Østr.prb

```
/* weight suspended by two strings */

  problem(0str,'Weight problem.\n%r=\n\n',[====,4]).

  cue linesys(string,string1,[top1,bottom1]).
  cue linesys(string,string2,[top2,bottom2]).

  isa(period,now).
  isa(particle,weight1).
  mass(weight1,mass1,now).
  incline(string1,180,bottom1).
  incline(string2,0,bottom2).
  fixed_contact(weight1,bottom1,now).
  fixed_contact(weight1,bottom2,now).
  tension(string1,t1,now).
  tension(string2,t2,now).

  given(mass1).
  sought(t1).
  sought(t2).

  :- assert( minimal_part(string,Str,Pt,Str) ),
     assert( static(_,_) ).
```

2spr.prb

```
/* Problem 4, p. 153 "Applied Mathematics"
   Two springs of different elasticities joined together.
      Alan Borning  September 24, 1979  */

  problem('2spr.prb','Two springs joined together.\n%r=\n\n',[====,4]).

  cue linesys(string,string1,[leftpt,mid1]).
  cue linesys(string,string2,[mid2,rightpt]).

  isa(period,now).
  fixed_contact(mid1,mid2,now).

  elastic(string1,true).
  elasticity(string1,elast1,now).
  natlength(string1,natlen1,now).

  elastic(string2,true).
  elasticity(string2,elast2,now).
  natlength(string2,natlen2,now).
  incline(string1,180,mid1).
  incline(string2,180,rightpt).
  separation(leftpt,rightpt,dist,0,now).

  tension(string1,t1,now).


  given(elast1).
  given(natlen1).
  given(elast2).
  given(natlen2).
  given(dist).
  sought(t1).

/* Kludges to get around inadequacies of "separation" stuff */
  varlength(string1,len1,now).
  varlength(string2,len2,now).
  equation(dist=len1+len2).
  sought(len1).
  sought(len2).


  :- assert( minimal_part(string,Str,Pt,Str) ),
     assert( static(_,_) ),
     /* Kludge for "incline" fix to tell it about current period */
     assert( current_period(now) ).
```

2wght.prb

```
/* two weights suspended by strings */

   problem(2wght,'Two Weights problem.\n%r=\n\n',[====,4]).

   cue linesys(string,string1,[top1,bottom1]).
   cue linesys(string,string2,[top2,bottom2]).

   isa(period,now).
   isa(particle,weight1).
   isa(particle,weight2).
   mass(weight1,mass1,now).
   mass(weight2,mass2,now).
   incline(string1,90,bottom1).
   incline(string2,90,bottom2).
   fixed_contact(weight1,bottom1,now).
   fixed_contact(weight2,bottom2,now).
   fixed_contact(bottom1,top2,now).
   tension(string1,t1,now).
   tension(string2,t2,now).

   given(mass1).
   given(mass2).
   sought(t1).
   sought(t2).

   :- assert( minimal_part(string,Str,Pt,Str) ),
      assert( static(_,_) ),
      /* Kludge for "incline" fix to tell it about current period */
      assert( current_period(now) ),
      consult('concav.fix').
```

30str.prb

```
/* weight suspended by two strings */

   problem(30str,'Weight problem.\n%r=\n\n',[====,4]).

   cue linesys(string,string1,[top1,bottom1]).
   cue linesys(string,string2,[top2,bottom2]).

   isa(period,now).
   isa(particle,weight1).
   mass(weight1,mass1,now).
   incline(string1,120,bottom1).
   incline(string2,60,bottom2).
   fixed_contact(weight1,bottom1,now).
   fixed_contact(weight1,bottom2,now).
   tension(string1,t1,now).
   tension(string2,t2,now).

   given(mass1).
   sought(t1).
   sought(t2).

   :- assert( minimal_part(string,Str,Pt,Str) ),
      assert( static(_,_) ).
```

45str.prb

```
/* weight suspended by two strings */

  problem(45str,'Weight problem.\n%r=\n\n',[====,4]).

  cue linesys(string,string1,[top1,bottom1]).
  cue linesys(string,string2,[top2,bottom2]).

  isa(period,now).
  isa(particle,weight1).
  mass(weight1,mass1,now).
  incline(string1,135,bottom1).
  incline(string2,45,bottom2).
  fixed_contact(weight1,bottom1,now).
  fixed_contact(weight1,bottom2,now).
  tension(string1,t1,now).
  tension(string2,t2,now).

  given(mass1).
  sought(t1).
  sought(t2).

  :- assert( minimal_part(string,Str,Pt,Str) ),
     assert( static(_,_) ).
```

age.pl

```
/* Information for the AGE problems */

% Predicates

argstruct(birth,2,
                [person,moment],
                [arg,val]).


argstruct(dateof,2,
                [moment,duration],
                [arg,val]).

argstruct(period,3,
                [period,moment,moment],
                [arg,val,val]).

argstruct(partition,2,
                [period,list(period)],
                [arg,arg]).



% Formulae

relates(daterule,[duration]).


preference(daterule,1).


prepare(daterule,Q,duration,duration,[Period,D],situation(Period)).


isform(daterule,situation(Period), D = D2 - D1 )
     :- cc period(Period,Start,Finish),
        cc duration(Period,D),
        cc dateof(Start,D1),
        cc dateof(Finish,D2).



relates(timesum,[duration]).


preference(timesum,2).


prepare(timesum,Q,duration,duration,[Period,D],situation(Period))
     :- cc partition(Period,Subperiods).

prepare(timesum,Q,duration,duration,[Period,D],situation(Superperiod))
     :- cc partition(Superperiod,Subperiods),
           memberchk(Period,Subperiods).


isform(timesum,situation(Period), D = Dsum)
     :- cc partition(Period,Subperiods),
        cc duration(Period,D),
```

```
          sumdurs(Subperiods,Dsum).


% Inference rules

partition(Period,[A,B])
     :- cc period(Period,M1,M3),
        cc period(A,M1,M2),
        cc period(B,M2,M3).
```

age1.prb

```
/* AGE1 */

problem(age1,'\tAGE1 - first age problem\n\t%r\n\n',[----,6]).

isa(moment,moment1).
isa(moment,moment2).
isa(duration,duration1).
dateof(moment2,duration1).
measure(duration1,1960,years).
birth(John,moment2).
isa(moment,moment3).
isa(duration,duration2).
dateof(moment3,duration2).
measure(duration2,1980,years).
isa(period,period1).
period(period1,moment2,moment3).
isa(duration,duration3).
duration(period1,duration3).
sought(duration3).
given(duration1).
 iven(duration2).
```

age2.prb

```
/* AGE2 */

problem(age2,'\tAGE2 - second age problem\n\t%r-\n\n',[----,6]).

isa(moment,moment1).
isa(moment,moment2).
birth(jack,moment2).
isa(period,period1).
period(period1,moment2,moment1).
isa(duration,duration1).
duration(period1,duration1).
measure(duration1,10,years).
isa(moment,moment3).
birth(jill,moment3).
isa(duration,d1).
measure(d1,2,years).
isa(period,period2).
period(period2,moment3,moment1).
isa(duration,duration2).
duration(period2,duration2).
 quation(duration2=duration1-d1).
sought(duration2).
given(duration1).
given(d1).
```

```
/* AGE3 problem.  Hand generated assertions */          age3.prb

problem(age3,'\tAGE3 - third age problem\n\t%r\n\n',[----,6]).

isa(moment,b1).
isa(moment,b2).
isa(moment,mom1).
birth(Jack,b1).
birth(Jill,b2).
period(period1,b1,mom1).
period(period2,b2,mom1).
duration(period1,age1).
duration(period2,age2).
equation(age1 = 4*age2).

isa(moment,mom2).
period(per1,mom1,mom2).
period(period11,b1,mom2).
period(period22,b2,mom2).
duration(per1,dur1).
duration(period11,age11).
duration(period22,age22).
measure(dur1,5,years).
equation(age11 = 3*age22).

given(dur1).
sought(age1).
sought(age2).
```

```
/* Problem 4, p. 153 "Applied Mathematics"
   Two springs of different elasticities joined together.
     Alan Borning  September 24, 1979  */
```

