth_num integer,
max_teeth_num integer,
teeth_num integer,
width integer 'mm',
%teeth_max_stress internal 'kg/mm2',
key_height integer 'mm',
max_teeth_spd integer [unit ('m/min') ,default (2000) ] ,
max_rot integer 'rpm',
base_rot integer 'rpm',
max_trq real 'kg.m';
where
int_dia == key!shaft_dia;
key_height == key!height;

 (@int_dia + key_height) / @module + 7.5 = ^min_teeth_num;
```

```
1000.0 * @max_teeth_spd / 3.1416/ @max_rot/ @module = ^max_teeth_num;

@teeth_num >= @min_teeth_num, @teeth_num =< @max_teeth_num,
@teeth_num * @module = ^pitch_dia;
end.

class max_spd_spur_gear
part
key:= #key;
parameter
module integer,
ext_dia integer 'mm',
int_dia integer 'mm',
min_teeth_num integer,
max_teeth_num integer,
teeth_num integer,
width integer 'mm',
%teeth_max_stress internal 'kg/mm2',
key_height integer 'mm',
max_teeth_spd integer [unit('m/min'),default(2000)],
max_rot integer 'rpm',
base_rot integer 'rpm',
max_trq real 'kg.m';
where
int_dia == key!shaft_dia;
key_height == key!height;

data(^module),
(@int_dia + key_height)/ ^module + 7.5 = ^min_teeth_num,
1000.0 * @max_teeth_spd / 3.1416/ @max_rot/ ^module = ^max_teeth_num,
^max_teeth_num >=  ^min_teeth_num;

@teeth_num >= @min_teeth_num,
@teeth_num =< @max_teeth_num,
@teeth_num * @module = ^ext_dia;

local
      data(12);
      data(8);
      data(5);
      data(4);
      data(3);
      data(2);
end.

class gear_pair
part
   drv_shaft:= #shaft,
   flw_shaft:= #shaft,
   drv_gear:= #spur_gear,
   flw_gear:= #spur_gear;
parameter
   gear_ratio real,
   shaft_dist integer 'mm',
   drv_max_rot integer 'rpm',
   drv_base_rot integer 'rpm',
   drv_max_trq real 'kg.m',
   drv_gear_teeth_num integer,
   flw_gear_teeth_num integer,
   module integer [];
```

```
where
   drv_gear! module == flw_gear! module;
   module ==drv_gear!module;
   flw_gear!width == drv_gea!width;
   drv_max_rot == drv_shaft!max_rot;
   drv_base_rot == drv_shaft!base_rot;
   drv_max_rot == drv_gear!max_rot;
   drv_base_rot == drv_gear!base_rot;
   drv_max_trq == drv_gear!max_trq;
   drv_max_trq == drv_shaft!max_trq;
   flw_gear_teeth_num == flw_gear!teeth_num;
   drv_gear_teeth_num == drv_gear!teeth_num;
   flw_shaft!ext_dia == flw_gear!int_dia;
   drv_shaft!ext_dia == drv_gear!int_dia;

   @shaft_dist*2 / (@module*(@gear_ratio +1)) = ^drv_gear_teeth_num,
   @shaft_dist*2 / @module - ^drv_gear_teeth_num = ^flw_gear_teeth_num;
end.

class max_spd_gear_pair
part
   drv_shaft:= #shaft,
   flw_shaft:= #shaft,
   drv_gear:= #spur_gear,
   flw_gear:= #max_spd_spur_gear;
parameter
   gear_ratio real,
   shaft_dist integer 'mm',
   drv_max_rot integer 'rpm',
   drv_base_rot integer 'rpm',
   flw_max_rot integer 'rpm',
   drv_max_trq real 'kg.m',
   min_drv_teeth_num integer,
   max_drv_teeth_num integer,
   drv_gear_teeth_num integer,
   min_flw_teeth_num integer,
   max_flw_teeth_num integer,
   flw_gear_teeth_num integer,
   module integer [];
where
   drv_gear! module == flw_gear! moduie;
   module ==drv_gear! module;
   flw_gear!width == drv_gea!width;
   drv_max_rot == drv_shaft!max_rot;
   drv_base_rot == drv_shaft!base_rot;
   flw_max_rot == flw_shaft!max_rot;
   flw_max_rot == flw_gear!max_rot;
   drv_max_rot == drv_gear!max_rot;
   drv_base_rot == drv_gear!base_rot;
   drv_max_trq == drv_gear!max_trq;
   drv_max_trq == drv_shaft!max_trq;
   min_flw_teeth_num == flw_gear!min_teeth_num;
   max_flw_teeth_num == flw_gear!max_teeth_num;
   flw_gear_teeth_num == flw_gear!teeth_num;
   min_drv_teeth_num == drv_gear!min_teeth_num;
   max_drv_teeth_num == drv_gear!max_teeth_num;
   drv_gear_teeth_num == drv_gear!teeth_num;
   flw_shaft!ext_dia == flw_gear!int_dia;
   drv_shaft!ext_dia == drv_gear!int_dia;
   @drv_max_rot / @flw_max_rot = ^gear_ratio;

   ((flw_gear_teeth_num)  + drv_gear_teeth_num) * @module = ^shaft_dist*2,
   (flw_gear_teeth_num)  / drv_gear_teeth_num = @gear_ratio,
   ^(flw_gear!base_rot)  *  (flw_gear_teeth_num)  = @drv_base_rot * drv_gear_teeth_num,
   @drv_max_trq * @gear_ratio   * 0.75 = ^(flw_gear!max_trq);
end.

class max_trq_gear_pair
part
   drv_shaft:= #shaft,
   flw_shaft:= #shaft,
   drv_gear:= #spur_gear,
   flw_gear:= #spur_gear;
parameter
   min_gear_ratio real,
   gear_ratio real ,
   shaft_dist integer 'mm',
   drv_max_rot integer 'rpm',
   drv_max_trq real 'kg.m',
   req_flw_max_trq real 'kg.m',
   flw_max_trq real 'kg.m',
   flw_gear_max_rot integer 'rpm',
   min_drv_teeth_num integer,
   max_drv_teeth_num integer,
   drv_gear_teeth_num integer,
   min_flw_teeth_num integer,
   max_flw_teeth_num integer,
   flw_gear_teeth_num integer,
   module integer [];
where
   drv_gear!module == flw_gear!module;
   module ==drv_gear!module;
   flw_shaft!width == drv_gear!width;
   drv_max_rot == drv_shaft!max_rot;
   drv_max_rot == drv_gear!max_rot;
   drv_shaft!base_rot == drv_gear!base_rot;
   drv_max_trq == drv_shaft!max_trq;
   drv_max_trq == drv_gear!max_trq;
   req_flw_max_trq == flw_shaft!req_trq;
   flw_max_trq == flw_shaft!max_trq;
   flw_max_trq == flw_gear!max_trq;
   flw_gear_max_rot == flw_gear!max_rot;
   min_flw_teeth_num == flw_gear!min_teeth_num;
   max_flw_teeth_num == flw_gear!max_teeth_num;
   flw_gear_teeth_num == flw_gear!teeth_num;
   min_drv_teeth_num == drv_gear!min_teeth_num;
   max_drv_teeth_num == drv_gear!max_teeth_num;
   drv_gear_teeth_num == drv_gear!teeth_num;
   flw_shaft!ext_dia == flw_gear!int_dia;
   drv_shaft!ext_dia == drv_gear!int_dia;

   @req_flw_max_trq / @drv_max_trq = ^min_gear_ratio * 0.75;

    @shaft_dist*2 / @module =X,
    X * @min_gear_ratio   / (1.0 + @min_gear_ratio) = Y,
    floating_point_to_integer(Y)  + 1 = ^flw_gear_teeth_num,
    X - ^flw_gear_teeth_num = ^drv_gear_teeth_num;

   ^flw_gear_max_rot *  @req_flw_max_trq = @drv_max_rot * @drv_max_trq * 0.75;
```

```
end.

class series_of_gears
part
   drv_shaft:= #shaft,
   flw_shaft:= #shaft,
   max_spd_gear_pair:= #max_spd_gear_pair,
   second_gear_pair from #gear_pair,
   third_gear_pair from #gear_pair,
   min_spd_gear_pair:= #max_trq_gear_pair;
parameter
   shaft_dist integer 'mm',
   module integer,
   max_flw_base_rot integer 'rpm',
   min_flw_max_rot integer 'rpm',
   max_drv_teeth_num integer [],
   max_flw_teeth_num integer [],
   min_drv_teeth_num integer [],
   min_flw_teeth_num integer [],
   drv_req_power real 'KW',
   flw_req_power real 'KW',
   reduction_range real [],
   num_of_pairs integer [];
where
   drv_req_power == drv_shaft!req_power;
   flw_req_power == flw_shaft!req_power;
   drv_shaft == max_spd_gear_pair!drv_shaft;
   drv_shaft == min_spd_gear_pair!drv_shaft;
   flw_shaft == max_spd_gear_pair!flw_shaft;
   flw_shaft == min_spd_gear_pair!flw_shaft;
   shaft_dist == max_spd_gear_pair!shaft_dist;
   shaft_dist == min_spd_gear_pair!shaft_dist;
   module == max_spd_gear_pair!module;
   module == min_spd_gear_pair!module;
   max_flw_base_rot == max_spd_gear_pair!flw_gear_base_rot;
   min_flw_max_rot == min_spd_gear_pair!flw_gear_max_rot;
   max_drv_teeth_num == max_spd_gear_pair!drv_gear_teeth_num;
   max_flw_teeth_num == max_spd_gear_pair!flw_gear_teeth_num;
   min_drv_teeth_num == min_spd_gear_pair!drv_gear_teeth_num;
   min_flw_teeth_num == min_spd_gear_pair!flw_gear_teeth_num;

   @max_flw_base_rot  =< @min_flw_max_rot;
   flw_req_power/0.75 = drv_req_power;
   @(flw_shaft!max_rot_trq) / @ (drv_shaft!max_rot_trq) = ^reduction_range,
   ^reduction_range / @(drv_shaft!max_base_ratio) = X,
   floor(X) + floor((X - floor(X))*1.25) +2 = Num_of_Pairs,
   floating_point_to_integer(Num_of_Pairs) = ^num_of_pairs;

   @num_of_pairs=3 ==>
     sys:assemble(instance,#gear_pair, second_gear_pair);

   @num_of_pairs=4 ==>
     sys:assemble(instance,#gear_pair, second_gear_pair),
     sys:assemble(instance,#gear_pair, third_gear_pair);

   @num_of_pairs=3 ==>
     sqrt(@(min_spd_gear_pair!min_gear_ratio)  *
            @(max_spd_gear_pair!gear_ratio) ) = ^(second_gear_pair!gear_ratio);

   @num_of_pairs=4 ==>
```

```
    (@(min_spd_gear_pair!min_gear_ratio) /
     @(max_spd_gear_pair!gear_ratio))**0.333 = X,
     @(max_spd_gear_pair!gear_ratio)  * X = ^(second_gear_pair!gear_ratio),
     @(max_spd_gear_pair!gear_ratio) *X*X = ^(third_gear_pair!gear_ratio);

end.

class motor
part
flw_shaft:= #shaft;
parameter
req_power real 'KW',
max_power real 'KW',
power   real 'KW',
max_current integer 'A',
current   integer 'A',
volt      integer 'V',
base_rot       integer 'rpm',
max_rot        integer 'rpm',
max_trq_30min real       'kg.m',
shaft_dia     integer 'mm';
where
req_power == flw_shaft!req_power;
shaft_dia ==flw_shaft!ext_dia;
max_power == flw_shaft!max_power;
base_rot == flw_shaft!base_rot;
max_rot == flw_shaft!max_rot;
max_trq_30min == flw_shaft!max_trq;
data(^max_power,^power,^max_current,^current,^volt,
                    ^base_rot,^max_rot,^max_trq_30min,^shaft_dia),
req_power =< ^max_power;
%req_power >= ^power;
@max_rot * @max_trq_30min = ^(flw_shaft!max_rot_trq);
@max_rot / @base_rot = ^(flw_shaft!max_base_ratio);
local
data(5.5,3.7,34,25,180,1500,6000,3.57,28);
data(7.5,5.5,43,35,180,1500,6000,4.86,32);
data(11,7.5,74,55,180,1500,6000,7.14,32);
data(15,11,97,75,180,1500,6000,9.7,42);
data(18.5,15,100,85,180,1500,6000,12.0,48);
data(22,18.5,121,105,180,1500,6000,14.2,55);
end.

class bearing
parameter
     int_dia              integer 'mm' ,
     max_ext_dia              integer 'mm',
     mean_d_load           integer 'kgf',
     max_rot           integer 'rpm',
     avg_rot          integer 'rpm',
     life_time              integer 'hour';

database
     name                      string,
     int_dia                 integer 'mm',
     ext_dia                 integer 'mm',
     width                    integer 'mm',
     round                    real      'mm',
     d_load                  integer 'kg',
     s_load                  integer 'kg',
     g_max_rot            integer 'rpm',
     o_max_rot            integer 'rpm',
     mass                  real    'kg';
     int_dia = d!int_dia,
     max_ext_dia >= d!ext_dia,
     max_rot =< d!g_max_rot,
     life_time =< (d!d_load/mean_d_load)**3.1666/avg_rot;

local
 data(['6800',10,19,5,0.5,175,75,38000,45000,0.0056]);
 data(['6900',10,22,6,0.5,275,117,34000,40000,0.0096]);
 data(['6000',10,26,8,0.5,465,196,30000,36000,0.0190]);
 data(['6200',10,30,9,1.0,520,229,24000,30000,0.0320]);
 data(['6300',10,35,11,1.0,825,365,22000,26000,0.0530]);
 data(['6801',12,21,5,0.5,195,91,33000,40000,0.0064]);
 data(['6901',12,24,6,0.5,295,132,30000,36000,0.0110]);
 data(['16001',12,28,7,0.5,520,228,28000,32000,0.0210]);
 data(['6001',12,28,8,0.5,520,228,28000,32000,0.0220]);
 data(['6201',12,32,10,1.0,695,305,22000,28000,0.0370]);
 data(['6301',12,37,12,1.5,990,450,20000,24000,0.0600]);
 data(['6802',15,24,5,0.5,212,107,28000,33000,0.0075]);
 data(['6902',15,28,7,0.5,440,205,26000,30000,0.0160]);
 data(['16002',15,32,8,0.5,570,263,23000,28000,0.0250]);
 data(['6002',15,32,9,0.5,570,263,23000,28000,0.0300]);
 data(['6202',15,35,11,1.0,780,360,20000,24000,0.0450]);
 data(['6302',15,37,12,1.5,990,450,20000,16000,0.0820]);
 data(['6803',17,26,5,0.5,268,135,26000,30000,0.0083]);
 data(['6903',17,30,7,0.5,470,228,24000,28000,0.0180]);
 data(['16003',17,35,8,0.5,610,296,21000,24000,0.0320]);
 data(['6003',17,35,10,0.5,610,296,21000,24000,0.0390]);
 data(['6203',17,40,12,1.0,975,460,17000,20000,0.0650]);
 data(['6303',17,47,14,1.5,1390,660,16000,19000,0.115]);
 data(['6804',20,32,7,0.5,410,214,21000,25000,0.0190]);
 data(['6904',20,37,9,0.5,650,330,19000,23000,0.0360]);
 data(['16004',20,42,8,0.5,810,405,18000,20000,0.0500]);
 data(['6004',20,42,12,1.0,955,465,18000,20000,0.0690]);
 data(['6204',20,47,14,1.5,1300,635,15000,18000,0.105]);
 data(['6304',20,52,15,2.0,1620,785,14000,17000,0.145]);
 data(['60/22',22,44,12,1.0,960,470,17000,20000,0.0730]);
 data(['62/22',22,50,14,1.5,1320,645,14000,17000,0.118]);
 data(['63/22',22,56,16,2.0,1870,920,13000,15000,0.176]);
 data(['6805',25,37,7,0.5,440,250,18000,21000,0.022]);
 data(['6905',25,42,9,0.5,715,395,16000,19000,0.043]);
 data(['16005',25,47,8,0.5,905,490,15000,18000,0.060]);
 data(['6005',25,47,12,1.0,1030,525,15000,18000,0.080]);
 data(['6205',25,52,15,1.5,1430,730,13000,15000,0.128]);
 data(['6305',25,62,17,2.0,2100,1080,11000,14000,0.232]);
 data(['60/28',28,52,12,1.0,1270,665,14000,16000,0.096]);
 data(['62/28',28,58,16,1.5,1700,885,12000,14000,0.172]);
 data(['63/28',28,68,18,2.0,2730,1400,10000,13000,0.288]);
 data(['6806',30,42,7,0.5,460,285,15000,18000,0.026]);
 data(['6906',30,47,9,0.5,740,430,14000,17000,0.049]);
 data(['16006',30,55,9,0.5,1150,645,13000,15000,0.088]);
 data(['6006',30,55,13,1.0,1350,845,13000,15000,0.113]);
 data(['6206',30,62,16,1.0,1980,1150,11000,13000,0.196]);
 data(['6306',30,72,19,1.1,2720,1530,9500,12000,0.339]);
   data(['6807',35,47,7,0.3,485,400,13000,16000,0.027]);
   data(['6907',35,55,10,0.6,1080,740,12000,15000,0.074]);
   data(['16007',35,62,9,0.3,1190,835,11000,13000,0.107]);
   data(['6007',35,62,14,1.0,1630,1050,11000,13000,0.147]);
```