6ad2sp.nrb

```
problem('2spr.prb','Two springs joined together.\n%r=\n\n',[====,4]).

cue linesys(string,string1,[leftpt,mid1]).
cue linesys(string,string2,[mid2,rightpt]).

isa(period,now).
fixed_contact(mid1,mid2,now).

elastic(string1,true).
elasticity(string1,elast1,now).
natlength(string1,natlen1,now).

elastic(string2,true).
elasticity(string2,elast2,now).
natlength(string2,natlen2,now).
incline(string1,180,mid1).
incline(string2,180,rightpt).
separation(leftpt,rightpt,dist,0,now).

tension(string1,t1,now).


given(elast1).
given(natlen1).
given(elast2).
given(natlen2).
given(dist).
sought(t1).


:- assert( minimal_part(string,Str,Pt,Str) ),
   assert( static(_,_) ),
   /* Kludge for "incline" fix to tell it about current period */
   assert( current_period(now) ).
```

badmid.nrb

```
/* Problem 2, p. 151 "Applied Mathematics"
   Weight suspended from the middle of an elastic string
      Alan Borning  September 20, 1979  */

/* In this version, the length of the extended string has been given
   to the program as a specific equation.  Actually, the program ought
   to calculate the length using some geometric knowledge. */


  problem('midwgt.prb','Weight in middle problem.\n%r=\n\n',[====,4]).

  cue linesys(string,string1,[leftpt,midpt,rightpt]).

  isa(period,now).
  isa(particle,weight1).
  mass(weight1,m1,now).
  fixed_contact(weight1,midpt,now).

  elastic(string1,true).
  elasticity(string1,elast1,now).
  separation(leftpt,midpt,dist1,150,now).
  separation(midpt,rightpt,dist1,210,now).

/* tell the program what the stretched length is */
  equation( dist1*cos(30)=natlen/2 ).


  given(elast1).
  given(natlen).
  sought(m1).

  :- assert( static(_,_) ),
     /* Kludge for 'incline' fix to tell it about current period */
     assert( current_period(now) ).
```

barge

[A barge whose deck is 10ft
below the level of a dock is being drawn
in by means of a cable attached to the deck and passing
through a ring on the dock.
When the barge is 24ft from and approaching the dock
at 34ft/sec, how fast is the cable
being pulled in?]

```
                                   o_________________
                                   |
          34ft/sec ->              |10ft
         ____________   24ft       |
         \          /              |
         ~~~~~~~~~~~~~~~~~~~~~
```

@end(example)
@caption(The barge and the dock)
@tag(barge)

bench.prb

```
isa(bench,bench1).
isa(length,length1).
constlength(bench1,length1).
measure(length1,55,ft).
isa(moment,moment1).
isa(length,length2).
seat_separation(dignitary,length2).
measure(length2,4,ft).
packed(bench1,dignitary).
isa(number,number1).
number_in(dignitary,bench1,number1).
sought(number1).
given(length1).
given(length2).
```

bench.pl

```
/* John's dignatory problem */

% Predicates

argstruct(seat_separation,2,
                [object,length],
                [arg,arg]).

argstruct(packed,2,
                [line,object],
                [arg,arg]).

argstruct(number_in,3,
                [object,line,number],
                [arg,arg,arg]).

argstruct(body_width,2,
                [object,length],
                [arg,arg]).


  Formulae

relates(linefill,[number]).


prepare(linefill,Q,number,number_in,[Object,Line,N],situation(Object,Line)).


isform(linefill,situation(Object,Line), L = N*W + (N-1)*D )
     :- cc constlength(Line,L),
        cc number_in(Object,Line,N),
        cc body_width(Object,W),
        cc seat_separation(Object,D).


% Inference rules


body_width(Object,W)
     :- cc seat_separation(Object,D),
        cc measure(D,Dm,U),
           bw_check(Dm,U,W).


bw_check(Dm,ft,0) :- Dm >= 2.

bw_check(Dm,ft,2) :- Dm < 2.
```

ladder

```
/* Ladder Problem */

problem(ladder,'\tThe Ladder Problem\n\t%r\n\n',[---,6]).

isa(period,now).
isa(particle,man).

cue linesys(rod,ladder,[endpt1,pt1,cog,endpt2]).
c_of_gravity(ladder,cog).

cue linesys(path,floor,[endpt3,pt2/moving,endpt4]).
cue linesys(path,wall,[endpt5,pt3/moving,endpt6]).

concavity(ladder,stline).                    tangent(ladder,125).
concavity(floor,stline).                     tangent(floor,180).
concavity(wall,stline).                      tangent(wall,90).

fixed_contact(endpt1,pt3,now).
fixed_contact(endpt2,pt2,now).
fixed_contact(man,pt1,now).
 ixed_contact(endpt3,endpt6,now).

separation(endpt2,endpt1,d1,125,now).
separation(endpt3,pt2,d2,0,now).
separation(endpt2,pt1,d3,125,now).

mass(man,m2,now).
mass(ladder,m1,now).
solid(floor).              coeff(floor,mu).
solid(wall).               coeff(wall,0).

wh_side(endpt1,wall,left,now).
wh_side(endpt2,floor,left,now).

given(d1).
given(d2).
given(m1).
given(m2).
given(mu).

sought(d3).

   :- assert( static(_,_) ).
```

```
/* Jill Larkin's Problem */
```

larkin.prb

```
  problem(larkin,'\tLarkin''s statics problem\n\t%r\n\n',[----,6]).


  isa(period,now).
  isa(particle,a).
  isa(particle,b).

cue linesys(path,table,[lend,pt/moving,rend]).
cue linesys(string,string1,[pt1,pt2]).
cue linesys(string,string2,[pt3,pt4]).
cue linesys(string,string3,[pt5,pt6]).


  concavity(table,stline).        tangent(table,180).
  concavity(string1,stline).      tangent(string1,180).
  concavity(string2,stline).      tangent(string2,225).
  concavity(string3,stline).      tangent(string3,90).


  fixed_contact(b,pt,now).
  fixed_contact(b,pt1,now).
  fixed_contact(pt2,pt3,now).
  fixed_contact(pt2,pt5,now).
  fixed_contact(pt6,a,now).

  fixed(pt4,now).
  static(a,now).
  static(b,now).
  wh_side(b,table,left,now).

  mass(a,ma,now).
  mass(b,mb,now).
  coeff(table,mu).
  solid(table).

  given(mb).
  given(mu).
  sought(ma).


  :- assert( minimal_part(string,Str,Pt,Str) ),
     assert( static(_,_) ).
```

midwgt.prb (21)

```
/* Equations produced from problem file: midwst.prb

     tension1*cos(-30)+(tension2*cos(-150)+(mass1*g*cos(-270)+0))=0
     tension1*cos(180)+(tension2*cos(60)+(mass1*g*cos(-60)+0))=0
     tension2=elast1*(extension1/natlen1)
     curlen1=natlen1+extension1
     curlen1*cos(30)=natlen1

    Unknowns
     [mass1,tension1,tension2,extension1,curlen1]
 */

so :- simsolve(
&(=(+(*(tension1,cos(-30)),+(*(tension2,cos(-150)),+(*(*(mass1,g),cos(-270)),0))
),+(*(*(mass1,g),cos(-60)),0))),0),&(=(tension2,*(elast1,/(extension1,natlen1))
s(30)),natlen1),true)))))
   , [mass1,tension1,tension2,extension1,curlen1] ).
```

```
/* PULLEY PROBLEM WITH P1 ON TABLE AND P2 HANGING OVER EDGE. */
```

pr5

```
steady :-      problem,

               checklist( dbentry,[ isa(period,period1),
                                    isa(particle,p1),
                                    isa(particle,p2),
                                    mass(p1,bq,period1),
                                    mass(p2,cq,period1),
                                    accel(p1,a1,0,period1),
                                    incline(table,0,lefttop),
                                    solid(table),
                                    rough(table),
                                    coeff(table,mu),

                                    quantity(bq,b,arbs),
                                    quantity(cq,c,arbs)
                                  ]  ),

               checklist( asserta,[ given(bq),
                                    given(cq),
                                    given(mu),
                                    sought(a1)
                                  ]  ),

               nl,

               cue(pullsys_maj(sys,pull,str,p1,0,p2,270,periodi)),
               cue(pathsys(table,lefttop,edge,hor,stline)),
               cue(line_motion(p1,table,period1)).

problem :- writef('\n\tPULLEY ON TABLE PROBLEM.\n\t%r\n\n',[-,24]).
```

```
/* PULLEY problem  -  for MECHO 10.1 */

  problem(pulley,'Simple pulley problem.\n%r==\n\n',[=====,4]).

  isa(period,period0).
  isa(particle,particle00).
  isa(particle,particle01).
  mass(particle00,mass1,period0).
  mass(particle01,mass2,period0).
  accel(particle00,acceleration0,90,period0).

  measure(mass1,10,arbs).
  measure(mass2,12,arbs).

  given(mass1).
  given(mass2).
  sought(acceleration0).

  cue pullsys_stan(sys,pulley0,string0,particle00,particle01,period0).
```

pulley.prb

scaff.pro

```
/*   Scaffold problem   */

problem(scaff,'Scaffold problem.\n%r=\n\n',[====,4]).


cue linesys(rod,scaffold,[lend,point1,point2,rend]).
cue linesys(string,string1,[top1,bottom1]).
cue linesys(string,string2,[top2,bottom2]).

isa(period,now).
isa(particle,man1).
isa(particle,man2).
mass(scaffold,zero,now).
mass(man1,mq1,now).
mass(man2,mq2,now).
incline(string1,90,bottom1).
incline(string2,90,bottom2).
slope(scaffold,hor).
concavity(scaffold,stline).
isa(length,dq1).
isa(length,dq2).
isa(length,dq3).
separation(lend,rend,dq1,0,now).
separation(lend,point1,dq2,0,now).
separation(point2,rend,dq3,0,now).
fixed_contact(lend,bottom1,now).
fixed_contact(rend,bottom2,now).
fixed_contact(man1,point1,now).
fixed_contact(man2,point2,now).
tension(string1,t1,now).
tension(string2,t2,now).

given(mq1).
given(mq2).
given(dq1).
given(dq2).
given(dq3).
sought(t1).
sought(t2).


:- assert( minimal_part(string,Str,Pt,Str) ),
   assert( static(_,_) ).
```

scaff2.prb

```
/*   Modified Scaffold problem   */


  problem(scaff2,'Scaffold problem.\n%r=\n\n',[====,4]).


  cue linesys(rod,scaffold,[lend,point1,rend]).
  cue linesys(string,string1,[top1,bottom1]).
  cue linesys(string,string2,[top2,bottom2]).

  isa(period,now).
  isa(particle,man1).
  mass(scaffold,zero,now).
  mass(man1,mq1,now).
  incline(string1,90,bottom1).
  incline(string2,90,bottom2).
  slope(scaffold,hor).
  concavity(scaffold,stline).
  separation(lend,rend,20,0,now).
  separation(lend,point1,10,0,now).
  fixed_contact(lend,bottom1,now).
  fixed_contact(rend,bottom2,now).
  fixed_contact(man1,point1,now).
  tension(string1,t1,now).
  tension(string2,t2,now).

  given(mq1).
  sought(t1).
  sought(t2).


  :- assert( minimal_part(string,Str,Pt,Str) ),
     assert( static(_,_) ).
```

```
/* simple spring problem -- find the tension in a stretched spring */
```

Spring.nrb

```
problem('spring.prb','Spring problem.\n%r=\n\n',[====,4]).

cue linesys(string,spring1,[top1,bottom1]).

isa(period,now).
elastic(spring1,true).
elasticity(spring1,elast,now).
varlength(spring1,len,now).
extension(spring1,extens,now).
natlength(spring1,natL,now).
tension(spring1,t1,now).

given(elast).
given(len).
given(extens).
given(natL).
sought(t1).


:- assert( minimal_part(string,Str,Pt,Str) ),
   assert( static(_,_) ).
```

```
/* Strut Problem */                                          Strut.prb

problem(strut,'\tThe Strut Problem\n\t------------------\n\n',[]).

isa(period,now).
isa(particle,a).

cue linesys(rod,strut,[lend,rend]).
cue linesys(string,string1,[top1,bottom1]).
cue linesys(string,string2,[top2,bottom2]).

concavity(strut,stline).          tangent(strut,180).
concavity(string1,stline).        tangent(string1,140).
concavity(string2,stline).        tangent(string2,90).

fixed_contact(rend,bottom1,now).
fixed_contact(rend,top2,now).
fixed_contact(a,bottom2,now).

fixed(lend,now).
 ixed(top1,now).

tension(string1,t1,now).

mass(a,ma,now).

given(ma).

sought(t1).


   :- assert( minimal_part(string,Str,Pt,Str) ),
      assert( static(_,_) ).
```

tower

```
/*TOWER*/
/*TOWER PROBLEM P21 PALMER & SNELL*/


/*GEORGE LUGER AUGUST 1978*/

steady :- cue(timesys(episode,start,finish)),
  cue(timesys(period1,start,change1)),
  cue(timesys(period2,change1,finish)),
  cue(pathsys(path,top,bottom,top,stline)),
  cue(pathsys(seg1,top,inter,top,stline)),
  cue(pathsys(seg2,inter,bottom,inter,stline)),
    checklist(dbentry,[
      partition(episode,[period1,period2]),
      partition(path,[seg1,seg2]),
      duration(period1,tq1),
      constlength(path,dq0),
      duration(period2,tq2),
      constlength(seg1,dq1),
      constlength(seg2,dq2),
      duration(episode,tq0),
      vel(particle,zero,270,start),
      accel(particle,aq0,270,period1),
      vel(particle,vqf,270,finish),
      accel(particle,aq0,270,period2),
      measure(tq1,t1,arbs),
      measure(tq2,t2,arbs),
      measure(tq0,t0,arbs),
      measure(dq1,d1,arbs),
      measure(dq2,d2,arbs),
      measure(dq0,d0,arbs),
      measure(aq1,g,arbs),
      measure(aq2,g,arbs),
      measure(aq0,g,arbs),
      measure(vqf,vf,arbs)   ]),
  cue(motion(particle,path,top,left,episode)),
  cue(motion(particle,seg1,top,left,period1)),
  cue(motion(particle,seg2,inter,left,period2)),

  checklist(asserta,[given(aq1),given(aq2), given(aq0),
             given(dq2), given(tq2), sought(dq0)  ]).



:- end.
```