```
data（［'6207',35,72,17,1.1,2620,1560,9500,11000,0.279］）；
data（［'6307',35,80,21,1.5,3400,1960,8500,10000,0.449］）；
data（［'6808',40,52,7,0.3,500,445,12000,14000,0.030］）；
data（［'6908',40,65,12,0.6,1390,1020,11000,13000,0.11］）；
data（［'16008',40,68,9,0.3,1290,985,10000,12000,0.13］）；
data（［'6008',40,68,15,1.0,1710,1180,10000,12000,0.186］）；
data（［'6208',40,80,18,1.1,2970,1820,8500,10000,0.359］）；
data（［'6308',40,90,23,1.5,4150,2450,7500,9000,0.62］）；
data（［'6809',45,58,7,0.3,550,535,11000,13000,0.038］）；
data（［'6909',45,68,12,0.6,1440,1110,9500,12000,0.124］）；
data（［'16009',45,75,10,0.6,1520,1160,9000,11000,0.167］）；
data（［'6009',45,75,16,1.0,2140,1550,9000,11000,0.236］）；
data（［'6209',45,85,19,1.1,3200,2080,7500,9000,0.413］）；
data（［'6309',45,100,25,1.5,5400,3250,6700,8000,0.811］）；
data（［'6810',50,65,7,0.3,655,635,9500,11000,0.049］）；
data（［'6910',50,72,12,0.6,1480,1200,9000,11000,0.133］）；
data（［'16010',50,80,10,0.6,1570,1260,8500,10000,0.175］）；
data（［'6010',50,80,16,1.0,2200,1700,8500,10000,0.256］）；
data（［'6210',50,90,20,1.1,3600,2370,7100,8500,0.451］）；
data（［'6310',50,110,27,2.0,6300,3900,6000,7500,1.05］）；
data（［'6811',55,72,7,0.3,900,865,8500,10000,0.080］）；
data（［'6911',55,80,13,1.0,1630,1350,8000,9500,0.184］）；
data（［'16011',55,90,11,0.6,1980,1660,7500,9000,0.257］）；
data（［'6011',55,90,18,1.1,2880,2170,7500,9000,0.373］）；
data（［'6211',55,100,21,1.5,4450,2980,6300,7500,0.599］）；
data（［'6311',55,120,29,2.0,7300,4550,5600,6700,1.35］）；
data（［'6812',60,78,10,0.3,1170,1120,8000,9500,0.102］）；
data（［'6912',60,85,13,1.0,1980,1660,7500,9000,0.187］）；
data（［'16012',60,95,11,0.6,2040,1780,7100,8500,0.281］）；
data（［'6012',60,95,18,1.1,3000,2370,7100,8500,0.403］）；
data（［'6212',60,110,22,1.5,5350,3700,5600,7100,0.762］）；
data（［'6312',60,130,31,2.1,8350,5300,5300,6300,1.69］）；
data（［'6813',65,85,10,0.6,1220,1230,7500,8500,0.125］）；
data（［'6913',65,90,13,1.0,1770,1640,7100,8500,0.212］）；
data（［'16013',65,100,11,0.6,2090,1910,6700,8000,0.30］）；
data（［'6013',65,100,18,1.1,3100,2570,6700,8000,0.43］）；
data（［'6213',65,120,23,1.5,5850,4100,5300,6300,0.979］）；
data（［'6313',65,140,33,2.1,9450,6100,4800,6000,2.08］）；
data（［'6814',70,90,10,0.6,1230,1300,6700,8000,0.131］）；
data（［'6914',70,100,16,1.0,2420,2160,6300,7500,0.343］）；
data（［'16014',70,110,13,0.6,2730,2410,6000,7100,0.437］）；
data（［'6014',70,110,20,1.1,3900,3150,6000,7100,0.598］）；
data（［'6214',70,125,24,1.5,6350,4500,5000,6300,1.07］）；
data（［'6314',70,150,35,2.1,10600,6950,4500,5300,2.53］）；
data（［'6815',75,95,10,0.6,1280,1410,6300,7500,0.145］）；
data（［'6915',75,105,16,1.0,2480,2300,6000,7100,0.357］）；
data（［'16015',75,115,13,0.6,2820,2580,5600,6700,0.463］）；
data（［'6015',75,115,20,1.1,4050,3400,5600,6700,0.638］）；
data（［'6215',75,130,25,1.5,6750,5050,4800,5600,1.17］）；
data（［'6315',75,160,37,2.1,11600,7850,4300,5000,3.03］）；
data（［'6816',80,100,10,0.6,1290,1470,6000,7100,0.147］）；
data（［'6916',80,115,16,1.0,2540,2450,5600,6700,0.383］）；
data（［'16016',80,125,14,0.6,3250,3000,5300,6300,0.621］）；
data（［'6016',80,125,22,1.1,4850,4050,5300,6300,0.854］）；
data（［'6216',80,140,26,2.0,7400,5400,4500,5300,1.38］）；
data（［'6316',80,170,39,2.1,12500,8850,4000,4800,3.6700］）
data（［'6817',85,110,13,1.0,1910,2040,5600,6700,0.255］）；
data（［'6917',85,120,18,1.1,3250,3000,5300,6300,0.539］）；
data（［'16017',85,130,14,0.6,3350,3200,5000,6000,0.652］）；
data（［'6818',90,115,13,1.0,1940,2140,5300,6300,0.268］）；
```

```
  data(['6918',90,125,18,1.1,3350,3200,5000,6000,0.566]);
  data(['16018',90,140,16,1.0,4250,4000,4800,5600,0.866]);
  data(['6819',95,120,13,1.0,1970,2240,5000,6000,0.288]);
  data(['6919',95,130,18,1.1,3450,3400,4800,5600,0.581]);
  data(['16019',95,145,16,1.0,4350,4250,4500,5300,0.904]);
  data(['6820',100,125,13,1.0,2000,2340,4800,5600,0.302]);
  data(['6920',100,140,20,1.1,4350,4250,4500,5300,0.807]);
  data(['16020',100,150,16,1.0,4300,4300,4300,5300,0.936]);
  data(['6821',105,130,13,1.0,2020,2440,4800,5600,0.315]);
  data(['6921',105,145,20,1.1,4300,4300,4300,5300,0.835]);
  data(['16021',105,160,18,1.0,5300,5150,4000,4800,1.23]);
  data(['6822',110,140,16,1.0,2860,3350,4300,5300,0.487]);
  data(['6922',110,150,20,1.1,4450,4550,4300,5000,0.87]);
  data(['16022',110,170,19,1.0,5850,5800,3800,4500,1.5]);
End.

class angular_bearing
parameter
    int_dia            integer 'mm' ,
    max_ext_dia            integer 'mm',
    mean_d_load          integer 'kgf',
    max_rot          integer 'rpm',
    avg_rot          integer 'rpm',
    life_time              integer 'hour';
database
    name                    string,
    int_dia                 integer 'mm',
    ext_dia                 integer 'mm',
    width                    integer 'mm',
    round                    real      'mm',
    d_load                  integer 'kg',
    s_load                  integer 'kg',
    g_max_rot               integer 'rpm',
    o_max_rot               integer 'rpm',
    mass                     real      'kg';
    int_dia = d!int_dia,
    max_ext_dia >= d!ext_dia,
    max_rot =< d!g_max_rot,
    life_time =< (d!d_load/mean_d_load)**3.1666/avg_rot;
local
 data(['7900A5 DB',10,22,6,0.3,475,296,32000,43000,0.018]);
 data(['7000A DB',10,26,8,0.3,890,530,24000,34000,0.038]);
 data(['7200A DB',10,30,9,0.6,900,555,22000,30000,0.062]);
 data(['7300A DB',10,35,11,0.6,1540,880,16000,22000,0.104]);
 data(['7901A5 DB',12,24,6,0.3,530,360,30000,43000,0.022]);
 data(['7001A DB',12,28,8,0.3,955,610,22000,30000,0.042]);
 data(['7201A DB',12,32,10,0.6,1330,820,20000,28000,0.072]);
 data(['7301A DB',12,37,12,1,1570,915,15000,20000,0.108]);
 data(['7902A5 DB',15,28,7,0.3,755,515,26000,34000,0.03]);
 data(['7002A DB',15,32,9,0.3,1010,700,19000,26000,0.060]);
 data(['7202A DB',15,35,11,0.6,1430,950,18000,24000,0.090]);
 data(['7302A DB',15,42,13,1,2220,1440,13000,17000,0.166]);
 data(['7903A5 DB',17,30,7,0.3,790,570,24000,32000,0.034]);
 data(['7003A DB',17,35,10,0.3,1060,780,17000,24000,0.08]);
 data(['7203A DB',17,40,12,0.6,1790,1220,16000,22000,0.132]);
 data(['7303A DB',17,47,14,1,2640,1760,11000,15000,0.228]);
 data(['7094A5 DB',20,37,9,0.3,1090,825,19000,26000,0.072]);
 data(['7004A DB',20,42,12,0.6,1800,1340,15000,20000,0.134]);
 data(['7204A DB',20,47,14,1,2400,1690,13000,19000,0.21]);
 data(['7304A DB',20,52,15,1.1,3100,2130,10000,13000,0.284]);
 data(['7905A5 DB',25,42,9,0.3,1230,1050,16000,22000,0.086]);
 data(['7005A DB',25,47,12,0.6,1870,1510,13000,17000,0.154]);
 data(['7205A DB',25,52,15,1,2690,2090,12000,16000,0.256]);
 data(['7305A DB',25,62,17,1.1,4400,3250,8500,11000,0.46]);
 data(['7906A5 DB',30,47,9,0.3,1300,1210,14000,19000,0.098]);
 data(['7006A',30,55,13,1,2410,2060,11000,15000,0.228]);
 data(['7206A DB',30,62,16,1,3750,3000,10000,13000,0.39]);
 data(['7306A DB',30,72,19,1.1,5600,4250,7100,9500,0.682]);
 data(['7907A5 DB',35,55,10,0.6,1890,1770,12000,17000,0.146]);
 data(['7007A DB',35,62,14,1,3050,2740,9500,13000,0.302]);
 data(['7207A DB',35,72,17,1.1,4900,4100,8500,12000,0.562]);
 data(['7307A DB',35,80,21,1.5,6600,5350,6300,8500,0.912]);
 data(['7908A5 DB',40,62,12,0.6,2370,2270,11000,15000,0.216]);
 data(['7008A DB',40,68,15,1,3250,3150,8500,11000,0.376]);
 data(['7208A DB',40,80,18,1.1,5850,5150,7500,10000,0.736]);
 data(['7308A DB',40,90,23,1.5,8100,6700,5600,7500,1.248]);
 data(['7909A5 DB',45,68,12,0.6,2510,2590,9500,13000,0.258]);
 data(['7009A DB',45,75,16,1,3850,3800,7500,10000,0.5]);
 data(['7209A DB',45,85,19,1.1,6550,5850,7100,9500,0.808]);
 data(['7309A DB',45,100,25,1.5,10500,8900,5000,6700,1.676]);
 data(['7910A5 DB',50,72,12,0.6,2640,2900,9000,12000,0.26]);
 data(['7010A DB',50,80,16,1,4050,4300,7100,9500,0.54]);
 data(['7210A DB',50,90,20,1.1,6850,6400,6300,9000,0.916]);
 data(['7310A DB',50,110,27,2,12300,10600,4500,6000,2.16]);
 data(['7911A5 DB',55,80,13,1,2990,3400,8000,11000,0.36]);
 data(['7011A DB',55,90,18,1.1,5350,5650,6300,8500,0.766]);
 data(['7211A DB',55,100,21,1.5,8450,8050,6000,8000,1.204]);
 data(['7311A DB',55,120,29,2,14200,12500,4000,5600,2.8]);
 data(['7912A5 DB',60,85,13,1,3050,3600,7500,10000,0.386]);
 data(['7012A DB',60,95,18,1.1,5450,6000,6000,8000,0.816]);
 data(['7212A DB',60,110,22,1.5,10200,9950,5300,7100,1.572]);
 data(['7312A DB',60,130,31,2.1,16200,14500,3800,5000,3.44]);
 data(['7913A DB',65,90,13,1,3150,3950,7100,9500,0.412]);
 data(['7013A DB',65,100,18,1.1,5750,6700,5600,7500,0.91]);
 data(['7213A DB',65,120,23,1.5,11600,11800,4800,6700,2.04]);
 data(['7313A DB',65,140,33,2.1,18400,16700,3600,4800,4.18]);
 data(['7914A DB',70,100,16,1,4400,5350,6300,9000,0.672]);
 data(['7014A DB',70,110,20,1.1,7300,8450,5000,6700,1.25]);
 data(['7214A DB',70,125,24,1.5,12600,13000,4500,6300,2.2]);
 data(['7314A DB',70,150,35,2.1,20700,19100,3200,4300,5.14]);
 data(['7915A DB',75,105,16,1,4450,5650,6000,8500,0.71]);
 data(['7015A DB',75,115,20,1.1,7450,8900,4800,6700,1.304]);
 data(['7215A DB',75,130,25,1.5,12600,13100,4300,6000,2.36]);
 data(['7315A DB',75,160,37,2.1,22500,21600,3000,4000,6.2]);
 data(['7916A5 DB',80,110,16,1,4550,5900,5600,8000,0.75]);
 data(['7016A DB',80,125,22,1.1,9150,10800,4300,6000,0.176]);
 data(['7216A DB',80,140,26,2,14700,15600,4000,5600,2.88]);
 data(['7316A DB',80,170,39,2.1,24400,24200,2800,3800,7.36]);
 data(['7917A5 DB',85,120,18,1.1,6100,7850,5300,7500,0.054]);
 data(['7017A DB',85,130,22,1.1,9350,11400,4300,5600,1.808]);
 data(['7217A DB',85,150,28,2,17100,18200,3800,5300,3.64]);
 data(['7317A DB',85,180,41,3,26300,27000,2600,3600,8.58]);
 data(['7918A5 DB',90,125,18,1.1,6500,8900,5000,7100,1.12]);
 data(['7018A DB',90,140,24,1.5,11200,13500,3800,5300,2.34]);
 data(['7218A DB',90,160,30,2,19050,21000,3600,5000,4.46]);
 data(['7318A DB',90,190,43,3,28300,30000,2600,3400,10.06]);
 data(['7919A5 DB',95,130,18,1.1,6600,9250,4800,6700,1.176]);
 data(['7019A DB',95,145,24,1.5,11100,13600,3800,5000,2.82]);
 data(['7219A DB',95,170,32,2.1,21200,22600,3400,4500,5.28]);
End.
```