weight.nr6

```
/* simple spring problem -- weight on end of spring */

 problem(weight,'Weight problem.\n%r=\n\n',[====,4]).

 cue linesys(string,spring1,[top1,bottom1]).

 isa(period,now).
 isa(particle,weight1).
 mass(weight1,mass1,now).
 incline(spring1,90,bottom1).
 fixed_contact(weight1,bottom1,now).
 elastic(spring1,true).
 elasticity(spring1,elast,now).
 varlength(spring1,len,now).
 extension(spring1,extens,now).
 natlength(spring1,natL,now).
 tension(spring1,t1,now).

 given(mass1).
 given(elast).
 given(natL).
 sought(t1).
 sought(len).
 sought(extens).


 :- assert( minimal_part(string,Str,Pt,Str) ),
    assert( static(_,_) ).
```

weight.prb

```
/* simple statics problem -- weight on end of string */

 problem(weight,'Weight problem.\n%r=\n\n',[====,4]).

 cue linesys(string,string1,[top1,bottom1]).

 isa(period,now).
 isa(particle,weight1).
 mass(weight1,mass1,now).
 incline(string1,90,bottom1).
 fixed_contact(weight1,bottom1,now).
 tension(string1,t1,now).

 given(mass1).
 sought(t1).


 :- assert( minimal_part(string,Str,Pt,Str) ),
    assert( static(_,_) ).
```

wgt2.prb

```
/* weight suspended by two springs in series */

  problem('2spr.prb','Two Springs problem.\n%r=\n\n',[====,4]).

  cue linesys(string,spring1,[top1,bottom1]).
  cue linesys(string,spring2,[top2,bottom2]).

  isa(period,now).
  isa(particle,weight1).
  mass(weight1,mass1,now).
  incline(spring1,90,bottom1).
  incline(spring2,90,bottom2).
  fixed_contact(weight1,bottom2,now).
  fixed_contact(bottom1,top2,now).

  elastic(spring1,true).
  elasticity(spring1,elast,now).
  extension(spring1,extens1,now).
  natlength(spring1,natL,now).

  elastic(spring2,true).
  elasticity(spring2,2*elast,now).
  extension(spring2,extens2,now).
  natlength(spring2,natL,now).

  tension(spring1,t1,now).
  tension(spring2,t2,now).

  given(mass1).
  given(elast).
  given(natL).
  sought(t1).
  sought(t2).
  sought(extens1).
  sought(extens2).

  :- assert( minimal_part(string,Str,Pt,Str) ),
     assert( static(_,_) ),
     /* Kludge for 'incline' fix to tell it about current period */
     assert( current_period(now) ).
```

```
/* The minimal part of a light string under tension is straight */   concav.pic

  concavity(String,stline) :-
    type(string,String),
    /* KLUDGE!  The period should be passed in as an argument */
    current_period(Per),
    cc tension(String,T,Per),
    cc mass(String,M,Per),
    condition(M =:= 0),
    condtion(T > 0).
```

varlen.fix

```
varlensth(Line1,D,Time) :-
   farend(Line1,End1,End2),
   separation(End1,End2,D,Dir,Time).
```

```
MM      MM    000000    TTTTTTTTTT  NN      NN      11                    SSSS
MM      MM    000000    TTTTTTTTTT  NN      NN      11                    SSSS
MMMM  MMMM  00      00      TT      NN      NN    1111                  SS
MMMM  MMMM  00      00      TT      NN      NN    1111                  SS
MM  MM  MM  00      00      TT      NNNN    NN      11                  SS
MM  MM  MM  00      00      TT      NNNN    NN      11                  SS
MM      MM  00      00      TT      NN  NN  NN      11                    SSSS
MM      MM  00      00      TT      NN  NN  NN      11                    SSSS
MM      MM  00      00      TT      NN    NNNN      11
MM      MM  00      00      TT      NN    NNNN      11
MM      MM  00      00      TT      NN      NN      11          ....
MM      MM  00      00      TT      NN      NN      11          ....
MM      MM    000000        TT      NN      NN    111111        ....    SSSSSS
MM      MM    000000        TT      NN      NN    111111        ....    SSSSSS


44    44      000000      000000                  44    44    44    44        11
44    44      000000      000000                  44    44    44    44        11
44    44    00      00  00      00                44    44    44    44      1111
44    44    00      00  00      00                44    44    44    44      1111
44    44    00    0000  00    0000                44    44    44    44        11
44    44    00    0000  00    0000                44    44    44    44        11
4444444444  00  00  00  00  00  00                4444444444  4444444444      11
4444444444  00  00  00  00  00  00                4444444444  4444444444      11
      44    0000    00  0000    00                      44          44        11
      44    0000    00  0000    00                      44          44        11
      44    00      00  00      00      ,,,,            44          44        11
      44    00      00  00      00      ,,,,            44          44        11
      44      000000      000000          ,,            44          44      1111
      44      000000      000000          ,,            44          44      1111


L       AAA   W   W RRRR  EEEEE N   N  CCCC EEEEE
L     A   A W   W R   R E     N   N C     E
L     A   A W   W R   R E     NN  N C     E
      A   A W   W RRRR  EEEE  N N N C     EEEE
L     AAAAA W W W R R   E     N  NN C     E
L     A   A WW WW R  R  E     N   N C     E
LLLLL A   A W   W R   R EEEEE N   N  CCCC EEEEE


 *START* User LAWRENCE [400,441]  Job  MOTN1 Seq. 7495 Date 07-Apr-81 11:51:33
File: DSKA:MOTN1.SYN<055>[400,441,MOTN] Created: 29-Jan-81 10:24:01 Printed: C
QUEUE Switches:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:257 /FORMS:NORMAL

File: DSKA:MOTN1.SYN<055>[400,441,MOTN] will be DELETED after printing
```

```
/* PROB2
   First motion problem -
     what Rob's program might produce
   C.M., 25/9/80

   The problem is:

        A stone is dropped from a cliff 100 meters above the sea.
        Find the speed with which it hits the sea.

*/

sentence(s2).
stype(s2,question).
sentence(s1).
stype(s1,statement).

/* Second sentence */

/* Relative clause */

aux_verb(s3,[pres]).
main_verb(s3,hit).
syn_subj(s3,np6).
syn_obj(s3,np7).
pp_linked(s3,pp3).
is_prep(pp3,with,np5).

headnoun(np7,sea).
num(np7,1,def).
headnoun(np6,it).
num(np6,1,def).

embedded_sent(s2,s3).
relc(np5,s3).

/* Main clause */

aux_verb(s2,[pres]).
main_verb(s2,find).
syn_subj(s2,np4).
syn_obj(s2,np5).

headnoun(np5,speed).
num(np5,1,def).
headnoun(np4,you).

/* First sentence */

aux_verb(s1,[pres]).
main_verb(s1,drop).
syn_subj(s1,np1).
pp_linked(s1,pp2).
is_prep(pp2,from,np2).
passive_sent(s1).

measure(qp1,100,meters).
qp_modify(np2,qp1).
pp_linked(qp1,pp1).
is_prep(pp1,above,np3).
```

```
headnoun(np3,sea).
num(np3,1,def).
headnoun(np2,cliff).
num(np2,1,indef).
headnoun(np1,stone).
num(np1,1,indef).
```

```
fixed_contact(object1,plane1,moment1).
fixed_contact(plane1,object1,moment1).

no
| ?- sol(s-3).


Interpreting sentence s-3
Invoking semantics for verb 'calculate'

no
| ?- spy calculate.
You have no clauses for calculate - nothing done.

yes
| ?- core     81920  (53760 lo-seg + 28160 hi-seg)
heap      48640 =  45988 in use +   2652 free
global     1218 =     16 in use +   1202 free
local      1024 =     16 in use +   1008 free
trail       511 =      0 in use +    511 free
   11.24 sec. for 27 GCs gaining 41693 words
    0.-67 sec. for 12 local shifts and 76 trail shifts
   26.24 sec. runtime
```

```
FFFFFFFFFF     000000     RRRRRRRR      CCCCCCCC       11                      PPPPPP
FFFFFFFFFF     000000     RRRRRRRR      CCCCCCCC       11                      PPPPPP
FF           00      00   RR      RR  CC             1111                      PP
FF           00      00   RR      RR  CC             1111                      PP
FF           00      00   RR      RR  CC               11                      PP
FF           00      00   RR      RR  CC               11                      PP
FFFFFFFF     00      00   RRRRRRRR    CC               11                      PPPPPP
FFFFFFFF     00      00   RRRRRRRR    CC               11                      PPPPPP
FF           00      00   RR  RR      CC               11                      PP
FF           00      00   RR  RR      CC               11                      PP
FF           00      00   RR    RR    CC               11               ::::   PP
FF           00      00   RR    RR    CC               11               ::::   PP
FF             000000     RR      RR    CCCCCCCC     111111             ::::   PP
FF             000000     RR      RR    CCCCCCCC     111111             ::::   PP


44     44      000000      000000                  44     44     44     44        11
44     44      000000      000000                  44     44     44     44        11
44     44    00      00  00      00                44     44     44     44      1111
44     44    00      00  00      00                44     44     44     44      1111
44     44    00    0000  00    0000                44     44     44     44        11
44     44    00    0000  00    0000                44     44     44     44        11
4444444444   00  00  00  00  00  00                4444444444   4444444444        11
4444444444   00  00  00  00  00  00                4444444444   4444444444        11
      44     0000    00  0000    00                      44           44          11
      44     0000    00  0000    00                      44           44          11
      44     00      00  00      00         ,,,,         44           44          11
      44     00      00  00      00         ,,,,         44           44          11
      44       000000      000000             ,,         44           44        1111
      44       000000      000000             ,,         44           44        1111


L      AAA   W    W RRRR  EEEEE N   N  CCCC EEEEE
L     A   A W    W R   R E     N   N C     E
L     A   A W    W R   R E     NN  N C     E
L     A   A W    W RRRR  EEEE  N N N C     EEEE
L     AAAAA W W W R R   E     N  NN C     E
L     A   A WW WW R  R  E     N   N C     E
LLLLL A   A W    W R   R EEEEE N   N  CCCC EEEEE


 *START* User LAWRENCE [400,441]  Job  FORC1 Seq. 7496 Date 07-Apr-81 08:54:2
File: DSKA:FORC1.PRB<055>[400,434] Created: 24-Mar-81 09:55:19 Printed: 07-Apr
QUEUE Switches:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:52 /FORMS:NORMAL
```

```
type(object,object1).
isa(particle,object1).
mass(object1,mass1,moment1).
measure(mass1,10,n).
vel(object1,velocity1,moment1).
magnitude(velocity1,scalar1).
measure(scalar1,0,arbs).
accel(object1,acceleration1,moment1).
magnitude(acceleration1,scalar2).
direction(acceleration1,angle2).
measure(scalar2,0,arbs).
isa(plane,plane1).
isa(constant,constant1).
coeff(plane1,constant1).
measure(constant1,mu,arbs).
isa(line,x_axis).
tangent(x_axis,zero).
isa(angle,angle1).
tangent(plane1,angle1).
equation(zero=angle1+difference1).
measure(difference1,30,degree).
fixed_contact(object1,plane1,moment1).
fixed_contact(plane1,object1,moment1).
supports(plane1,object1).
isa(moment,moment1).
isa(mass,mass1).
isa(velocity,velocity1).
isa(scalar,scalar1).
isa(acceleration,acceleration1).
isa(scalar,scalar2).
isa(angle,angle2).
type(scraper,object1).
isa(force,force1).
isa(scalar,scalar3).
sought(scalar3).
magnitude(force1,scalar3).
friction(object1,plane1,force1,moment1).
given(mass1).
given(scalar1).
given(scalar2).
given(constant1).
given(difference1).
```

```
MM      MM    000000    TTTTTTTTTT  NN      NN      11                  PPPPP
MM      MM    000000    TTTTTTTTTT  NN      NN      11                  PPPPP
MMMM  MMMM  00      00      TT      NN      NN    1111                  PP
MMMM  MMMM  00      00      TT      NN      NN    1111                  PP
MM  MM  MM  00      00      TT      NNNN    NN      11                  PP
MM  MM  MM  00      00      TT      NNNN    NN      11                  PP
MM      MM  00      00      TT      NN  NN  NN      11                  PPPPP
MM      MM  00      00      TT      NN  NN  NN      11                  PPPPP
MM      MM  00      00      TT      NN    NNNN      11                  PP
MM      MM  00      00      TT      NN    NNNN      11                  PP
MM      MM  00      00      TT      NN      NN      11          ....    PP
MM      MM  00      00      TT      NN      NN      11          ....    PP
MM      MM    000000        TT      NN      NN    111111        ....    PP
MM      MM    000000        TT      NN      NN    111111        ....    PP


44    44      000000      000000                  44    44    44    44        1
44    44      000000      000000                  44    44    44    44        1
44    44    00      00  00      00                44    44    44    44      111
44    44    00      00  00      00                44    44    44    44      111
44    44    00    0000  00    0000                44    44    44    44        1
44    44    00    0000  00    0000                44    44    44    44        1
4444444444  00  00  00  00  00  00                4444444444  4444444444      1
4444444444  00  00  00  00  00  00                4444444444  4444444444      1
      44    0000    00  0000    00                      44          44        1
      44    0000    00  0000    00                      44          44        1
      44    00      00  00      00      ,,,,            44          44        1
      44    00      00  00      00      ,,,,            44          44        1
      44      000000      000000          ,,            44          44      111
      44      000000      000000          ,,            44          44      111


L      AAA  W   W RRRR  EEEEE N   N  CCCC EEEEE
L     A   A W   W R   R E     N   N C     E
L     A   A W   W R   R E     NN  N C     E
L     A   A W   W RRRR  EEEE  N N N C     EEEE
L     AAAAA W W W R R   E     N  NN C     E
L     A   A WW WW R  R  E     N   N C     E
LLLLL A   A W   W R   R EEEEE N   N  CCCC EEEEE
```

START User LAWRENCE [400,441] Job MOTN1 Seq. 7495 Date 07-Apr-81 09:02:4
File: DSKA:MOTN1.PRB<005>[400,441,MOTN] Created: 29-Jan-81 10:24:00 Printed:
QUEUE Switches: /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:257 /FORMS:NORMAL