```
class conical_bearing
parameter
     int_dia              integer 'mm' ,
     max_ext_dia            integer 'mm',
     mean_d_load          integer 'kgf',
     max_rot          integer 'rpm',
     avg_rot          integer 'rpm',
     life_time              integer 'hour';
database
     name                    string,
     int_dia                integer 'mm',
     ext_dia                integer 'mm',
     width                   integer 'mm',
     round                   real      'mm',
     d_load                 integer 'kg',
     s_load                 integer 'kg',
     g_max_rot              integer 'rpm',
     o_max_rot              integer 'rpm',
     mass                    real      'kg';
     int_dia = d!int_dia,
     max_ext_dia >= d!ext_dia,
     max_rot =< d!g_max_rot,
     life_time =< (d!d_load/mean_d_load) **3.1666/avg_rot;
local
  data(['30202',15,35,11,0.6,1510,1350,11000,15000,0.053]);
  data(['HR30302J',15,42,13,1,2400,2160,9500,13000,0.098]);
  data(['HR30203J',17,40,12,1,2050,2030,9500,13000,0.079]);
  data(['32203',17,40,16,1,2310,2380,9500,13000,0.104]);
  data(['HR30303J',17,47,14,1,2980,2720,8500,12000,0.134]);
  data(['HR32004XJ',20,42,15,0.6,2510,2800,9000,12000,0.097])
  data(['HR30204C',20,47,14,1,2430,2450,8000,11000,0.122]);
  data(['32204X',20,47,18,1,3000,3150,8500,11000,0.16]);
  data(['HR30304J',20,52,15,1.5,3550,3400,7500,11000,0.171]);
  data(['HR32304C',20,52,21,1.5,4300,4500,7500,10000,0.238]);
  data(['HR320/22XJ',22,44,15,0.6,2610,3000,8500,11000,0.103])
  data(['HR302/22',22,50,14,1,3050,3250,7500,10000,0.139]);
  data(['HR322/22',22,50,18,1,3750,4100,7500,11000,0.18]);
  data(['HR303/22C',22,56,16,1.5,3500,3500,6700,9500,0.206]);
  data(['HR323/22C',22,56,21,1.5,4500,4950,6700,9500,0.266]);
  data(['HR32005XJ',25,47,15,0.6,2800,3400,8000,11000,0.116])
  data(['HR33005J',25,47,17,0.6,3150,3900,8000,11000,0.131]);
  data(['HR30205C',25,52,15,1,2860,3200,6700,9500,0.154]);
  data(['HR32205',25,52,18,1,3900,4450,7500,10000,0.186]);
  data(['30305D',25,62,17,1.5,3200,3150,6000,8000,0.254]);
  data(['HR32305C',25,62,24,1.5,5650,6550,6300,8500,0.365]);
  data(['HR320/28XJ',28,52,16,1,3300,3950,7100,9500,0.146]);
  data(['HR302/28C',28,58,16,1,3450,3900,6300,8500,0.199]);
  data(['HR322/28',58,19,1,4850,5500,6300,9000,0.242]);
  data(['HR303/28C',28,68,18,1.5,5000,5150,5600,7500,0.335]);
  data(['HR32009XJ',30,55,17,1,3700,4550,6700,9000,0.172]);
  data(['HR33006J',30,55,20,1,4250,5500,6700,9000,0.208]);
  data(['HR30206C',30,62,16,1,3650,3850,5600,7500,0.221]);
  data(['HR32206C',30,62,20,1,4900,5750,6000,8000,0.293]);
  data(['HR33206J',30,62,25,1,6800,8100,6000,8000,0.355]);
  data(['HR30306C',30,72,19,1.5,5800,5650,5300,7100,0.382]);
  data(['HR32306J',30,72,27,1.5,8150,9000,5600,7500,0.569]);
  data(['HR320/32XJ',32,58,17,1,3800,4800,6300,8500,0.191]);
  data(['HR302/32C',32,65,17,1,4650,5350,5600,7500,0.273]);
  data(['HR322/32C',32,65,21,1,5050,6100,5600,7500,0.335]);
data(['HR303/32C',32,75,20,1.5,6050,6950,5000,6700,0.459]);
data(['HR32907J',35,55,14,0.6,2790,3950,6300,8500,0.123]);
data(['HR32007XJ',35,62,18,1,4400,5650,5600,5600,8000,0.229]);
data(['HR33007J',35,62,21,1,4950,6650,5600,8000,0.267]);
data(['HR30207C',35,72,17,1.5,4750,5550,5000,6700,0.33]);
data(['HR32207C',35,72,23,1.5,6200,7300,5000,7100,0.441]);
data(['HR33207J',35,72,28,1.5,8850,11100,5300,7100,0.54]);
data(['HR30307DJ',35,80,21,2,6350,6950,4300,6000,0.517]);
data(['HR32307C',35,80,31,2,9050,11200,4800,6300,0.782]);
data(['HR32908J',40,62,15,0.6,3450,4800,5600,7500,0.161]);
data(['HR32008XJ',40,68,19,1,5400,7250,5300,7100,0.279]);
data(['HR33008J',40,68,22,1,6000,8300,5300,7100,0.322]);
data(['HR30208C',40,80,18,1.5,6050,7050,4500,6000,0.425]);
data(['HR32208J',40,80,23,1.5,7900,9200,4800,6300,0.547]);
data(['HR33208J',40,80,32,1.5,10900,14000,4800,6300,0.744]);
data(['HR30308DJ',40,90,23,2,8150,9150,3800,5300,0.726]);
data(['HR32308C',40,90,33,2,10900,13100,4300,5600,1.03]);
data(['HR32909J',45,68,15,0.6,3600,5200,5000,6700,0.187]);
data(['HR32009XJ',45,75,20.1,6150,8450,4500,6300,0.353]);
data(['HR33009J',45,75,24,1,6800,9650,4800,6300,0.414]);
data(['HR33109J',45,80,26,1.5,8500,11600,4500,6000,0.552]);
data(['HR30209C',45,85,19,1.5,6450,8000,4300,5600,0.475]);
data(['HR32209J',45,85,23,1.5,8500,10400,4300,6000,0.601]);
data(['HR33209J',45,85,32,1.5,11300,15000,4300,6000,0.817]);
data(['T7FC045',45,95,26.5,2.5,9250,11800,3600,5000,0.945]);
data(['HR30309DJ',45,100,25,2,9750,11100,3400,4800,0.955]);
data(['HR32309C',45,100,36,2,13600,17100,3800,5000,1.42]);
data(['HR32910J',50,72,15,0.6,3800,5800,4500,6300,0.193]);
data(['HR32010XJ',50,80,20,1,6250,8900,4300,6000,0.379]);
data(['HR33010J',50,80,24,1,7150,10600,4300,6000,0.452]);
data(['HR33110J',50,85,26,1.5,9100,12800,4300,5600,0.597]);
data(['HR30210C',50,90,20,1.5,7150,9500,3800,5300,0.54]);
data(['HR32210CJ',50,90,23,1.5,8000,10500,3800,5300,0.655]);
data(['HR33210J',50,90,32,1.5,12100,16800,4000,5300,0.867]);
data(['T2ED050',50,100,35,2.5,14600,18800,3800,5000,1.3]);
data(['HR30310DJ',50,110,27,2.5,11700,13400,3200,4300,1.25]);
data(['HR32310J',50,110,40,2.5,17900,22400,3600,4800,1.88]);
data(['HR32911J',55,80,17,1,4600,7600,4300,5600,0.281]);
data(['HR32011XJ',55,90,23,1.5,8300,11900,3800,5300,0.567]);
data(['HR33011J',55,90,27,1.5,9600,14600,3800,5300,0.672]);
data(['HR33111J',55,95,30,1.5,11500,16100,3800,5000,0.877]);
data(['HR30211J',55,100,21,2,9650,1150,3600,5000,0.733]);
data(['HR32211J',55,100,25,2,11200,14000,3600,5000,0.857]);
data(['HR33211J',55,100,35,2,14400,19700,3600,5000,1.18]);
data(['T7FC055',55,115,31,3,12800,16700,3000,4300,1.59]);
data(['HR30311DJ',55,120,29,2.5,13400,15600,2800,4000,1.57]);
data(['HR32311J',55,120,43,2.5,20800,26300,3200,4300,2.39]);
data(['HR32912J',60,85,17,1.5000,8650,3800,5300,0.306]);
data(['HR32012XJ',60,95,23,1.5,8700,12900,3600,5000,0.607]);
data(['HR33012J',60,95,27,1.5,9800,15300,3600,5000,0.713]);
data(['HR33112J',60,100,30,1.5,11700,16900,3400,4800,0.91]);
data(['HR30212J',60,110,22,2,10600,12500,3400,4500,0.927]);
data(['32212C',60,110,28,2,10700,13200,3200,4500,1.11]);
data(['HR33212J',60,110,38,2,16900,23600,3400,4500,1.56]);
data(['T7FC060',60,125,33.5,3,15400,20100,2800,3800,2.04]);
data(['HR30312DJ',60,130,31,3,15400,18100,2600,3800,1.98]);
data(['32312C',60,130,46,3,20000,25400,2800,3800,2.85]);
data(['HR32913J',65,90,17,1,5000,8800,3600,5000,0.323]);
data(['HR32013XJ',65,100,23,1,5,8800,13500,3400,45000,0.645]);
data(['HR33013J',65,100,27,1.5,9950,15900,3400,45000,0.758]);
```

```
data（［'HR33113J',65,110,34,1.5,15100,22200,3200,4300,1.32］）；
data（［'HR30213J',65,120,23,2,12500,15400,3000,4000,1.18］）；
data（［'HR32213J',65,120,31,2,16000,20600,3000,4000,1.55］）；
data（［'HR33213J',65,120,41,2,20600,28800,3000,4000,2.04］）；
data（［'HR30313DJ',65,140,33,3,17700,20900,2400,3400,2.42］）；
data（［'HR32313J',65,140,48,3,27300,35000,2800,3800,3.6］）；
data（［'HR32914J',70,100,20,1,7150,11500,3200,4500,0.493］）；
data（［'HR32014XJ',70,110,25,1.5,10600,16100,3200,4300,0.869］）；
data（［'HR33014J',70,110,31,1.5,12900,20800,3000,4300,1.11］）；
data（［'HR33114J',70,120,37,2,18100,26700,3000,4000,1.71］）；
data（［'HR30214J',70,125,24,2,13500,16700,2800,4000,1.3］）；
data（［'HR32214J',70,125,31,2,16100,20900,2800,4000,1.64］）；
data（［'HR33214J',70,125,41,2,21300,30500,2800,4000,2.15］）；
data（［'T7FC070',70,140,35.5,3,18000,23400,2400,3400,2.55］）；
data（［'HR30314DJ',70,150,35,3,19600,23300,2200,3200,2.93］）；
data（［'32314C',70,150,51,3,26700,34500,2400,3400,4.23］）；
data（［'HR32915J',75,105,20,1,7400,12300,3200,4300,0.53］）；
data（［'HR32015XJ',75,115,25,1.5,11100,17400,3000,4000,0.925］）；
data（［'HR33015J',75,115,31,1.5,13500,22500,3000,4000,1.18］）；
data（［'HR33115J',75,125,37,2,18600,28100,2800,3800,1.8］）；
data（［'HR30215J',75,130,25,2,14600,18500,2800,3800,1.42］）；
data（［'HR32215J',75,130,31,2,16900,22400,2800,3800,1.72］）；
data（［'HR33215J',75,130,41,2,21900,32000,2800,3800,2.25］）；
data（［'HR30315DJ',75,160,37,3,21500,25600,2200,3000,3.47］）；
data（［'323115CA',75,160,55,3,32000,43000,2200,3200,5.29］）；
data（［'HR32916C',80,110,20,1,7650,13100,3000,4000,0.56］）；
data（［'HR32016XJ',80,125,29,1.5,14300,22700,2800,3600,1.32］）；
data（［'HR33116J',80,130,37,2,19000,29400,2600,3600,1.88］）；
data（［'30216CA',80,140,26,2.5,15000,19400,2400,3400,1.65］）；
data（［'HR32216J',80,140,33,2.5,19600,25900,2600,3400,2.13］）；
data（［'HR33216J',80,140,46,2.5,26200,39000,2600,3400,2.94］）；
data（［'HR30316DJ',80,170,39,3,24000,28900,2000,2800,4.06］）；
data（［'HR32316J',80,170,58,3,39000,51500,2200,3000,6.34］）；
data（［'HR32917J',85,120,23,1.5,9550,16000,2800,3800,0.80］）；
data（［'HR32017XJ',85,130,29,1.5,14600,23600,2600,3600,1.38］）；
data（［'HR33117J',85,140,41,2.5,23500,37000,2400,3400,2.51］）；
data（［'30217CA',85,150,28,2.5,17500,23000,2200,3200,2.07］）；
data（［'HR32217J',85,150,36,2.5,21400,28200,2400,3200,2.63］）；
data（［'HR33217J',85,150,49,2.5,28700,42500,2400,3200,3.56］）；
data（［'30317DX',85,180,41,4,24600,28600,1900,2600,4.61］）；
data（［'HR32317J',85,180,60,4,42000,54500,2000,2800,7.3］）；
data（［'HR32918J',90,125,23,1.5,9850,17000,2600,3600,0.839］）；
data（［'HR32018XJ',90,140,32,2,17300,27800,2400,3200,1.78］）；
data（［'HR33118J',90,150,45,2.5,26500,41500,2400,3200,3.14］）；
data（［'HR30218J',90,160,30,2.5,20500,26100,2200,3000,2.6］）；
data（［'HR32218J',90,160,40,2.5,26100,35500,2200,3000,3.41］）；
data（［'30318DX',90,190,43,4,25500,29700,1800,2400,5.39］）；
data（［'HR32919J',95,130,23,1.5,10000,17500,2400,3400,0.875］）；
data（［'HR32019XJ',95,145,32,2,17600,28900,2400,3200,1.87］）；
data（［'HR30219J',95,170,32,3,22800,29200,2200,2800,3.12］）；
data（［'HR32219J',95,170,43,3,29500,40500,2200,2800,4.21］）；
data（［'HR30319DJ',95,200,45,4,31500,38500,1700,2400,6.63］）；
data（［'32319',95,200,67,4,47000,61500,1900,2600,9.64］）；
data（［'HR32920J',100,140,25,1.5,12000,20900,2200,3200,1.18］）；
data（［'T4CB100',100,145,22.5,3,11500,16600,2200,3000,1.18］）；
data（［'HR32020XJ',100,150,32,2,17900,30000,2200,3000,1.94］）；
data（［'HR30220J',100,180,34,3,26000,34000,2000,2600,3.78］）；
data（［'HR32220J',100,180,46,3,33000,46000,2000,2600,5.05］）；
data（［'30320',100,215,47,4,37500,4400,1700,2400,7.93］）；
data（［'HR3230J',100,215,73,4,57500,77000,1700,2400,12.6］）；
```

```
 data(['HR32921J',105,145,25,1.5,12100,21600,2200,3000,1.23]);
 data(['HR32021XJ',105,160,35,2.5,20800,34500,2000,2800,2.48]);
 data(['HR30221J',105,190,36,3,28500,37500,1900,2600,4.51]);
 data(['HR32221J',105,190,50,3,37000,5200,1900,2600,6.25]);
 data(['30321',105,225,49,4,40500,48000,1600,2200,9.1]);
 data(['32321',105,225,77,4,59500,79500,1700,2200,14.2]);
 data(['HR32922J',110,150,25,1.5,12500,22800,2200,2800,1.29]);
 data(['HR32022XJ',110,170,38,2.5,24000,40000,2000,2600,3.08]);
 data(['HR30222J',110,200,38,3,32000,43000,1800,2400,5.27]);
 data(['HR32222J',110,200,53,3,40500,57500,1800,2400,7.34]);
 data(['30322',110,240,50,4,41500,48500,1500,2000,10.5]);
 data(['32322',110,240,80,4,63500,84500,1600,2200,16.8]);
End.

class pulley3v
part
    key:= #key;
parameter
model_num string,
num_of_grooves integer,
ext_dia integer 'mm',
max_int_dia integer 'mm',
int_dia integer 'mm',
key_height integer 'mm';
where
int_dia == key!shaft_dia;
end.

class belt3v
parameter
model_num integer,
circum_length integer 'mm',
req_shaft_dist integer 'mm',
shaft_dist integer 'mm',
small_ply_rot integer 'rpm',
power integer 'KW',
ply_ratio real,
drv_pulley_dia integer 'mm',
flw_ply_dia integer 'mm',
small_ply_dia integer 'mm',
large_ply_dia integer 'mm',
power_capa real 'KW',
num_of_belt integer [];
where
bigger(@drv_pulley_dia,@flw_ply_dia) = ^large_ply_dia,
smaller(@drv_pulley_dia,@flw_ply_dia) = ^small_ply_dia;
end.

class belt_drive
part
drv_ply:= #pulley3v,
drv_shaft:= #shaft,
flw_ply:= #pulley3v,
flw_shaft:= #shaft,
belt:= #belt3v;
parameter
ply_ratio real ,
drv_ply_dia integer 'mm',
flw_ply_dia integer 'mm',
req_shaft_dist integer 'mm',
shaft_dist integer 'mm',
drv_max_rot integer 'rpm',
drv_base_rot integer 'rpm',
drv_max_trq real 'kg.m',
flw_max_rot integer 'rpm',
flw_base_rot integer 'rpm',
flw_max_trq real 'kg.m',
power_capa real 'KW',
belt_coef real,
trns_power real 'KW',
power_coef real,
rq_trns_power real 'KW';
where
drv_shaft!req_power == flw_shaft!req_power;
drv_shaft!max_rot_trq == flw_shaft!max_rot_trq;
drv_shaft!req_power == belt!rq_trns_power;
drv_ply_dia!int_dia == drv_shaft!ext_dia;
flw_ply_dia!int_dia == flw_shaft!ext_dia;
drv_ply_dia == drv_ply!ext_dia;
drv_ply_dia == belt!drv_ply_dia;
flw_ply_dia == flw_ply!ext_dia;
flw_ply_dia == belt!flw_ply_dia;
req_shaft_dist == belt!req_shaft_dist;
shaft_dist == belt!shaft_dist;
drv_max_rot == drv_shaft!max_rot;
drv_base_rot ==drv_shaft!base_rot;
drv_max_trq == drv_shaft!max_trq;
flw_max_rot == flw_shaft!max_rot;
flw_base_rot == flw_shaft!base_rot;
flw_max_trq == flw_shaft!max_trq;
drv shaft!max_base_ratio == flw_shaft!max_base_ratio;
drv_ply_dia * @ply_ratio = flw_ply_dia;
@drv_max_trq * @ply_ratio = ^flw_max_trq;
@drv_max_rot / @ply_ratio = ^flw_max_rot;
@drv_base_rot / @ply_ratio = ^flw_base_rot;
end.