File: DSKA:MOTN1.PRB<005>[400,441,MOTN] will be DELETED after printing

```
isa(stone,stone1).
isa(cliff,cliff1).
isa(surface,sea).
isa(solid,sea).
isa(point,point1).
point_of(sea,point1).
isa(moment,moment1).
isa(moment,moment2).
consec_moments(moment1,moment2).
isa(separation,separation1).
isa(scalar,scalar1).
magnitude(separation1,scalar1).
isa(angle,angle1).
direction(separation1,angle1).
separation(cliff1,point1,separation1,moment2).
measure(scalar1,100,meters).
measure(angle1,90,degrees).
isa(velocity,velocity1).
vel(stone1,velocity1,moment2).
isa(scalar,scalar2).
magnitude(velocity1,scalar2).
measure(scalar2,0,arbs).
isa(angle,angle2).
direction(velocity1,angle2).
unsupported(stone1,moment2).
isa(point_of_ref,point_of_ref1).
pathat(stone1,point_of_ref1,moment2).
at(stone1,cliff1,moment2).
isa(solid,sea).
isa(moment,moment3).
consec_moments(moment2,moment3).
isa(system,system1).
motion(stone1,cliff1,point1,moment2,moment3,system1).
isa(motion,system1).
isa(period,period1).
```

```
MM        MM      000000      TTTTTTTTTT   NN        NN     333333              PPPPPP
MM        MM      000000      TTTTTTTTTT   NN        NN     333333              PPPPPP
MMMM    MMMM    00        00      TT       NN        NN   33        33          PP
MMMM    MMMM    00        00      TT       NN        NN   33        33          PP
MM  MM  MM      00        00      TT       NNNN      NN             33          PP
MM  MM  MM      00        00      TT       NNNN      NN             33          PP
MM        MM    00        00      TT       NN   NN   NN           33            PPPPPP
MM        MM    00        00      TT       NN   NN   NN           33            PPPPPP
MM        MM    00        00      TT       NN      NNNN             33          PP
MM        MM    00        00      TT       NN      NNNN             33          PP
MM        MM    00        00      TT       NN        NN   33        33    ....  PP
MM        MM    00        00      TT       NN        NN   33        33    ....  PP
MM        MM      000000          TT       NN        NN     333333      ....    PP
MM        MM      000000          TT       NN        NN     333333      ....    PP
```

```
44    44      000000      000000                  44    44    44    44       11
44    44      000000      000000                  44    44    44    44       11
 4    44    00      00  00      00                44    44    44    44     1111
44    44    00      00  00      00                44    44    44    44     1111
44    44    00    0000  00    0000                44    44    44    44       11
44    44    00    0000  00    0000                44    44    44    44       11
4444444444  00  00  00  00  00  00                4444444444  4444444444     11
4444444444  00  00  00  00  00  00                4444444444  4444444444     11
      44    0000    00  0000    00                      44          44       11
      44    0000    00  0000    00                      44          44       11
      44    00      00  00      00      ,,,,            44          44       11
      44    00      00  00      00      ,,,,            44          44       11
      44      000000      000000          ,,            44          44     1111
      44      000000      000000          ,,            44          44     1111
```

```
L       AAA   W     W RRRR   EEEEE N     N   CCCC EEEEE
L      A   A  W     W R    R E     N     N  C     E
L      A   A  W     W R    R E     NN    N  C     E
L      A   A  W     W RRRR   EEEE  N  N  N  C     EEEE
L      AAAAA  W  W  W R  R   E     N    NN  C     E
L      A   A  WW   WW R   R  E     N     N  C     E
LLLLL  A   A  W     W R    R EEEEE N     N   CCCC EEEEE
```

```
 *START* User LAWRENCE [400,441]  Job  MOTN1 Seq. 7495 Date 07-Apr-81 11:51:3
File: DSKA:MOTN3.PRB<005>[400,441,MOTN] Created: 29-Jan-81 10:24:01 Printed: (
QUEUE Switches:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:257 /FORMS:NORMAL

File: DSKA:MOTN3.PRB<005>[400,441,MOTN] will be DELETED after printing
```

```
isa(stone,stone1).
isa(moment,moment1).
isa(moment,moment2).
consec_moments(moment1,moment2).
isa(velocity,velocity1).
vel(stone1,velocity1,moment2).
isa(scalar,scalar1).
magnitude(velocity1,scalar1).
isa(angle,angle1).
direction(velocity1,angle1).
measure(angle1,90,degrees).
isa(point_of_ref,point_of_ref1).
pathat(stone1,point_of_ref1,moment2).
unsupported(stone1,moment2).
isa(system,system1).
maximal_motion(stone1,system1).
isa(motion,system1).
isa(period,period1).
period_of(system1,period1).
isa(moment,moment3).
period(period1,moment2,moment3).
isa(point_of_ref,point_of_ref2).
pathat(stone1,point_of_ref2,moment3).
isa(line,line1).
farend(line1,point_of_ref1,point_of_ref2).
path_of(system1,line1).
pathsys(line1,point_of_ref1,point_of_ref2).
isa(path,line1).
motion_of(stone1,period1,system1).
object_of(system1,stone1).
isa(moment,moment4).
isa(period,period2).
period(period2,moment2,moment4).
measure(period2,3,s).
isa(system,system2).
motion_of(stone1,period2,system2).
isa(motion,system2).
period_of(system2,period2).
object_of(system2,stone1).
isa(path,path1).
path_of(system2,path1).
isa(point,point1).
isa(point,point2).
pathsys(path1,point1,point2).
farend(path1,point1,point2).
pathat(stone1,point2,moment4).
pathat(stone1,point1,moment2).
isa(length,length1).
distance(stone1,length1,period2).
sought(length1).
given(angle1).
given(period2).
```

```
MM        MM     000000     TTTTTTTTTT  NN        NN     333333                    SSSS
MM        MM     000000     TTTTTTTTTT  NN        NN     333333                    SSSS
MMMM    MMMM  00        00      TT      NN        NN  33        33               SS
MMMM    MMMM  00        00      TT      NN        NN  33        33               SS
MM  MM  MM    00        00      TT      NNNN      NN            33               SS
MM  MM  MM    00        00      TT      NNNN      NN            33               SS
MM        MM  00        00      TT      NN  NN    NN          33                   SSSS
MM        MM  00        00      TT      NN  NN    NN          33                   SSSS
MM        MM  00        00      TT      NN    NNNN              33
MM        MM  00        00      TT      NN    NNNN              33
MM        MM  00        00      TT      NN        NN  33        33     ....
MM        MM  00        00      TT      NN        NN  33        33     ....
MM        MM     000000         TT      NN        NN     333333        ....      SSSSSS
MM        MM     000000         TT      NN        NN     333333        ....      SSSSSS
```

```
44    44     000000      000000                44    44    44    44        11
44    44     000000      000000                44    44    44    44        11
 4    44  00        00  00        00           44    44    44    44      1111
44    44  00        00  00        00           44    44    44    44      1111
44    44  00      0000  00      0000           44    44    44    44        11
44    44  00      0000  00      0000           44    44    44    44        11
4444444444  00  00  00  00  00  00             4444444444  4444444444      11
4444444444  00  00  00  00  00  00             4444444444  4444444444      11
      44  0000      00  0000      00                 44          44        11
      44  0000      00  0000      00                 44          44        11
      44  00        00  00        00   ,,,,          44          44        11
      44  00        00  00        00   ,,,,          44          44        11
      44     000000      000000          ,,          44          44      1111
      44     000000      000000          ,,          44          44      1111
```

```
L       AAA   W     W RRRR   EEEEE N     N  CCCC EEEEE
L      A   A  W     W R    R E     N     N C     E
L      A   A  W     W R    R E     NN    N C     E
.      A   A  W     W RRRR   EEEE  N  N  N C     EEEE
L      AAAAA  W  W  W R  R   E     N    NN C     E
L      A   A  WW   WW R   R  E     N     N C     E
LLLLL  A   A  W     W R    R EEEEE N     N  CCCC EEEEE
```

```
 *START* User LAWRENCE [400,441]  Job  MOTN1 Seq. 7495 Date 07-Apr-81 11:51:3
File: DSKA:MOTN3.SYN<055>[400,441,MOTN] Created: 29-Jan-81 10:24:01 Printed: (
QUEUE Switches:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:257 /FORMS:NORMA

File: DSKA:MOTN3.SYN<055>[400,441,MOTN] will be DELETED after printing
```

```
/* MOTN3.SYN
   Proposed syntactic analysis
   C.M., 16/10/80
*/

sentence(s2).
main_verb(s2,find).
aux_verb(s2,[pres]).
stype(s2,command).
syn_subj(s2,np4).
headnoun(np4,you).
syn_obj(s2,np5).
num(np5,_,def).
headnoun(np5,distance).
relc(np5,s3).

   passive_sent(s3).
   aux_verb(s3,[pres]).
   main_verb(s3,travel).
   stype(s3,statement).
   syn_subj(s3,np6).
   wh_trace(np6,np5).
   pp_linked(s3,pp3).
   is_prep(pp3,by,np7).
   num(np7,1,def).
   headnoun(np7,stone).
   pp_linked(s3,pp4).
   is_prep(pp4,in,np8).
   num(np8,1,def).
   hasfeat(np8,first).
   qp_det(np8,qp2).
   measure(qp2,3,s).
   pp_linked(np8,pp5).
   is_prep(pp5,of,np9).
   poss_det(np10,np9).
   headnoun(np9,motion).
   headnoun(np10,it).

sentence(s1).
passive_sent(s1).
aux_verb(s1,[pres]).
stype(s1,statement).
main_verb(s1,project).
syn_subj(s1,np1).
num(np1,1,indef).
headnoun(np1,stone).
pp_linked(s1,pp1).
is_prep(pp1,with,np2).
num(np2,1,indef).
headnoun(np2,speed).
pp_linked(np2,pp2).
is_prep(pp2,of,np3).
num(np3,[],[]).
qp_det(np3,qp1).
measure(qp1,21,'ms-1').
adverb(s1,adv1).
headadv(adv1,upward).
hasfeat(adv1,vertically).
```

```
MM        MM      000000     TTTTTTTTTT  NN        NN  5555555555             SSSS
MM        MM      000000     TTTTTTTTTT  NN        NN  5555555555             SSSS
MMMM    MMMM    00      00       TT      NN        NN  55                   SS
MMMM    MMMM    00      00       TT      NN        NN  55                   SS
MM  MM  MM      00      00       TT      NNNN      NN  555555               SS
MM  MM  MM      00      00       TT      NNNN      NN  555555               SS
MM        MM    00      00       TT      NN  NN    NN        55               SSSS
MM        MM    00      00       TT      NN  NN    NN        55               SSSS
MM        MM    00      00       TT      NN      NNNN          55
MM        MM    00      00       TT      NN      NNNN          55
MM        MM    00      00       TT      NN        NN  55      55      ....
MM        MM    00      00       TT      NN        NN  55      55      ....
MM        MM      000000         TT      NN        NN    555555        ....   SSSSSS
MM        MM      000000         TT      NN        NN    555555        ....   SSSSSS


44      44      000000       000000                    44      44      44      44         11
44      44      000000       000000                    44      44      44      44         11
44      44    00      00   00      00                  44      44      44      44       1111
44      44    00      00   00      00                  44      44      44      44       1111
44      44    00    0000   00    0000                  44      44      44      44         11
44      44    00    0000   00    0000                  44      44      44      44         11
4444444444    00  00  00   00  00  00                  4444444444      4444444444         11
4444444444    00  00  00   00  00  00                  4444444444      4444444444         11
        44    0000    00   0000    00                          44              44         11
        44    0000    00   0000    00                          44              44         11
        44    00      00   00      00       ,,,,               44              44         11
        44    00      00   00      00       ,,,,               44              44         11
        44      000000       000000           ,,               44              44       1111
        44      000000       000000           ,,               44              44       1111


L       AAA   W     W RRRR   EEEEE N     N  CCCC EEEEE
L      A   A W     W R    R E      N     N C     E
L      A   A W     W R    R E      NN    N C     E
L      A   A W     W RRRR   EEEE   N N   N C     EEEE
L      AAAAA W  W  W R  R   E      N   NN C     E
L      A   A WW   WW R   R  E      N     N C     E
LLLLL  A   A W     W R    R EEEEE N     N  CCCC EEEEE
```