class lathe_machine
part
   rot_main_shaft:= #rot_main_shaft,
   rot_counter_shaft:= #rot_counter_shaft,
   series_of_gears:= #series_of_gears,
   motor:= #motor,
   belt_drive:= #belt_drive;
where
   rot_main_shaft!main_shaft == series_of_gears!flw_shaft;
   rot_counter_shaft!counter_shaft == series_of_gears!drv_shaft;
   rot_main_shaft!main_shaft!material == rot_counter_shaft!counter_shaft!material;
   motor!flw_shaft == belt_drive!drv_shaft;
   rot_counter_shaft!counter_shaft == belt_drive!flw_shaft;
end.
```

付録５　FDLによるエレベータモデル記述

ここではエレベータ設計（ＶＴ）問題のＦＤＬ表現、要求仕様、生成検証法のための値生成器、設計計算結果を記述する。

（１）エレベータ設計問題のＦＤＬ記述

```
class car
parameter
   car_cable_hanger_weight integer [],
   car_duct_and_wiring_weight integer [],
   car_guideshoe_weight integer [],
   car_landing_switch_weight integer [],
   car_limit_switch_cam_weight integer [],
   car_maintenance_station_weight integer [],
   car_supplement_weight integer [>=0,=<500],
   model_id string [];
where
   data(^car_cable_hanger_weight, ^car_duct_and_wiring_weight, ^car_guideshoe_weight,
      ^car_landing_switch_weight, ^car_limit_switch_cam_weight, ^car_maintenance_station_weight, @model_id) ;
local
   data(30,45,18,20,35,7,car1) ;
end.

class car_buffer_model
parameter
   car_buffer_quantity_at_most integer [],
   car_buffer_load_total real [],
   car_buffer_stroke_minimum real [],
   car_buffer_footing_channel_height real [],
   car_buffer_height real [],
   car_buffer_load_maximum integer [],
   car_buffer_load_minimum integer [],
   car_buffer_stroke integer [],
   car_buffer_quantity integer [],
   model_id string [];
where
   data(^car_buffer_footing_channel_height, ^car_buffer_height, ^car_buffer_load_maximum,
       ^car_buffer_load_minimum,  ^car_buffer_stroke,    ^car_buffer_quantity, ^model_id) ,
   @car_buffer_quantity_at_most >= ^car_buffer_quantity,
   @car_buffer_load_total/ ^car_buffer_quantity =<  ^car_buffer_load_maximum,
   @car_buffer_load_total/ ^car_buffer_quantity >= ^car_buffer_load_minimum,
   ^car_buffer_stroke >= @car_buffer_stroke_minimum;
local
   data(3.5,28.75,11000,2900,8.25,1, model_oh1) ;
   data(3.5,38.5, 11000,2900,14,1, model_om14) ;
   data(3.5,28.75,11000,2900,8.25,2, model_oh1) ;
   data(3.5,38.5, 11000,2900,14,2, model_om14) ;
end.

class car_guiderail_model
parameter
   model_id string [],
   car_guiderail_weight integer [];
where
   data(@model_id, ^car_guiderail_weight) ;
local
   data(car_guiderail_model_m01,8) ;
   data(car_guiderail_model_m02,11) ;
   data(car_guiderail_model_m03,16) ;
   data(car_guiderail_model_m04,18.5) ;
   data(car_guiderail_model_m05,32) ;
end.

class compensation_cable_model
parameter
   compensation_cable_unit_weight_optimal real [],
   compensation_cable_quantity_c02 integer [default(0)],
   compensation_cable_unit_weight real [default(0)],
   model_id string [default(compensation_cable_model_m07)];
where
   data(^compensation_cable_quantity_c02,  ^compensation_cable_unit_weight, ^model_id) ,
   @compensation_cable_unit_weight_optimal =<  ^compensation_cable_unit_weight;
local
   data(0,0,compensation_cable_model_m07) ;
   data(2,0.05,compensation_cable_model_m03) ;
   data(2,0.0848,compensation_cable_model_m02) ;
   data(2,0.1171,compensation_cable_model_m05) ;
   data(2,0.1617,compensation_cable_model_m04) ;
   data(2,0.2117,compensation_cable_model_m06) ;
   data(2,0.2657,compensation_cable_model_m01) ;
end.
```

```
class control_cable_model
parameter
  control_cable_unit_weight  real [],
  model_id string [];
where
  data (^control_cable_unit_weight, @model_id) ;
local
  data (0.167,control_cable_model_m01) ;
  data (0.199,control_cable_model_m02) ;
  data (0.209,control_cable_model_m03) ;
end.

class counterweight_buffer_model
super vt_component;
parameter
  counterweight_buffer_height real [],
  counterweight_buffer_load_maximum integer [],
  counterweight_buffer_load_minimum integer [],
  counterweight_buffer_stroke real [],
  model_id string [];
where
  data (^counterweight_buffer_height, ^counterweight_buffer_load_maximum,
        ^counterweight_buffer_load_minimum, ^counterweight_buffer_stroke, @model_id) ;
local
  data (28.75,11000,2900,8.25,model_m01) ;
  data (38.5,11000,2900,14,model_m02) ;
end.

class counterweight_guard_required
parameter
  counterweight_guard_thickness integer [],
  model_id string [];
where
  data (^counterweight_guard_thickness, @model_id) ;
local
  data (0,counterwtguard_thickness_m01) ;
  data (0.75,counterwtguard_thickness_m02) ;
end.

class counterweight_guiderail_model
parameter
  counterweight_weight real [],
  counterweight_guiderail_unit_weight  integer [],
  counterweight_weight_limit  integer [],
  model_id string [];
where
  data (^counterweight_guiderail_unit_weight, ^counterweight_weight_limit, ^model_id) ,
  ^counterweight_weight_limit >= @counterweight_weight;
local
  data (8,15000,counterweight_guiderail_model_m01) ;
  data (11,27000,counterweight_guiderail_model_m02) ;
  data (16,40000,counterweight_guiderail_model_m03) ;
  data (18.5,56000,counterweight_guiderail_model_m04) ;
end.

class crosshead_model
parameter
  crosshead_height real [],
  model_id string [];
where
  data (^crosshead_height, @model_id) ;
local
  data (10.0,crosshead_model_m01) ;
  data (13.5,crosshead_model_m02) ;
  data (8.0,crosshead_model_m03) ;
  data (8.125,crosshead_model_m04) ;
  data (8.25,crosshead_model_m05) ;
end.

class deflector_sheave_model
parameter
  hoist_cable_diameter real [],
  deflector_sheave_diameter_s integer [],
  deflector_sheave_protrusion real [],
  deflector_sheave_weight integer [],
  model_id string [];
where
  data (^deflector_sheave_diameter_s, ^deflector_sheave_protrusion,^deflector_sheave_weight, ^model_id) ,
  @hoist_cable_diameter * 40 =< ^deflector_sheave_diameter_s;
local
  data (20,15.125,250,deflector_sheave_model_m01) ;
  data (25,18.125,350,deflector_sheave_model_m02) ;
end.
```

```
class door_model
parameter
   car_saf_t_edge_weight integer [],
   door_opening_strike_side string [],
   door_opening_type string [member([center,side])],
   door_operator_constant real [],
   door_operator_engine_weight integer [default([135])],
   door_operator_header_weight real [],
   door_operator_weight real [],
   door_space real [],
   opening_width_door real [],
   door_speed string [member([single,double])],
   model_id string [];
where
   data(^car_saf_t_edge_weight, @door_opening_strike_side, @door_opening_type,
         ^door_operator_constant, ^door_operator_engine_weight, @door_speed, ^model_id);
door_operator_constant * opening_width_door = door_operator_header_weight;
door_operator_engine_weight + door_operator_header_weight = door_operator_weight;
door_speed = `double ==> door_space := 6.5;
door_speed = `single ==> door_space := 5.0;
local
   data(13,na,center,2.083,135,double,door_model_code_m01);
   data(7,left,side,1.5,135,double,door_model_code_m02);
   data(7,right,side,1.5,135,double,door_model_code_m03);
   data(13,na,center,1.33,135,single,door_model_code_m04);
   data(7,left,side,1.25,135,single,door_model_code_m05);
   data(7,right,side,1.25,135,single,door_model_code_m06);
end.

class elevator
part
   car:= #car,
   carbuffer:= #car_buffer_model,
   carguiderail:= #car_guiderail_model,
   compensationcable:= #compensation_cable_model,
   controlcable:= #control_cable_model,
   counterwtbuffer:= #counterweight_buffer_model,
   counterwtguard:= #counterweight_guard_required,
   counterwtguiderail:= #counterweight_guiderail_model,
   crosshead:= #crosshead_model,
   deflectorsheave:= #deflector_sheave_model,
   door:= #door_model,
   governor:= #governor_model,
   governorcable:= #governor_cable_model,
   hoistcable:= #hoist_cable_model,
   intercom:= #car_intercom,
   lantern:= #car_lantern,
   machine:= #machine_model,
   mbeam:= #machine_beam_model,
   motgenerator:= #motor_generator_model,
   motor:= #motor,
   phone:= #car_phone,
   platform:= #platform_model,
   positionind:= #car_position_indicator,
   safety:= #safety_beam_model,
   sling:= #sling_model;
parameter
   cable_load_car_side_car_bottom real [],
   cable_load_car_side_car_top real [],
   cable_load_counterweight_side_car_bottom real [],
   cable_load_counterweight_side_car_top real [],
   cable_load_total_car_side_maximum real [],
   cable_load_unbalanced real [],
   carslingub real [],
   car_buffer_blocking_height integer [>=6,=<10000],
   car_buffer_compressed_clearance real [>=24.0, =<1000.0],
   car_buffer_load real [],
   car_buffer_load_total real [],
   car_buffer_quantity integer [>=1,=<2],
   car_buffer_quantity_at_most integer [>=1,=<2],
   car_buffer_striking_speed_maximum real [],
   car_buffer_stroke_minimum real [>=0.0,=<14.0],
   car_cable_hitch_to_counterweight_cable_hitch real [],
   car_cable_hitch_to_platform_front real [],
   car_cable_hitch_to_platform_rear real [],
   car_cab_height real [>=84.0,=<240.0],
   car_cab_weight real [],
   car_capacity_range integer [>=2000,=<4000],
   car_fixture_weight real [],
   car_guiderail_vertical_force real [],
   car_guiderail_vertical_force_maximum real [],
   car_intercom_spec string [member([yes,no])],
   car_lantern_spec string [member([yes,no])],
car_misc_weight real [],
car_overtravel real [],
car_phone_spec string [member([yes,no])],
car_position_indicator_spec string [member([yes,no])],
car_return_left real [],
car_return_right real [>=1],
car_runby integer [>= 6,=< 24],
car_speed integer [member([200,250,300,350,400])],
car_supplement_weight integer [],
car_weight real [],
compensation_cable_length real [],
compensation_cable_load_car_side_car_bottom real [],
compensation_cable_load_car_side_car_top real [],
compensation_cable_load_counterweight_side_car_bottom real [],
compensation_cable_load_counterweight_side_car_top real [],
compensation_cable_quantity integer [],
compensation_cable_unit_weight_optimal real [],
control_cable_load_average real [],
control_cable_load_car_side_car_bottom real [],
control_cable_load_car_side_car_top real [],
control_cable_loop_below_car_bottom real [],
counterweight_above_top_car_bottom real [],
counterweight_between_guiderails_value integer [member([28, 38, 54])],
counterweight_bottom_reference real [],
counterweight_buffer_blocking_height integer [>=0,=<120,default(0)],
counterweight_buffer_footing_channel_height  integer [default([1])],
counterweight_buffer_load real [],
counterweight_buffer_quantity integer [>=1,=<2],
counterweight_frame_height real [>=90.0, =<174.0, default(138.0)],
counterweight_frame_thickness real [default([31.0])],
counterweight_frame_weight real [],
counterweight_frame_weight_s real [],
counterweight_overtravel real [],
counterweight_plate_depth real [>=7.0,=<12.0, default(7.0)],
counterweight_plate_quantity integer [],
counterweight_plate_thickness integer [default([1])],
counterweight_plate_weight real [],
counterweight_runby real [>=6.0,=<36.0],
counterweight_space real [],
counterweight_stack_height real [],
counterweight_stack_weight real [],
counterweight_stack_weight_desired real [],
counterweight_system_weight real [],
counterweight_to_hoistway_rear real [],
counterweight_to_platform_rear real [],
counterweight_u_bracket_protrusion  real [default([0.75])],
counterweight_weight real [=<56000],
counterweight_weight_desired real [],
crosshead_bending_moment real [],
crosshead_deflection_index real [],
deflector_sheave_diameter integer [],
deflector_sheave_protrusion real [],
door_opening_strike_side string [member([left, right])],
door_opening_type string [member([center,side])],
door_operator_header_weight real [],
door_operator_weight real [],
door_space real [],
door_speed string [member([single, double])],
governor_cable_length real [],
hoistway_bracket_spacing real [],
hoistway_depth real [],
hoistway_floor_height real [],
hoistway_overhead real [>=0.0,=<360.0],
hoistway_pit_depth real [>=0.0,=<240.0],
hoistway_top_landing_to_underside_machine_beam real [],
hoistway_travel real [],
hoistway_width real [],
hoist_cable_above_counterweight_car_bottom real [],
hoist_cable_length real [],
hoist_cable_load_car_side_car_bottom real [],
hoist_cable_load_car_side_car_top real [],
hoist_cable_load_counterweight_side_car_bottom real [],
hoist_cable_load_counterweight_side_car_top real [],
hoist_cable_quantity integer [>=3,=<6],
hoist_cable_safety_factor real [],
hoist_cable_safety_factor_minimum real [],
hoist_cable_traction_factor real [default([1.304])],  %modified 98.8.13
hoist_cable_traction_ratio real [],
hoist_cable_traction_ratio_car_bottom_empty real [],
hoist_cable_traction_ratio_car_bottom_full real [],
hoist_cable_traction_ratio_car_top_empty real [],
hoist_cable_traction_ratio_car_top_full real [],
machine_acceleration_maximum  real [default([4.25])],
machine_angle_of_contact real [],
    machine_beam_bearing_plate_thickness integer [default([1])],
    machine_beam_bending_moment_left_maximum real [],
    machine_beam_bending_moment_right_maximum real [],
    machine_beam_center_to_machine_sheave_center real [],
    machine_beam_impact_load_left real [],
    machine_beam_impact_load_right real [],
    machine_beam_impact_load_total real [],
    machine_beam_length real [],
    machine_beam_load_front_left real [],
    machine_beam_load_front_right real [],
    machine_beam_load_rear_left real [],
    machine_beam_load_rear_right real [],
    machine_beam_reaction_front_left real [],
    machine_beam_reaction_front_right real [],
    machine_beam_reaction_rear_left real [],
    machine_beam_reaction_rear_right real [],
    machine_beam_safety_factor real [default([1.05])],
    machine_beam_section_modulus_left_minimum real [],
    machine_beam_section_modulus_right_minimum real [],
    machine_beam_support_bottom_to_machine_room_top real [],
    machine_beam_support_bottom_to_machine_room_top_spec real [],
    machine_beam_support_distance real [],
    machine_beam_support_front_to_hoistway real [],
    machine_beam_support_to_machine_beam_center real [],
    machine_beam_support_type string [member([pocket,steel])],
    machine_beam_total_load_left real [],
    machine_beam_total_load_right real [],
    machine_beam_weight real [],
    machine_efficiency real [],
    machine_gear_ratio_numeric real [],
    machine_groove_pressure real [],
    machine_groove_pressure_acceptable_maximum real [],
    machine_groove_pressure_constant real [],
    machine_sheave_center_to_front_machine_beam_support real [],
    machine_sheave_height integer [],
    machine_sheave_to_deflector_sheave real [],
    machine_sheave_to_deflector_sheave_diagonal real [],
    machine_sheave_to_deflector_sheave_horizontal real [],
    msheavegroove_model_id string [],
    machine_suspended_load real [],
    machine_total_weight real [],
    minacceptablembeammodulus real [],
    motor_current_supported_maximum integer [default([150])],
    motor_horsepower_required real [>=0.0,=<40.0],
    motor_peak_current_required real [],
    motor_torque_releveling real [],
    opening_count real [],
    opening_height real [],
    opening_to_hoistway_left integer [],
    opening_to_hoistway_right integer [],
    opening_width_building real [],
    opening_width_door real [>=0.0,=<72.0],
    platform_depth_spec integer [],
    platform_running_clearance real [default([1.25])],
    platform_to_hoistway_front real [],
    platform_to_hoistway_left integer [>= 8],
    platform_to_hoistway_right integer [>= 8],
    platform_weight real [],
    platform_weight_factor_ap integer [>=3,=<10],
    platform_weight_factor_s integer [member([63, 71])],
    platform_weight_factor_x real [],
    platform_weight_factor_z real [],
    platform_width integer [>= 60],
    safety_beam_bending_moment real [],
    safety_beam_between_guiderails real [],
    safety_beam_bg real [],
    safety_beam_load real [],
    safety_beam_weight real [],
    sling_stile_length real [],
    sling_underbeam real [],
    sling_underbeam_space real [>=12,=<1000,default(21)],
    sling_weight real [];
where
car_intercom_spec == intercom!spec;     car_lantern_spec == lantern!spec;
car_phone_spec == phone!spec;          car_position_indicator_spec == positionind!spec;
car_buffer_quantity_at_most == carbuffer!car_buffer_quantity_at_most;
car_buffer_quantity == carbuffer!car_buffer_quantity;
car_buffer_load_total == carbuffer!car_buffer_load_total;
car_buffer_stroke_minimum == carbuffer!car_buffer_stroke_minimum;
car_supplement_weight == car!car_supplement_weight;
compensation_cable_unit_weight_optimal ==
compensationcable!compensation_cable_unit_weight_optimal;
compensationcable!compensation_cable_quantity_c02 == compensation_cable_quantity;
counterweight_weight == counterwtguiderail!counterweight_weight;
```

```
crosshead_bending_moment == sling!crosshead_bending_moment;
crosshead_deflection_index == sling!crosshead_deflection_index;
door_opening_strike_side == door!door_opening_strike_side;  door_opening_type == door!door_opening_type;
door_operator_header_weight == door!door_operator_header_weight;
door_operator_weight == door!door_operator_weight;
door_speed == door!door_speed;    door_space == door!door_space;
opening_width_door == door!opening_width_door;    opening_width_building == opening_width_door;
hoistcable!hoist_cable_diameter == deflectorsheave!hoist_cable_diameter;
hoist_cable_quantity == hoistcable!hoist_cable_quantity_f;   hoistway_floor_height == hoistway_bracket_spacing;
machine!machine_efficiency == machine_efficiency;
machine!machine_gear_ratio_numeric == machine_gear_ratio_numeric;
machine!machine_sheave_height_f == machine_sheave_height;
motor_horsepower_required == motor!motor_horsepower_required;
platform_depth_spec == platform!platform_depth;    platform_width == platform!platform_width_f;
car_capacity_range == platform!car_capacity;
deflectorsheave!deflector_sheave_diameter_s == deflector_sheave_diameter;

platform!model_id = `platform_model_m01 ==> compensation_cable_load_car_side_car_top =< 600;

counterweight_buffer_quantity = 2 ==> counterweight_between_guiderails_value >= 38;

counterwtbuffer!counterweight_buffer_load_minimum =< counterweight_buffer_load,
counterweight_buffer_load =< counterwtbuffer!counterweight_buffer_load_maximum;

car_runby + 1.5*carbuffer!car_buffer_stroke + 6 =< counterweight_overtravel;

counterweight_stack_height =< counterweight_frame_height - counterweight_frame_thickness;

counterweight_to_hoistway_rear >= 0.75 + counterweight_u_bracket_protrusion;

counterwtguard!model_id = `counterwtguard_thickness_m01 ==>   counterweight_to_platform_rear >= 1;

counterwtguard!model_id ¥== `counterwtguard_thickness_m01 ==>
counterweight_to_platform_rear >= 1.5 + counterwtguard!counterweight_guard_thickness;

sys:member(machine!model_id,[`machine_model_m02, `machine_model_m03])==>
3 =< hoist_cable_quantity, hoist_cable_quantity =< 6;

machine!model_id = `machine_model_m01 ==>   3 =< hoist_cable_quantity, hoist_cable_quantity =< 5;

machine!model_id = `machine_model_m04 ==>   5 =< hoist_cable_quantity, hoist_cable_quantity =< 6;

hoist_cable_safety_factor >= hoist_cable_safety_factor_minimum;

(hoistcable!hoist_cable_ultimate_strength * hoist_cable_quantity)  / cable_load_car_side_car_top =
hoist_cable_safety_factor;

@minacceptablembeammodulus =< @mbeam!machine_beam_section_modulus;

machine_beam_section_modulus_right_minimum > machine_beam_section_modulus_left_minimum ==>
minacceptablembeammodulus = machine_beam_section_modulus_right_minimum;

machine_beam_section_modulus_right_minimum =< machine_beam_section_modulus_left_minimum ==>
minacceptablembeammodulus = machine_beam_section_modulus_left_minimum;

machine_groove_pressure =< machine_groove_pressure_acceptable_maximum;

machine!model_id = `machine_model_m01 ==> sys:member(motor!model_id,[`motor_model_m01,`motor_model_m02,`nil]);

machine!model_id  =  `machine_model_m02  ==>  sys:member(motor!model_id,[`motor_model_m02,  `motor_model_m03,`nil]);

machine!model_id = `machine_model_m03 ==>
sys:member(motor!model_id, [`motor_model_m03, `motor_model_m04, `motor_model_m05, `motor_model_m06, `nil]);

machine!model_id = `machine_model_m04 ==> sys:member(motor!model_id, [`motor_model_m06, `nil]);

opening_height >= 78, opening_height =< car_cab_height;

platform!model_id = `platform_model_m01, door_opening_type = `center ==> opening_width_door =< platform_width / 2 - 12;

platform!model_id = `platform_model_m01, door_opening_type = `side ==> opening_width_door =< platform_width / 2 - 6;

safety_beam_load =< safety!safety_beam_load_maximum;

platform!model_id = `platform_model_m01,
sys:member(sling!model_id, [`sling_model_m03, `sling_model_m04,`sling_model_m05])  ==>
platform!model_id := `platform_model_m02;

sys:member(sling!model_id,[`sling_model_m01, `sling_model_m02]),sling_underbeam > 123 ==>
sling!model_id := `sling_model_m04;

sling!model_id = `sling_model_m04, sling_underbeam >  153 ==>sling!model_id := `sling_model_m03;

sling!model_id = `sling_model_m03, sling_underbeam >  189.16,
sys:member(safety!model_id,[`safety_beam_model_m01,`safety_beam_model_m02])  ==>sling!model_id  :=  `sling_model_m05;

sling!model_id = `sling_model_m03, safety!model_id = `safety_beam_model_m03,
sling_underbeam > 188.55 ==> sling!model_id := `sling_model_m05;

@machine_suspended_load =< @machine!machine_suspended_load_maximum;

msheavegroove_model_id = `machine_groove_model_m01 ==>
hoist_cable_traction_ratio =< 0.00788800000000001 * machine_angle_of_contact + 0.675;

msheavegroove_model_id = `machine_groove_model_m02 ==>
hoist_cable_traction_ratio =< 0.006555 * machine_angle_of_contact + 0.755;

(car_weight + car_capacity_range)  / car_buffer_quantity = car_buffer_load;

(car_weight + car_capacity_range)  = car_buffer_load_total;

car_guiderail_vertical_force =< car_guiderail_vertical_force_maximum;
```

```
sys:member(safety!model_id,[`safety_beam_model_m01,`safety_beam_model_m02]) ==> car_buffer_quantity_at_most := 1;

safety!model_id = `safety_beam_model_m03 ==> car_buffer_quantity_at_most = 2;

counterweight_runby + 1.5*counterwtbuffer!counterweight_buffer_stroke + 24 =< car_overtravel;

door_operator_weight +  door!car_saf_t_edge_weight + 155 = car_misc_weight;

hoist_cable_load_car_side_car_bottom + compensation_cable_load_car_side_car_bottom +
control_cable_load_car_side_car_bottom +car_capacity_range + car_weight = cable_load_car_side_car_bottom;

hoist_cable_load_car_side_car_top + compensation_cable_load_car_side_car_top +
control_cable_load_car_side_car_top + car_capacity_range + car_weight = cable_load_car_side_car_top;

cable_load_car_side_car_top > cable_load_car_side_car_bottom ==>
cable_load_total_car_side_maximum = cable_load_car_side_car_top;

cable_load_car_side_car_top =< cable_load_car_side_car_bottom ==>
cable_load_total_car_side_maximum = cable_load_car_side_car_bottom;

hoist_cable_load_counterweight_side_car_bottom + (compensation_cable_load_counterweight_side_car_bottom +
counterweight_weight) = cable_load_counterweight_side_car_bottom;

hoist_cable_load_counterweight_side_car_top +compensation_cable_load_counterweight_side_car_top +
counterweight_weight = cable_load_counterweight_side_car_top;

hoistway_travel * (hoistcable!hoist_cable_quantity_f * hoistcable!hoist_cable_unit_weight -
(compensationcable!compensation_cable_quantity_c02 * compensationcable!compensation_cable_unit_weight +
0.25 * controlcable!control_cable_unit_weight)) = cable_load_unbalanced;

floor(hoistway_pit_depth - (6 + platform!platform_height +safety!safety_beam_height +
carbuffer!car_buffer_height +carbuffer!car_buffer_footing_channel_height)) = car_buffer_blocking_height;

1.15 * car_speed = car_buffer_striking_speed_maximum;

(car_buffer_striking_speed_maximum * car_buffer_striking_speed_maximum) / 19320 = car_buffer_stroke_minimum;

130 * (platform_width + platform!platform_depth) / 12 = car_cab_weight;

platform!platform_depth =< 60 ==> (platform!platform_depth / 2) - 2 = car_cable_hitch_to_platform_front;

platform!platform_depth > 60, platform!platform_depth =< 66 ==>
(platform!platform_depth / 2) - 2.5 = car_cable_hitch_to_platform_front;

platform!platform_depth > 66 ==> (platform!platform_depth / 2) - 4 = car_cable_hitch_to_platform_front;

platform!platform_depth - car_cable_hitch_to_platform_front = car_cable_hitch_to_platform_rear;

car_cable_hitch_to_platform_rear + counterweight_to_platform_rear +counterweight_plate_depth / 2 =
car_cable_hitch_to_counterweight_cable_hitch;

hoistway_top_landing_to_underside_machine_beam -sling_underbeam + crosshead!crosshead_height =  car_overtravel;

intercom!car_intercom_weight + phone!car_phone_weight +lantern!car_lantern_weight +
positionind!car_position_indicator_weight = car_fixture_weight;

platform!car_capacity + car_weight + compensation_cable_load_car_side_car_top +
control_cable_load_car_side_car_top = car_guiderail_vertical_force;

carguiderail!model_id = `car_guiderail_model_m01, hoistway_bracket_spacing =< 72 ==>
car_guiderail_vertical_force_maximum := 5500;

carguiderail!model_id = `car_guiderail_model_m01, hoistway_bracket_spacing > 72 ==>
7750-31.25*hoistway_bracket_spacing = car_guiderail_vertical_force_maximum;

carguiderail!model_id = `car_guiderail_model_m02, hoistway_bracket_spacing =< 126 ==>
car_guiderail_vertical_force_maximum := 10000;

carguiderail!model_id = `car_guiderail_model_m02, hoistway_bracket_spacing > 126 ==>
16000-47.619*hoistway_bracket_spacing = car_guiderail_vertical_force_maximum;

carguiderail!model_id = `car_guiderail_model_m03, hoistway_bracket_spacing =< 120 ==>
car_guiderail_vertical_force_maximum := 15000;

carguiderail!model_id = `car_guiderail_model_m03, hoistway_bracket_spacing > 120 ==>
22500-62.5* hoistway_bracket_spacing = car_guiderail_vertical_force_maximum;

carguiderail!model_id = `car_guiderail_model_m04, hoistway_bracket_spacing =< 108 ==>
car_guiderail_vertical_force_maximum := 21000;

carguiderail!model_id = `car_guiderail_model_m04, hoistway_bracket_spacing > 108 ==>
26400 - 50 * hoistway_bracket_spacing = car_guiderail_vertical_force_maximum;

carguiderail!model_id = `car_guiderail_model_m05, hoistway_bracket_spacing =< 126 ==>
car_guiderail_vertical_force_maximum := 41000;

carguiderail!model_id = `car_guiderail_model_m05, hoistway_bracket_spacing > 126 ==>
45500-35.714*hoistway_bracket_spacing = car_guiderail_vertical_force_maximum;

sys:member(door!model_id,[`door_model_code_m01,`door_model_code_m04]) ==>
(platform_width - opening_width_building)/2 = car_return_left;

sys:member(@door!model_id,[`door_model_code_m02, `door_model_code_m05]) ==> car_return_left := 3;

door!model_id = `door_model_code_m03 ==> platform_width - (opening_width_building + 3) = car_return_left;

platform_width - opening_width_building - car_return_left = car_return_right;

hoistway_pit_depth - (platform!platform_height +safety!safety_beam_height + carbuffer!car_buffer_height +
car_buffer_blocking_height + carbuffer!car_buffer_footing_channel_height) = car_runby;

car_cab_weight + platform_weight + sling_weight + safety_beam_weight + car_fixture_weight +
car!car_supplement_weight + car_misc_weight+54   = car_weight;

carbuffer!car_buffer_footing_channel_height +car_buffer_blocking_height +
carbuffer!car_buffer_height - carbuffer!car_buffer_stroke = car_buffer_compressed_clearance;

compensationcable!model_id ¥== `compensation_cable_model_m07 ==>
hoistway_travel + 2*hoistway_pit_depth + 120 = compensation_cable_length;

compensation_cable_quantity * compensationcable!compensation_cable_unit_weight *
(hoistway_pit_depth + hoistway_travel)  = compensation_cable_load_car_side_car_top;

compensation_cable_quantity * compensationcable!compensation_cable_unit_weight *
(counterweight_above_top_car_bottom + hoistway_travel)  = compensation_cable_load_counterweight_side_car_bottom;

compensation_cable_quantity * compensationcable!compensation_cable_unit_weight *
counterweight_above_top_car_bottom = compensation_cable_load_counterweight_side_car_top;

compensation_cable_quantity * compensationcable!compensation_cable_unit_weight *
hoistway_pit_depth = compensation_cable_load_car_side_car_bottom;

hoist_cable_quantity * hoistcable!hoist_cable_unit_weight - 0.25 *
controlcable!control_cable_unit_weight =compensation_cable_unit_weight_optimal;

(control_cable_load_car_side_car_top +control_cable_load_car_side_car_bottom)  / 2 =  control_cable_load_average;

controlcable!control_cable_unit_weight * control_cable_loop_below_car_bottom = control_cable_load_car_side_car_bottom;

controlcable!control_cable_unit_weight * (0.5 * hoistway_travel + control_cable_loop_below_car_bottom)  =
control_cable_load_car_side_car_top;

carbuffer!car_buffer_stroke + 78 = control_cable_loop_below_car_bottom;

opening_count =< 15 ==> controlcable!model_id := `control_cable_model_m01;

opening_count > 15, opening_count =< 25 ==>   controlcable!model_id := `control_cable_model_m02;

opening_count > 25 ==> controlcable!model_id := `control_cable_model_m03;

carbuffer!model_id = `model_oh1 ==> counterwtbuffer!model_id := `model_m01;

carbuffer!model_id = `model_om14 ==> counterwtbuffer!model_id := `model_m02;

counterweight_buffer_footing_channel_height + counterweight_buffer_blocking_height +
counterwtbuffer!counterweight_buffer_height + counterweight_runby = counterweight_above_top_car_bottom;

0.01 * hoistway_travel + 6 + hoistway_pit_depth - counterweight_buffer_footing_channel_height -
counterweight_buffer_blocking_height - counterwtbuffer!counterweight_buffer_height = counterweight_bottom_reference;

counterweight_weight / counterweight_buffer_quantity = counterweight_buffer_load;

compensation_cable_quantity > 0 ==> counterwtguard!model_id := `counterwtguard_thickness_m01;

compensation_cable_quantity = 0 ==> counterwtguard!model_id := `counterwtguard_thickness_m02;

counterwtguiderail!counterweight_weight_limit >= counterweight_weight;

hoistway_top_landing_to_underside_machine_beam - (deflector_sheave_protrusion +
counterweight_buffer_footing_channel_height  + counterweight_buffer_blocking_height +
counterwtbuffer!counterweight_buffer_height + counterweight_runby + counterweight_frame_height - hoistway_pit_depth)  =
counterweight_overtravel;

floor (counterweight_stack_weight_desired / counterweight_plate_weight)  = counterweight_plate_quantity;