START User LAWRENCE [400,441] Job MOTN1 Seq. 7495 Date 07-Apr-81 11:51:3
File: DSKA:MOTN5.SYN<055>[400,441,MOTN] Created: 29-Jan-81 10:24:02 Printed: (
QUEUE Switches: /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:257 /FORMS:NORMA

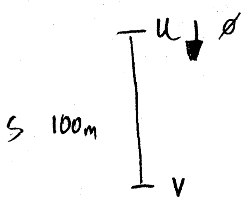
File: DSKA:MOTN5.SYN<055>[400,441,MOTN] will be DELETED after printing

```
/* MOTN5.SYN
   Proposed output of the parser for MOTN5
   C.M., 31/10/80
*/

sentence(s4).
sentence(s2).
sentence(s1).

passive_sent(s1).
aux_verb(s1,[pres]).
stype(s1,statement).
main_verb(s1,drop).
syn_subj(s1,np1).
num(np1,1,indef).
headnoun(np1,stone).
pp_linked(s1,pp1).
is_prep(pp1,from,np2).
num(np2,_,def).
headnoun(np2,top).
pp_linked(np2,pp2).
is_prep(pp2,of,np3).
num(np3,1,indef).
headnoun(np3,tower).

aux_verb(s2,[pres]).
main_verb(s2,fall).
pp_linked(s2,pp3).
is_prep(pp3,in,np4).
num(np4,_,def).
hasfeat(np4,last).
haednoun(np4,second).
pp_linked(np4,pp4).
is_prep(pp4,of,np5).
poss_det(np6,np5).
headnoun(np6,it).
headnoun(np5,motion).
syn_subj(s2,np7).
headnoun(np7,it).
pp_linked(s2,pp5).
is_prep(pp5,through,np8).
num(np8,1,indef).
headnoun(np8,distance).
relc(np8,s3).

   aux_verb(s3,[pres]).
   main_verb(s3,be).
   syn_subj(s3,np9).
   wh_trace(np9,np8).
   syn_obj(s3,ap1).
   qp_modify(ap1,qp1).
   measure(qp1,'1/5',arbs).
   comparative(ap1,np10).
   num(np10,_,def).
   headnoun(np10,height).
   pp_linked(np10,pp6).
   is_prep(pp6,of,np11).
   num(np11,_,def).
   headnoun(np11,tower).
```

```
aux_verb(s4,[pres]).
stype(s4,command).
main_verb(s4,find).
syn_subj(s4,np12).
headnoun(np12,you).
syn_obj(s4,np13).
num(np13,_,def).
headnoun(np13,tower).
```