0.2816 * counterweight_plate_thickness * (counterweight_plate_depth *
(counterweight_between_guiderails_value - 2)  -3.5 *  (counterweight_plate_depth - 5)  -
6* (counterweight_plate_depth-7) )  = counterweight_plate_weight;

counterweight_bottom_reference - hoistway_pit_depth -counterweight_buffer_footing_channel_height +
counterweight_buffer_blocking_height + counterwtbuffer!counterweight_buffer_height = counterweight_runby;

hoistway_depth - platform!platform_depth - platform_to_hoistway_front = counterweight_space;

counterweight_plate_quantity  * 1 = counterweight_stack_height;

counterweight_plate_quantity  * counterweight_plate_weight = counterweight_stack_weight;

counterweight_weight_desired - counterweight_system_weight = counterweight_stack_weight_desired;

6.2* (counterweight_between_guiderails_value - 1.75)  + 2*counterweight_frame_height + 70 =  counterweight_frame_weight;

counterweight_frame_weight + 4*7 = counterweight_system_weight;

counterweight_space - counterweight_plate_depth - counterweight_to_platform_rear =  counterweight_to_hoistway_rear;

0.5* (counterweight_space - counterweight_plate_depth -counterweight_u_bracket_protrusion)  =counterweight_to_platform_rear;

counterweight_stack_weight + counterweight_system_weight = counterweight_weight;

car_weight + control_cable_load_average + 0.4 * platform!car_capacity = counterweight_weight_desired;

safety_beam_load * safety_beam_between_guiderails = crosshead_bending_moment;

safety_beam_load * safety_beam_between_guiderails * safety_beam_between_guiderails = crosshead_deflection_index;

sling_crosshead (@sling!model_id,^crosshead!model_id) ;

deflectorsheave!deflector_sheave_protrusion = deflector_sheave_protrusion;

(hoistway_travel + hoistway_overhead + hoistway_pit_depth)  * 2 + 144 = governor_cable_length;

machine_sheave_height + machine_beam_support_bottom_to_machine_room_top +
deflector_sheave_protrusion + counterweight_overtravel = hoist_cable_above_counterweight_car_bottom;

hoistway_travel + hoistway_overhead + machine_sheave_height +
hoist_cable_above_counterweight_car_bottom + 120 - sling_underbeam = hoist_cable_length;
```

```
(hoist_cable_quantity * hoistcable!hoist_cable_unit_weight) * (hoistway_overhead +
hoistway_travel + 5 - sling_underbeam) = hoist_cable_load_car_side_car_bottom;

(hoist_cable_quantity * hoistcable!hoist_cable_unit_weight) * (hoistway_overhead + 5 - sling_underbeam) =
hoist_cable_load_car_side_car_top;

hoist_cable_quantity * hoistcable!hoist_cable_unit_weight *
hoist_cable_above_counterweight_car_bottom = hoist_cable_load_counterweight_side_car_bottom;

(hoist_cable_quantity * hoistcable!hoist_cable_unit_weight) * (hoistway_travel +
hoist_cable_above_counterweight_car_bottom) = hoist_cable_load_counterweight_side_car_top;

car_speed_cable_safety (@car_speed, ^hoist_cable_safety_factor_minimum);

sys:max_list ([@hoist_cable_traction_ratio_car_top_full, @hoist_cable_traction_ratio_car_bottom_empty,
   @hoist_cable_traction_ratio_car_bottom_full, @hoist_cable_traction_ratio_car_top_empty], ^hoist_cable_traction_ratio);

(cable_load_counterweight_side_car_bottom / (cable_load_car_side_car_bottom -
car_capacity_range)) * hoist_cable_traction_factor = hoist_cable_traction_ratio_car_bottom_empty;

(cable_load_car_side_car_bottom / cable_load_counterweight_side_car_bottom) *
hoist_cable_traction_factor = hoist_cable_traction_ratio_car_bottom_full;

(cable_load_counterweight_side_car_top / (cable_load_car_side_car_top - car_capacity_range)) *
hoist_cable_traction_factor = hoist_cable_traction_ratio_car_top_empty;

(cable_load_car_side_car_top / cable_load_counterweight_side_car_top) *
hoist_cable_traction_factor = hoist_cable_traction_ratio_car_top_full;

hoistway_overhead - machine_beam_support_bottom_to_machine_room_top =
hoistway_top_landing_to_underside_machine_beam;

90 + 57.3*asin((machine_sheave_to_deflector_sheave / machine_sheave_to_deflector_sheave_diagonal)) + 57.3*
asin(((machine!machine_sheave_diameter / 2 - deflector_sheave_diameter / 2) /
machine_sheave_to_deflector_sheave_diagonal)) = machine_angle_of_contact;

machine_beam_center_to_machine_sheave_center =< 0 ==>
machine_beam_reaction_front_left * machine_sheave_center_to_front_machine_beam_support =
machine_beam_bending_moment_left_maximum;

machine_beam_center_to_machine_sheave_center > 0 ==>
machine_beam_reaction_front_left*machine_beam_support_to_machine_beam_center +
machine_beam_center_to_machine_sheave_center * ( machine_beam_reaction_front_left-machine_beam_weight)
=machine_beam_bending_moment_left_maximum;

machine_beam_center_to_machine_sheave_center =< 0 ==>
machine_beam_reaction_front_right * machine_sheave_center_to_front_machine_beam_support =
machine_beam_bending_moment_right_maximum;

machine_beam_center_to_machine_sheave_center > 0 ==>
machine_beam_reaction_front_right * machine_beam_support_to_machine_beam_center +
machine_beam_center_to_machine_sheave_center * (machine_beam_reaction_front_right -
machine_beam_weight) = machine_beam_bending_moment_right_maximum;

machine_sheave_center_to_front_machine_beam_support - machine_beam_support_to_machine_beam_center =
machine_beam_center_to_machine_sheave_center;

machine_beam_impact_load_total * (machine!machine_right_offset /
(machine!machine_left_offset + machine!machine_right_offset)) = machine_beam_impact_load_left;

machine_beam_impact_load_total * (machine!machine_left_offset /
(machine!machine_left_offset + machine!machine_right_offset)) = machine_beam_impact_load_right;

machine_total_weight + deflectorsheave!deflector_sheave_weight + 2 * machine_suspended_load =
machine_beam_impact_load_total;

machine_beam_safety_factor * machine_beam_reaction_front_left = machine_beam_load_front_left;

machine_beam_reaction_front_right * machine_beam_safety_factor = machine_beam_load_front_right;

machine_beam_safety_factor * machine_beam_reaction_rear_left = machine_beam_load_rear_left;

machine_beam_reaction_rear_right * machine_beam_safety_factor = machine_beam_load_rear_right;

machine_beam_total_load_left - machine_beam_reaction_rear_left = machine_beam_reaction_front_left;

machine_beam_total_load_right - machine_beam_reaction_rear_right = machine_beam_reaction_front_right;

((machine_beam_weight * machine_beam_support_to_machine_beam_center) +
(machine_beam_impact_load_left *machine_sheave_center_to_front_machine_beam_support)) /
machine_beam_support_distance = machine_beam_reaction_rear_left;

((machine_beam_weight * machine_beam_support_to_machine_beam_center) +
(machine_beam_impact_load_right *machine_sheave_center_to_front_machine_beam_support)) /
machine_beam_support_distance = machine_beam_reaction_rear_right;

(machine_beam_bending_moment_left_maximum * machine_beam_safety_factor) / 16000 =
machine_beam_section_modulus_left_minimum;

(machine_beam_bending_moment_right_maximum * machine_beam_safety_factor) / 16000 =
machine_beam_section_modulus_right_minimum;

machine_beam_support_type = `steel ==>
machine_beam_support_bottom_to_machine_room_top = machine_beam_support_bottom_to_machine_room_top_spec;

machine_beam_support_distance/2 = machine_beam_support_to_machine_beam_center;

machine_beam_support_type = `pocket ==>
machine_beam_support_bottom_to_machine_room_top_spec -
```

```
machine_beam_bearing_plate_thickness = machine_beam_support_bottom_to_machine_room_top;
machine_beam_weight + machine_beam_impact_load_left = machine_beam_total_load_left;
machine_beam_weight + machine_beam_impact_load_right = machine_beam_total_load_right;
(machine_beam_length / 12) * mbeam!machine_beam_unit_weight = machine_beam_weight;
data_effi (@machine!model_id, @car_speed,^machine!machine_efficiency) ;
data_numeric (@machine!model_id,@car_speed,^machine!machine_gear_ratio_numeric) ;
hoistcable!hoist_cable_diameter * machine_groove_pressure_constant = machine_groove_pressure_acceptable_maximum;
cable_load_car_side_car_top / ((machine!machine_sheave_diameter / 2) *hoist_cable_quantity) = machine_groove_pressure;
pressure_cons (@car_speed, ^msheavegroove_model_id, ^machine_groove_pressure_constant) ,
^machine_groove_pressure_constant * @hoistcable!hoist_cable_diameter >= @machine_groove_pressure;
car_cable_hitch_to_platform_front + platform_to_hoistway_front + machine_beam_support_front_to_hoistway =
machine_sheave_center_to_front_machine_beam_support;
machine_sheave_height + (machine_beam_support_bottom_to_machine_room_top +
deflector_sheave_protrusion - 0.5 * deflector_sheave_diameter) = machine_sheave_to_deflector_sheave;
sqrt((machine_sheave_to_deflector_sheave_horizontal ** 2 +
machine_sheave_to_deflector_sheave ** 2)) = machine_sheave_to_deflector_sheave_diagonal;
car_cable_hitch_to_counterweight_cable_hitch - (machine!machine_sheave_diameter +
deflector_sheave_diameter) / 2 = machine_sheave_to_deflector_sheave_horizontal;
cable_load_car_side_car_top + cable_load_counterweight_side_car_top=machine_suspended_load;
machine!machine_weight + motor!motor_weight = machine_total_weight;
(car_capacity_range * (car_speed * (1 - 0.4))) /(33000 * machine!machine_efficiency * 0.95) = motor_horsepower_required;
(machine!machine_sheave_diameter / (24 * machine!machine_efficiency*machine!machine_gear_ratio_numeric) ) *
(2 * cable_load_unbalanced + (car_capacity_range * ((1 - 0.4) / 0.95) + (1.5 - 0.4))) =
motor_torque_releveling;
hoistway_width - (opening_width_door + opening_to_hoistway_left) = opening_to_hoistway_right;
motor_torque_releveling =< 40, motor!model_id = `motor_model_m01 ==>
motor_peak_current_required = 1.25 * motor_torque_releveling;
motor_torque_releveling > 40, motor_torque_releveling =< 62.5, motor!model_id = `motor_model_m01 ==>
motor_peak_current_required = 1.333 * motor_torque_releveling - 3.333;
motor_torque_releveling > 62.5, motor!model_id = `motor_model_m01 ==>
motor_peak_current_required = 1.6 * motor_torque_releveling - 20.0;
motor_torque_releveling =< 70, motor!model_id = `motor_model_m02 ==>
motor_peak_current_required = 1.286 * motor_torque_releveling;
motor_torque_releveling > 70, motor_torque_releveling =< 120, motor!model_id = `motor_model_m02 ==>
1.4 * motor_torque_releveling - 8 = motor_peak_current_required;
motor_torque_releveling > 120, motor!model_id = `motor_model_m02 ==>
1.6 * motor_torque_releveling + 60 = motor_peak_current_required;
motor_torque_releveling =< 83, motor!model_id = `motor_model_m03 ==>
1.205 * motor_torque_releveling = motor_peak_current_required;
motor_torque_releveling > 83, motor!model_id = `motor_model_m03 ==>
1.739 * motor_torque_releveling - 66.087 = motor_peak_current_required;
motor_torque_releveling =< 170, motor!model_id = `motor_model_m04 ==>
1.176 * motor_torque_releveling = motor_peak_current_required;
motor_torque_releveling > 170, motor_torque_releveling =< 230, motor!model_id = `motor_model_m04 ==>
1.25 * motor_torque_releveling - 12.5 = motor_peak_current_required;
motor_torque_releveling > 230, motor!model_id = `motor_model_m04 ==>
1.667 * motor_torque_releveling - 108.333 = motor_peak_current_required;
motor_torque_releveling =< 200, motor!model_id = `motor_model_m05 ==>
1.2 * motor_torque_releveling = motor_peak_current_required;
motor_torque_releveling > 200, motor!model_id = `motor_model_m05 ==>
1.455 * motor_torque_releveling - 50.909 = motor_peak_current_required;
motor_torque_releveling =< 270, motor!model_id = `motor_model_m06 ==>
1.296 * motor_torque_releveling = motor_peak_current_required;
motor_torque_releveling > 270, motor!model_id = `motor_model_m06 ==>
motor_torque_releveling + 80 = motor_peak_current_required;
platform!car_capacity =< 2500, platform!platform_depth =< 60, platform_width =< 84 ==>
platform!model_id := `platform_model_m01;
platform!car_capacity > 2500, platform!platform_depth =< 108, platform_width =< 128 ==>
platform!model_id := `platform_model_m02;
platform!car_capacity > 2500, platform!platform_depth =< 115, platform_width =< 126 ==>
platform!model_id := `platform_model_m02;
platform!car_capacity > 0, platform!platform_depth > 0,platform_width > 0, platform!model_id = `nil ==>
platform!model_id := `platform_model_m03;
door_opening_type = `center ==> platform!model_id := `platform_model_m01;
door!door_space + platform_running_clearance = platform_to_hoistway_front;
```

```
opening_to_hoistway_left - car_return_left = platform_to_hoistway_left;

opening_to_hoistway_right - car_return_right = platform_to_hoistway_right;

platform!model_id = `platform_model_m01 ==>
platform_weight_factor_s + 0.318 * (5.06 * platform_width + 0.11 * platform_width * platform!platform_depth +
(platform!platform_depth - 7.6)  * (3.14 + 0.8 * platform_weight_factor_ap))  = platform_weight ;

platform!model_id = `platform_model_m03 ==>
35 + platform_width * platform_weight_factor_x +3.228 * platform!platform_depth + 0.34 *opening_width_door +
platform_weight_factor_ap * (0.226 * platform!platform_depth - platform_weight_factor_z)  = platform_weight;

platform!model_id = `platform_model_m02 ==>
35 + platform_width *platform_weight_factor_x + 2.774 * platform!platform_depth + 0.03 *
platform_width * platform!platform_depth + 0.226 * opening_width_door +
platform_weight_factor_ap * 0.226 * (platform!platform_depth -platform_weight_factor_z)  = platform_weight;

platform_width =< 60, platform!model_id = `platform_model_m01 ==> platform_weight_factor_ap := 3;

platform_width > 60, platform_width =< 76,platform!model_id = `platform_model_m01 ==> platform_weight_factor_ap := 4;

platform_width > 76, platform!model_id = `platform_model_m01 ==> platform_weight_factor_ap := 5;

platform!model_id = `platform_model_m02 ==> floor((platform_width - 0.125) / 9)  =  platform_weight_factor_ap;

platform!model_id = `platform_model_m03 ==>factor_ap(Min,Max,^platform_weight_factor_ap),
@platform_width > Min,    @platform_width =< Max;

platform!car_capacity = 2000, platform!platform_depth =< 53,platform!model_id = `platform_model_m01 ==>
platform_weight_factor_s := 63;

platform!car_capacity ¥== 2000, platform!model_id = `platform_model_m01 ==> platform_weight_factor_s := 71;

platform!platform_depth > 53, platform!model_id = `platform_model_m01 ==> platform_weight_factor_s := 71;

platform_xz(@platform!model_id, @door_speed,   ^platform_weight_factor_x,^platform_weight_factor_z);

safety_beam_load * (safety_beam_between_guiderails - 8)  = safety_beam_bending_moment;

@safety_beam_bending_moment =< @safety!safety_beam_bending_moment_maximum;

platform_width + safety!safety_beam_constant = safety_beam_between_guiderails;

car_weight + platform!car_capacity + compensation_cable_load_car_side_car_top +
control_cable_load_car_side_car_top = safety_beam_load;

@safety_beam_load =< @safety!safety_beam_load_maximum;

platform_width =< 93 ==> safety!model_id := `safety_beam_model_m01;

platform_width > 93, platform_width =< 114 ==> safety!model_id := `safety_beam_model_m02;

platform_width > 114 ==> safety!model_id := `safety_beam_model_m03;

safety!safety_beam_a * platform_width + safety!safety_beam_b = safety_beam_weight;

sling_underbeam + (crosshead!crosshead_height +platform!platform_height - 0.81)  = sling_stile_length;

car_cab_height + sling_underbeam_space = sling_underbeam;

sling!sling_weight_factor_a * platform_width +sling!sling_weight_factor_b * sling_stile_length +
sling!sling_weight_factor_c = sling_weight;

6 + machine_beam_support_distance   = machine_beam_length;

local
car_speed_cable_safety(200,8.75);      car_speed_cable_safety(250,9.2);      car_speed_cable_safety(300,9.5);
car_speed_cable_safety(350,9.75);      car_speed_cable_safety(400,10.0);
data_effi(`machine_model_m01, 200, 0.78);              data_effi(`machine_model_m01, 250, 0.81);
data_effi(`machine_model_m01, 300, 0.84);              data_effi(`machine_model_m01, 350, 0.86);
data_effi(`machine_model_m01, 400, 0.87);              data_effi(`machine_model_m02, 200, 0.73);
data_effi(`machine_model_m02, 250, 0.76);              data_effi(`machine_model_m02, 300, 0.8);
data_effi(`machine_model_m02, 350, 0.81);              data_effi(`machine_model_m02, 400, 0.84);
data_effi(`machine_model_m03, 200, 0.73);              data_effi(`machine_model_m03, 250, 0.76);
data_effi(`machine_model_m03, 300, 0.8);               data_effi(`machine_model_m03, 350, 0.81);
data_effi(`machine_model_m03, 400, 0.84);              data_effi(`machine_model_m04, 200, 0.815);
data_effi(`machine_model_m04, 250, 0.842);            data_effi(`machine_model_m04, 300, 0.87);
data_effi(`machine_model_m04, 350, 0.88);             data_effi(`machine_model_m04, 400, 0.89);
data_numeric(`machine_model_m01, 200, 55);             data_numeric(`machine_model_m01, 250, 42.5);
data_numeric(`machine_model_m01, 300, 35.5);          data_numeric(`machine_model_m01, 350, 30.5);
data_numeric(`machine_model_m01, 400, 26.3);          data_numeric(`machine_model_m02, 200, 67.0);
data_numeric(`machine_model_m02, 250, 55);             data_numeric(`machine_model_m02, 300, 44.5);
data_numeric(`machine_model_m02, 350, 39.5);          data_numeric(`machine_model_m02, 400, 33.5);
data_numeric(`machine_model_m03, 200, 67);             data_numeric(`machine_model_m03, 250, 55);
data_numeric(`machine_model_m03, 300, 44.5);          data_numeric(`machine_model_m03, 350, 39.5);
data_numeric(`machine_model_m03, 400, 33.5);          data_numeric(`machine_model_m04, 200, 47.5);
data_numeric(`machine_model_m04, 250, 35.5);
data_numeric(`machine_model_m04, 300, 31.7);          data_numeric(`machine_model_m04, 350, 28.3);
data_numeric(`machine_model_m04, 400, 26.7);
pressure_cons(200, `machine_groove_model_m01, 196);      pressure_cons(250, `machine_groove_model_m01, 180);
pressure_cons(300, `machine_groove_model_m01, 164);      pressure_cons(350, `machine_groove_model_m01, 152);
pressure_cons(400, `machine_groove_model_m01, 142);      pressure_cons(200, `machine_groove_model_m02, 264);
pressure_cons(250, `machine_groove_model_m02, 238);      pressure_cons(300, `machine_groove_model_m02, 216);
pressure_cons(350, `machine_groove_model_m02, 202);      pressure_cons(400, `machine_groove_model_m02, 194);
sling_crosshead(`sling_model_m01, `crosshead_model_m04);   sling_crosshead(`sling_model_m02, `crosshead_model_m05);
sling_crosshead(`sling_model_m04, `crosshead_model_m03);   sling_crosshead(`sling_model_m03, `crosshead_model_m01);
sling_crosshead(`sling_model_m05, `crosshead_model_m02);
```

```
factor_ap(0,67,5);      factor_ap(67,76.5,6);      factor_ap(76.5,86,7);
factor_ap(86,96.5,8);   factor_ap(96.5,105,9);     factor_ap(105,1000,10);
platform_xz(`platform_model_m02,`single,2.26, 1.72);     platform_xz(`platform_model_m02,`double,2.354, 2.0);
platform_xz(`platform_model_m03,`single,2.441, 1.72);     platform_xz(`platform_model_m03,`double,2.394, 2.0);
platform_xz(_,`single,_, 1.72);     platform_xz(_,`double,_, 2.0);
end.

class governor_model
parameter
  model_id string [];
where
  data(model_id);
local
  data(governor_model_m01);
end.

class governor_cable_model
parameter
  governor_cable_diameter real [],
  model_id string [];
where
  data(^governor_cable_diameter, @model_id);
local
  data(0.375, governor_cable_model_m01);
end.

class hoist_cable_model
parameter
  hoist_cable_diameter real [],
  hoist_cable_quantity_f integer [],
  hoist_cable_ultimate_strength integer [],
  hoist_cable_unit_weight real [],
  model_id string [];
where
  data(^hoist_cable_diameter, ^hoist_cable_quantity_f, ^hoist_cable_ultimate_strength,  ^hoist_cable_unit_weight, @model_id);
local
  data(0.5,3,14500,0.03,hoist_cable_model_m01);
  data(0.625,3,23000,0.048,hoist_cable_model_m02);
  data(0.5,4,14500,0.03,hoist_cable_model_m03);
  data(0.625,4,23000,0.048,hoist_cable_model_m04);
  data(0.5,5,14500,0.03,hoist_cable_model_m05);
  data(0.625,5,23000,0.048,hoist_cable_model_m06);
  data(0.5,6,14500,0.03,hoist_cable_model_m07);
  data(0.625,6,23000,0.048,hoist_cable_model_m08);
end.

class car_intercom
parameter
  spec string[],
  model_id string [],
  car_intercom_weight integer [];
where
  data(@spec,^model_id, ^car_intercom_weight);
local
  data(no,car_intercom_m01,0);
  data(yes,car_intercom_m02,5);
end.

class car_lantern
parameter
  spec string [],
  model_id string [],
  car_lantern_weight integer [];
where
  data(@spec,^model_id, ^car_lantern_weight);
local
  data(no,car_lantern_m01,0);
  data(yes,car_lantern_m02,9);
end.

class machine_model
parameter
  machine_efficiency real [>=0.0,=<1.0],
  machine_gear_ratio_numeric real [],
  machine_left_offset real [],
  machine_right_offset real [],
  machine_sheave_diameter integer [],
  machine_sheave_height_f integer [],
  machine_suspended_load_maximum integer [],
  machine_weight integer [],
  model_id string [];
where
  data(^machine_left_offset, ^machine_right_offset, ^machine_sheave_diameter,
       ^machine_sheave_height_f, ^machine_suspended_load_maximum, ^machine_weight, @model_id);
```

```
local
  data(4.4375,9.4375,24,19,11500,1100,machine_model_m01);
  data(8,8,30,26,13200,1700,machine_model_m02);
  data(8,8,30,30,20000,2400,machine_model_m03);
  data(14.75,14.75,32,33,24600,2750,machine_model_m04);
end.

class machine_beam_model
parameter
  machine_beam_section_modulus real [],
  machine_beam_unit_weight real [],
  model_id string [];
where
  data(^machine_beam_section_modulus, ^machine_beam_unit_weight, @model_id);
local
  data(24.7,25.4,machine_beam_model_m01);
  data(29.4,35.0,machine_beam_model_m02);
  data(36.4,31.8,machine_beam_model_m03);
  data(38.2,35.0,machine_beam_model_m04);
  data(45.4,40.8,machine_beam_model_m05);
  data(50.8,50.0,machine_beam_model_m06);
  data(59.6,42.9,machine_beam_model_m07);
  data(64.8,50.0,machine_beam_model_m08);
  data(89.4,54.7,machine_beam_model_m09);
  data(103.0,70.0,machine_beam_model_m10);
end.

class motor_generator_model
parameter
  motor_peak_current_required real [],
  model_id string [],
  motor_generator_current_maximum integer [];
where
  data(^model_id, ^motor_generator_current_maximum),
  ^motor_generator_current_maximum >= @motor_peak_current_required;
local
  data(motor_generator_model_m01,143);
  data(motor_generator_model_m02,238);
  data(motor_generator_model_m03,340);
  data(motor_generator_model_m04,450);
end.

class motor
parameter
  motor_horsepower_required real [],
  model_id string [],
  motor_current_maximum integer [],
  motor_current_supported_maximum  integer [default([150])],
  motor_horsepower integer [],
  motor_weight integer [];
where
  data(^model_id, ^motor_current_maximum, ^motor_horsepower, ^motor_weight),
  @motor_horsepower_required =< ^motor_horsepower;
local
  data(motor_model_m01,150,10,374);
  data(motor_model_m02,250,15,473);
  data(motor_model_m03,260,20,539);
  data(motor_model_m04,340,25,615);
  data(motor_model_m05,440,30,715);
  data(motor_model_m06,530,40,990);
end.

class car_phone
parameter
  spec string [],
  model_id string [],
  car_phone_weight integer [];
where
  data(@spec,^model_id, ^car_phone_weight);
local
  data(no,car_phone_m01, 0);
  data(yes,car_phone_m02, 10);
end.

class platform_model
parameter
  car_capacity integer [],
  model_id string [],
  platform_depth  integer [],
  platform_height real [],
  platform_width_f integer[>=60,=<1000];
where
  data(@model_id, ^platform_height);
local
```

```
   data(platform_model_m01,6.625);
   data(platform_model_m02,6.625);
   data(platform_model_m03,6.6875);
end.

class car_position_indicator
parameter
   spec string [],
   model_id string [],
   car_position_indicator_weight integer [];
where
   data(@spec, ^model_id, ^car_position_indicator_weight);
local
   data(no,car_position_indicator_m01,0);
   data(yes,car_position_indicator_m02,12);
end.

class safety_beam_model
parameter
   safety_beam_bending_moment real [],
   safetu_beam_load_ real [],
   model_id string [],
   safety_beam_a real [],
   safety_beam_b integer [],
   safety_beam_bending_moment_maximum integer [],
   safety_beam_constant real [],
   safety_beam_height integer [],
   safety_beam_load_maximum integer [];
where
   data(@model_id, ^safety_beam_a, ^safety_beam_b, ^safety_beam_bending_moment_maximum,
        ^safety_beam_constant,  ^safety_beam_height, ^safety_beam_load_maximum);
local
   data(safety_beam_model_m01,1.69,1,682500,2.25,9,8000);
   data(safety_beam_model_m02,2.3,540,1233600,2.25,10,12000);
   data(safety_beam_model_m03,2.6,1035,1506000,2.625,13,20000);
end.

class sling_model
parameter
   crosshead_bending_moment real [],
   crosshead_deflection_index real [],
   model_id string [],
   sling_bending_moment_maximum integer [],
   sling_deflection_index_maximum integer [],
   sling_weight_factor_a real [],
   sling_weight_factor_b real [],
   sling_weight_factor_c integer [];
where
   data(^model_id, ^sling_bending_moment_maximum, ^sling_deflection_index_maximum,
        ^sling_weight_factor_a, ^sling_weight_factor_b, ^sling_weight_factor_c),
   @crosshead_bending_moment =< ^sling_bending_moment_maximum,
   @crosshead_deflection_index =< ^sling_deflection_index_maximum;
   data(@model_id, ^sling_bending_moment_maximum, ^sling_deflection_index_maximum,
        ^sling_weight_factor_a, ^sling_weight_factor_b, ^sling_weight_factor_c);
local
   data(sling_model_m01,705000,84600000,1.5,1.002,56);
   data(sling_model_m02,850000,103800000,1.75,1.002,94);
   data(sling_model_m04,810000,96900000,1.8,1.2,223);
   data(sling_model_m03,1340000,200700000,2.5,1.6,223);
   data(sling_model_m05,1790000,200700000,3.1,2.2,317);
end.
```

（２） 要求仕様入力

```
:set(#1,car_cab_height ,96).
:set(#1,car_capacity_range ,3000).
:set(#1,car_intercom_spec ,no).
:set(#1,car_lantern_spec ,no).
:set(#1,car_phone_spec ,yes).
:set(#1,car_position_indicator_spec ,yes).
:set(#1,door_opening_type ,side).
:set(#1,door_speed ,double).
:set(#1,hoistway_floor_height  ,165).
:set(#1,hoistway_depth ,110).
:set(#1,machine_beam_support_front_to_hoistway ,3).
:set(#1,hoistway_width ,90).
:set(#1,machine_beam_support_distance ,118).
:set(#1,machine_beam_support_type ,pocket).
:set(#1,machine_beam_support_bottom_to_machine_room_top_spec ,16).
:set(#1,opening_height ,84).
:set(#1,door_opening_strike_side ,right).
:set(#1,opening_width_door ,42).
```

```
:set(#1,opening_count ,6).
:set(#1,hoistway_overhead ,192).
:set(#1,hoistway_pit_depth ,72).
:set(#1!platform,platform_depth ,84).
:set(#1!platform,platform_width_f ,70).
:set(#1,car_speed ,250).
:set(#1,hoistway_travel ,729).
```

（３）値生成器

```
config(elevator,id1,[
   generator(opening_to_hoistway_left ,[31,35],1),
   part(car,data(_,_,_,_,_,_,model_id)),
   part(governor,data(model_id)),
   part(governorcable,data(_,model_id)),
   part(sling,data(model_id,_,_,_,_,_)),
   member(counterweight_between_guiderails_value,[28,38,54]),
   member(counterweight_buffer_quantity,[1,2]),
   generator(counterweight_plate_depth,[7.0,12.0],1.0),
   generator(car!car_supplement_weight,[0,500],100),
   part(mbeam,data(_,_,model_id)),
   part(machine,data(_,_,_,_,_,_,model_id)),
   part(hoistcable,data(_,_,_,_,model_id)),
   part(carguiderail,data(model_id,_)) ]).
```

（４）設計結果例

要求仕様に対して、４５８通りの設計解が存在するが、ここでは最初に求められた設計解を一つだけ示す。

```
 elevator#1
part
   car   car#2
   carbuffer   car_buffer_model#12
   carguiderail   car_guiderail_model#24
   compensationcable   compensation_cable_model#28
   controlcable   control_cable_model#34
   counterwtbuffer   counterweight_buffer_model#38
   counterwtguard   counterweight_guard_required#45
   counterwtguiderail   counterweight_guiderail_model#49
   crosshead   crosshead_model#55
   deflectorsheave   deflector_sheave_model#59
   door   door_model#66
   governor   governor_model#79
   governorcable   governor_cable_model#82
   hoistcable   hoist_cable_model#86
   intercom   car_intercom#93
   lantern   car_lantern#98
   machine   machine_model#103
   mbeam   machine_beam_model#114
   motgenerator   motor_generator_model#119
   motor   motor#124
   phone   car_phone#132
   platform   platform_model#137
   positionind   car_position_indicator#144
   safety   safety_beam_model#149
   sling   sling_model#160
parameter
   cable_load_car_side_car_bottom real#171 ^6416.8216133333335
   cable_load_car_side_car_top real#172 ^6539.074913333334
   cable_load_counterweight_side_car_bottom real#173 ^4676.137368
   cable_load_counterweight_side_car_top real#174 ^4614.755568
   cable_load_total_car_side_maximum real#175   6539.074913333334
   cable_load_unbalanced        real#176 ^-91.81754999999998
   carslingub                   real#177   nil
   car_buffer_blocking_height int#178   ^18     [>=6,=<10000]
   car_buffer_compressed_clearance real#179 ^42.25     [>=24.0,=<1000.0]
   car_buffer_load              real#180 ^6264.247213333334
   car_buffer_load_total      real#181 ^6264.247213333334
   car_buffer_quantity         int#182   ^1     [>=1,=<2]
   car_buffer_quantity_at_most int#183   ^1     [>=1,=<2]
   car_buffer_striking_speed_maximum real#184 ^287.5
   car_buffer_stroke_minimum real#185 ^4.278273809523809     [>=0.0,=<14.0]
   car_cable_hitch_to_counterweight_cable_hitch real#186 ^54.75
   car_cable_hitch_to_platform_front real#187 ^38.0
   car_cable_hitch_to_platform_rear real#188 ^46.0
   car_cab_height               real#189 @96.0     [>=84.0,=<240.0]
   car_cab_weight                real#190 ^1668.3333333333333
   car_capacity_range          int#191   @3000     [>=2000,=<4000]
   car_fixture_weight          real#192 ^22.0
   car_guiderail_vertical_force real#193 ^6527.074913333334
   car_guiderail_vertical_force_maximum real#194 ^8142.865
```

```
car_intercom_spec           str#195 @no     [member([yes,no])]
car_lantern_spec           str#196 @no     [member([yes,no])]
car_misc_weight              real#197 ^360.0
car_overtravel             real#198 ^68.125
car_phone_spec               str#199 @yes     [member([yes,no])]
car_position_indicator_spec str#200 @yes     [member([yes,no])]
car_return_left           real#201 ^25.0
car_return_right          real#202 ^3.0     [>=1]
car_runby                     int#203   ^6     [>=6,=<24]
car_speed                     int#204   @250     [member([200,250,300,350,400])]
car_supplement_weight       int#205   @0     [>=0,=<500]
car_weight                    real#206 ^3264.2472133333335
compensation_cable_length real#207 ^993.0
compensation_cable_load_car_side_car_bottom real#208 ^16.8624
compensation_cable_load_car_side_car_top real#209 ^187.5942
compensation_cable_load_counterweight_side_car_bottom real#210 ^180.343368
compensation_cable_load_counterweight_side_car_top real#211 ^9.611567999999998
compensation_cable_quantity int#31     ^2
compensation_cable_unit_weight_optimal real#213 ^0.10824999999999999
control_cable_load_average real#214 ^44.79775
control_cable_load_car_side_car_bottom real#215 ^14.362
control_cable_load_car_side_car_top real#216 ^75.2335
control_cable_loop_below_car_bottom real#217 ^86.0
counterweight_above_top_car_bottom real#218 ^41.03999999999999
counterweight_between_guiderails_value int#219   @28     [member([28,38,54])]
counterweight_bottom_reference real#220 ^55.53999999999999
counterweight_buffer_blocking_height int#221   @0     [>=0,=<120]
counterweight_buffer_footing_channel_height int#222   [@1]
counterweight_buffer_load real#223 ^4479.15
counterweight_buffer_quantity int#224   @1     [>=1,=<2]
counterweight_frame_height real#225 @138.0     [>=90.0,=<174.0]
counterweight_frame_thickness real#226 [@31.0]
counterweight_frame_weight real#227 ^508.75
counterweight_frame_weight_s real#228   nil
counterweight_overtravel real#229 ^54.83500000000001
counterweight_plate_depth real#230 @7.0     [>=7.0,=<12.0]
counterweight_plate_quantity int#231   ^80
counterweight_plate_thickness int#232   [@1]
counterweight_plate_weight real#233 ^49.28
counterweight_runby         real#234 ^11.289999999999992     [>=6.0,=<36.0]
counterweight_space         real#235 ^18.25
counterweight_stack_height real#236 ^80.0
counterweight_stack_weight real#237 ^3942.4
counterweight_stack_weight_desired real#238 ^3972.2949633333337
counterweight_system_weight real#239 ^536.75
counterweight_to_hoistway_rear real#240 ^6.0
counterweight_to_platform_rear real#241 ^5.25
counterweight_u_bracket_protrusion real#242 [@0.75]
counterweight_weight        real#243 ^4479.15     [=<56000]
counterweight_weight_desired real#244 ^4509.044963333334
crosshead_bending_moment real#245 ^471581.16248833336
crosshead_deflection_index real#246 ^34071738.98978209
deflector_sheave_diameter int#62     ^20
deflector_sheave_protrusion real#248 ^15.125
door_opening_strike_side str#249 @right     [member([left,right])]
door_opening_type            str#250 @side     [member([center,side])]
door_operator_header_weight real#251 ^63.0
door_operator_weight        real#252 ^198.0
door_space                    real#253 ^6.5
door_speed                    str#254 @double     [member([single,double])]
governor_cable_length       real#255 ^2130.0
hoistway_bracket_spacing real#258 @165.0
hoistway_depth                real#257 @110.0
hoistway_floor_height       real#258 @165.0
hoistway_overhead            real#259 @192.0     [>=0.0,=<360.0]
hoistway_pit_depth          real#260 @72.0     [>=0.0,=<240.0]
hoistway_top_landing_to_underside_machine_beam real#261 ^177.0
hoistway_travel              real#262 @729.0
hoistway_width                real#263 @90.0
hoist_cable_above_counterweight_car_bottom real#264 ^110.96000000000001
hoist_cable_length          real#265 ^1060.96
hoist_cable_load_car_side_car_bottom real#266 ^121.35
hoist_cable_load_car_side_car_top real#267 ^12.0
hoist_cable_load_counterweight_side_car_bottom real#268 ^16.644000000000002
hoist_cable_load_counterweight_side_car_top real#269 ^125.994
hoist_cable_quantity        int#270   ^5     [>=3,=<6]
hoist_cable_safety_factor real#271 ^11.08719520129227
hoist_cable_safety_factor_minimum real#272 ^9.2
hoist_cable_traction_factor real#273 [@1.304]
hoist_cable_traction_ratio real#274 ^1.8477584698342289
hoist_cable_traction_ratio_car_bottom_empty real#275 ^1.7846068123888124
hoist_cable_traction_ratio_car_bottom_full real#276 ^1.789411799800375
hoist_cable_traction_ratio_car_top_empty real#277 ^1.7003430015004086
hoist_cable_traction_ratio_car_top_full real#278 ^1.8477584698342289
```

```
   machine_acceleration_maximum real#279 [ 4.25]
   machine_angle_of_contact real#280 ^152.40980990239777
   machine_beam_bearing_plate_thickness int#281   [@1]
   machine_beam_bending_moment_left_maximum real#282 ^361109.3945705615
   machine_beam_bending_moment_right_maximum real#283 ^361109.3945705615
   machine_beam_center_to_machine_sheave_center real#284 ^-10.25
   machine_beam_impact_load_left real#285 ^12398.330481333334
   machine_beam_impact_load_right real#286 ^12398.330481333334
   machine_beam_impact_load_total real#287 ^24796.66096266667
   machine_beam_length        real#288 ^124.0
   machine_beam_load_front_left real#289 ^7777.740806135171
   machine_beam_load_front_right real#290 ^7777.740806135171
   machine_beam_load_rear_left real#291 ^5516.096199264831
   machine_beam_load_rear_right real#292 ^5516.096199264831
   machine_beam_reaction_front_left real#293 ^7407.37219631921
   machine_beam_reaction_front_right real#294 ^7407.37219631921
   machine_beam_reaction_rear_left real#295 ^5253.424951680791
   machine_beam_reaction_rear_right real#296 ^5253.424951680791
   machine_beam_safety_factor real#297 [@1.05]
   machine_beam_section_modulus_left_minimum real#298 ^23.6978040186931
   machine_beam_section_modulus_right_minimum real#299 ^23.6978040186931
   machine_beam_support_bottom_to_machine_room_top real#300 ^15.0
   machine_beam_support_bottom_to_machine_room_top_spec real#301 @16.0
   machine_beam_support_distance real#302 @118.0
   machine_beam_support_front_to_hoistway real#303 @3.0
   machine_beam_support_to_machine_beam_center real#304 ^59.0
   machine_beam_support_type str#305 @pocket    [member([pocket,steel])]
   machine_beam_total_load_left real#306 ^12660.797148000001
   machine_beam_total_load_right real#307 ^12660.797148000001
   machine_beam_weight        real#308 ^262.46666666666664
   machine_efficiency         real#105 ^0.76    [>=0.0,=<1.0]
   machine_gear_ratio_numeric real#106 ^55.0
   machine_groove_pressure  real#311 ^87.18766551111112
   machine_groove_pressure_acceptable_maximum real#312 ^90.0
   machine_groove_pressure_constant real#313 ^180.0
   machine_sheave_center_to_front_machine_beam_support real#314 ^48.75
   machine_sheave_height     int#110  ^26
   machine_sheave_to_deflector_sheave real#316 ^46.125
   machine_sheave_to_deflector_sheave_diagonal real#317 ^54.88695769488413
   machine_sheave_to_deflector_sheave_horizontal real#318 ^29.75
   msheavegroove_model_id   str#319 ^machine_groove_model_m01
   machine_suspended_load    real#320 ^11153.830481333334
   machine_total_weight      real#321 ^2239.0
   minacceptablembeammodulus real#322 @23.6978040186931
   motor_current_supported_maximum int#323  [ 150]
   motor_horsepower_required real#324 ^18.886930244270964    [>=0.0,=<40.0]
   motor_peak_current_required real#325 ^74.0031919030156
   motor_torque_releveling  real#326 ^61.41343726391336
   opening_count              real#327 @6.0
   opening_height             real#328 @84.0
   opening_to_hoistway_left int#329  @33
   opening_to_hoistway_right int#330  ^15
   opening_width_building   real#331 @42.0    [>=0.0,=<72.0]
   opening_width_door         real#331 @42.0    [>=0.0,=<72.0]
   platform_depth_spec       int#333  @84
   platform_running_clearance real#334 [@1.25]
   platform_to_hoistway_front real#335 ^7.75
   platform_to_hoistway_left int#336  ^8    [>=8]
   platform_to_hoistway_right int#337  ^12    [>=8]
   platform_weight            real#338 ^748.412
   platform_weight_factor_ap int#339  ^7    [>=3,=<10]
   platform_weight_factor_s int#340  ^nil    [member([63,71])]
   platform_weight_factor_x real#341 ^2.354
   platform_weight_factor_z real#342 ^2.0
   platform_width             int#343  @70    [>=60,=<1000]
   safety_beam_bending_moment real#344 ^419364.5631816667
   safety_beam_between_guiderails real#345 ^72.25
   safety_beam_bg              real#346  nil
   safety_beam_load           real#347 ^6527.074913333334
   safety_beam_weight         real#348 ^119.3
   sling_stile_length       real#349 ^130.94
   sling_underbeam            real#350 ^117.0
   sling_underbeam_space     real#351 @21    [>=12,=<1000]
   sling_weight               real#352 ^292.20187999999996
end

car#2
owner elevator#1-car
parameter
   car_cable_hanger_weight  int#4     ^30
   car_duct_and_wiring_weight int#5     ^45
   car_guideshoe_weight      int#6    ^18
   car_landing_switch_weight int#7    ^20
   car_limit_switch_cam_weight int#8     ^35
   car_maintenance_station_weight int#9      ^7
   car_supplement_weight      int#205  @0    [>=0,=<500]
   model_id                    str#11  @car1
end

car_buffer_model#12
owner
   elevator#1-carbuffer
parameter
   car_buffer_quantity_at_most int#183  ^1    [>=1,=<2]
   car_buffer_load_total      real#181 ^6264.247213333334
   car_buffer_stroke_minimum real#185 ^4.278273809523809     [>=0.0,=<14.0]
   car_buffer_footing_channel_height real#17  ^3.5
   car_buffer_height          real#18  ^28.75
   car_buffer_load_maximum   int#19    ^11000
   car_buffer_load_minimum   int#20    ^2900
   car_buffer_stroke          int#21   ^8
   car_buffer_quantity        int#182  ^1    [>=1,=<2]
   model_id                    str#23  ^model_oh1
end

car_guiderail_model#24
owner elevator#1-carguiderail
parameter
   model_id                    str#26  @car_guiderail_model_m02
   car_guiderail_weight       int#27   ^11
end

compensation_cable_model#28
owner elevator#1-compensationcable
parameter
   compensation_cable_unit_weight_optimal real#213 ^0.10824999999999999
   compensation_cable_quantity_c02 int#31    ^2
   compensation_cable_unit_weight real#32  ^0.1171
   model_id                    str#33  ^compensation_cable_model_m05
end

control_cable_model#34
owner elevator#1-controlcable
parameter
   control_cable_unit_weight real#36  ^0.167
   model_id                    str#37  ^control_cable_model_m01
end

counterweight_buffer_model#38
owner elevator#1-counterwtbuffer
parameter
   counterweight_buffer_height real#40  ^28.75
   counterweight_buffer_load_maximum int#41    ^11000
   counterweight_buffer_load_minimum int#42    ^2900
   counterweight_buffer_stroke real#43  ^8.25
   model_id                    str#44  ^model_m01
end

counterweight_guard_required#45
owner elevator#1-counterwtguard
parameter
   counterweight_guard_thickness int#47    ^0
   model_id                    str#48  ^counterwtguard_thickness_m01
end

counterweight_guiderail_model#49
owner elevator#1-counterwtguiderail
parameter
   counterweight_weight       real#243 ^4479.15    [=<56000]
   counterweight_guiderail_unit_weight int#52    ^8
   counterweight_weight_limit int#53    ^15000
   model_id                    str#54  ^counterweight_guiderail_model_m01
end

crosshead_model#55
owner elevator#1-crosshead
parameter
   crosshead_height           real#57  ^8.125
   model_id                    str#58  ^crosshead_model_m04
end

deflector_sheave_model#59
owner elevator#1-deflectorsheave
parameter
   hoist_cable_diameter      real#88  ^0.5
   deflector_sheave_diameter_s int#62    ^20
   deflector_sheave_protrusion real#63  ^15.125
   deflector_sheave_weight   int#64    ^250
   model_id                   str#65  ^deflector_sheave_model_m01
end

door_model#66
owner elevator#1-door
parameter
   car_saf_t_edge_weight      int#68    ^7
   door_opening_strike_side str#249 @right    [member([left,right])]
   door_opening_type          str#250 @side    [member([center,side])]
   door_operator_constant    real#71   ^1.5
   door_operator_engine_weight int#72    [^135]
   door_operator_header_weight real#251 ^63.0
   door_operator_weight       real#252 ^198.0
   door_space                  real#253 ^6.5
   opening_width_door         real#331 @42.0    [>=0.0,=<72.0]
   door_speed                  str#254 @double    [member([single,double])]
   model_id                    str#78   ^door_model_code_m03
end

governor_model#79
owner elevator#1-governor
parameter
   model_id                   str#81    governor_model_m01
end

governor_cable_model#82
owner elevator#1-governorcable
parameter
   governor_cable_diameter   real#84  ^0.375
   model_id                   str#85  @governor_cable_model_m01
end

hoist_cable_model#86
owner elevator#1-hoistcable
parameter
   hoist_cable_diameter      real#88  ^0.5
   hoist_cable_quantity_f    int#270   ^5    [>=3,=<6]
   hoist_cable_ultimate_strength int#90    ^14500
   hoist_cable_unit_weight  real#91   ^0.03
   model_id                   str#92   @hoist_cable_model_m05
end

car_intercom#93
owner elevator#1-intercom
parameter
   spec                        str#195 @no     [member([yes,no])]
   model_id                    str#96   ^car_intercom_m01
   car_intercom_weight       int#97     ^0
end

car_lantern#98
owner elevator#1-lantern
parameter
   spec                        str#196 @no     [member([yes,no])]
   model_id                    str#101 ^car_lantern_m01
   car_lantern_weight        int#102   ^0
end

machine_model#103
owner elevator#1-machine
parameter
   machine_efficiency         real#105 ^0.76     [>=0.0,=<1.0]
   machine_gear_ratio_numeric real#106 ^55.0
   machine_left_offset        real#107 ^8.0
   machine_right_offset       real#108 ^8.0
   machine_sheave_diameter   int#109   ^30
   machine_sheave_height_f   int#110   ^26
   machine_suspended_load_maximum int#111   ^13200
   machine_weight              int#112   ^1700
   model_id                   str#113 @machine_model_m02
end

machine_beam_model#114
owner elevator#1-mbeam
parameter
   machine_beam_section_modulus real#116 ^24.7
   machine_beam_unit_weight real#117 ^25.4
   model_id                   str#118 @machine_beam_model_m01
end

motor_generator_model#119
owner elevator#1-motgenerator
parameter
   motor_peak_current_required real#121 @nil
   model_id                       str#122 ^nil
   motor_generator_current_maximum int#123   ^nil
end

motor#124
owner elevator#1-motor
parameter
   motor_horsepower_required real#324 ^18.886930244270964     [>=0.0,=<40.0]
   model_id                       str#127 ^motor_model_m03
   motor_current_maximum      int#128   ^260
   motor_current_supported_maximum int#129   [ 150]
   motor_horsepower               int#130   ^20
   motor_weight                   int#131   ^539
end

car_phone#132
owner elevator#1-phone
parameter
   spec   ·                       str#199 @yes     [member([yes,no])]
   model_id                       str#135 ^car_phone_m02
   car_phone_weight              int#136   ^10
end

platform_model#137
owner elevator#1-platform
parameter
   car_capacity               int#191   @3000      [>=2000,=<4000]
   model_id                    str#140 ^platform_model_m02
   platform_depth             int#333   @84
   platform_height            real#142 ^6.625
   platform_width_f           int#343   @70     [>=60,=<1000]
end

car_position_indicator#144
owner elevator#1-positionind
parameter
   spec                           str#200 @yes     [member([yes,no])]
   model_id                       str#147 ^car_position_indicator_m02
   car_position_indicator_weight int#148   ^12
end

safety_beam_model#149
owner elevator#1-safety
parameter
   safety_beam_bending_moment real#151   nil
   safetu_beam_load_           real#152   nil
   model_id                     str#153 ^safety_beam_model_m01
   safety_beam_a               real#154 ^1.69
   safety_beam_b               int#155   ^1
   safety_beam_bending_moment_maximum int#156   ^682500
   safety_beam_constant       real#157 ^2.25
   safety_beam_height          int#158   ^9
   safety_beam_load_maximum int#159   ^8000
end

sling_model#160
owner   elevator#1-sling
parameter
   crosshead_bending_moment real#245 ^471581.16248833336
   crosshead_deflection_index real#246 ^34071738.98978209
   model_id                     str#164 ^sling_model_m01
   sling_bending_moment_maximum int#165   ^705000
   sling_deflection_index_maximum int#166   ^84600000
   sling_weight_factor_a      real#167 ^1.5
   sling_weight_factor_b      real#168 ^1.002
   sling_weight_factor_c      int#169   ^56
end
```