Motion 1

BASICS



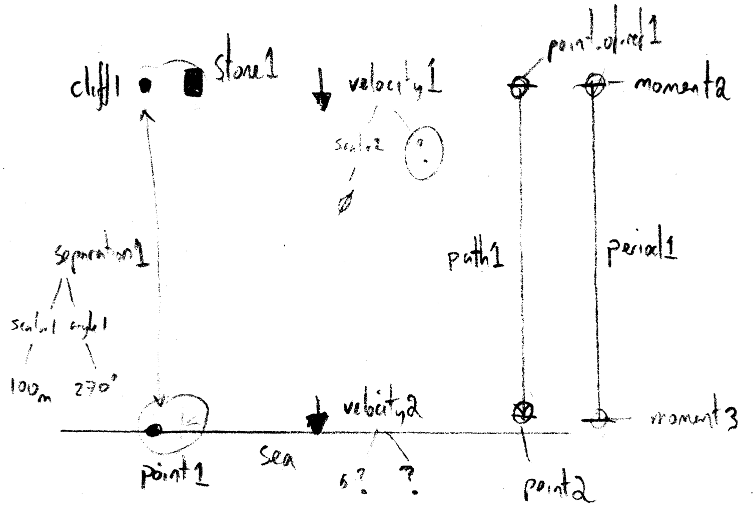
[v] [u s]

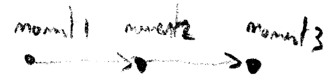
[vel(?)] [vel(?) length]

applicable : [relvel, convenergy, constvel, constaccel-1, constaccel-2, constaccel-3, constaccel-4]

REPN

unsupported(stone1, ~)





given(scalar1)
given(angle1)
given(scalar2)

sought(velocity2)

Quantities

separation1
velocity1
velocity2

SOLN

velocity2 → constaccel3 $V^2 - u^2 = 2AS$

→ consvenergy $gH = \frac{1}{2}v^2 - \frac{1}{2}u^2$

3) meta/object confusions

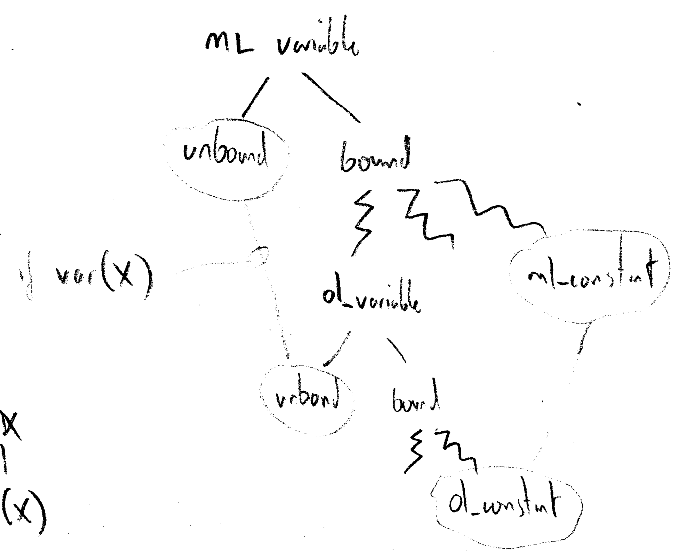
NB meta(X) &
< foo(A) >
→ meta2(X, A)

⇒

meta(X) &
true(Fact) &
predicate(Fact, foo, 1) &
arg(1, Fact, A) &
→ meta2(X, A).

Two uses : - moving object level constants about
- moving object level variables about
In this case the OL & ML vars become unified !

ol_bound( ) ml_bound( )
ol_unbound( ) ml_unbound( )
ol_symbol( ) ml_symbol( )



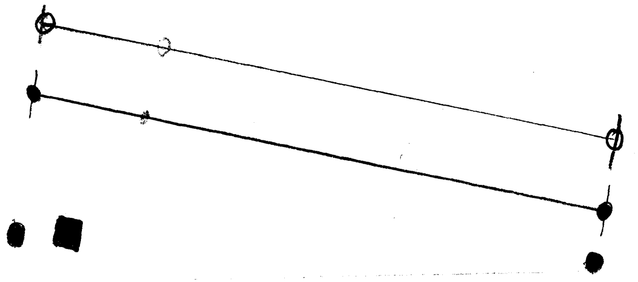
use MX
|
ol(X)

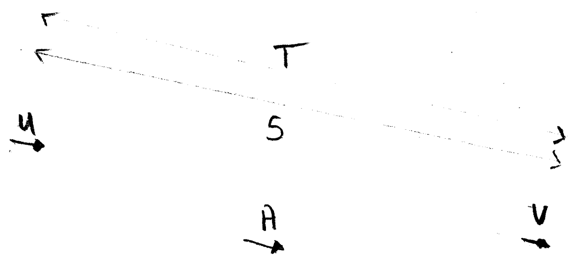
tension(s1 . ol(T) . t2) - Need to do own matching

Motion

Motion
- Object
- Start point
- End point
- Start moment
- End moment

- Path
  - Path start
  - Path end
- Period





u	initial vel
V	final vel
A	acceleration [constant]
S	distance travelled
T	time taken

Motion 1

thoughts

1) magnitude (X, ∅) → angle (X, k ?)

NB no angle given for velocity1

2) separation (cliff1, point1, s1, moment2)

unsupported (stone1, moment2)

| Only about initial moments
| require facts over full period of motion.

- The separation is forever
- How to use the initial unsupported to justify the default of being unsupported

(A) {default rules}

unsupported (ob, P) ← initial (P, M1) & unsupported (ob, M1)

unsupported (ob, P)

No hook for conditions/support for defaults.

(B) {inference_rules}

unsupported (ob, P) ← not ∃x. contact(ob, x, P).

Perhaps no need? General default condition – closed world assumption

(A) & (B) whats the payoff?

3) Magnitudes and Angles, esp. re: v^2 etc.

Formulae

constvel	$s = vt$	[length, vel, duration]	(u	v)		s	T
constaccel 1	$v = u + at$	[vel, accel, duration]	u	v	A		T
2	$s = ut + \frac{1}{2}at^2$	[length, vel, accel, duration]	u		A	S	T
3	$v^2 - u^2 = 2as$	[vel, accel, length]	u	v	A	S	
4	$s = vt - \frac{1}{2}at^2$	[length, vel, accel, duration]		v	A	S	T

initial vel

final vel.

distance travelled.

acceleration.

time taken.

motion(obj, u, v, A, S, T)

---

consvenergy	$gH = \frac{v^2}{2} - \frac{u^2}{2}$	[length, vel]	u	v		S	

<u>Prepare</u>

* vel(obj, V, Time) → situation(obj, Time)

* accel(obj, A, Time) → situation(obj, Time)

* dist_travelled(obj, D, Time) → situation(obj, Time)

duration(Time, Dur)
& period_of(Motion, Time)
& object_of(motion, obj) → situation(obj, Time)

(the four rules above bracketed together:) motion_of(obj, Time, Motion)

←

initvel(obj, U, Time)

finvel(obj, V, Time)

dist_travelled(obj, S, Time)

accel(obj, A, Time)

duration(Time, T)

PROBLEMS

1) Explosive nature of transitive formulae

A) relvel

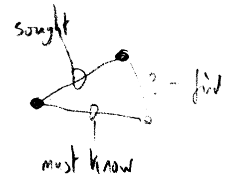
relaccel

B) timesum

lengthsum

We require a suitable generalisation of the sameclass mechanism.

A) (vectors) Rely on knowing the other arc + loop check (heuristic?)



applicable(relvel) &

< relvel(A, B, Vel, P) &

oneof(A, B, X) &

NB - relation! relvel(X, C, Vel2, P) >

→ strategy(relvel, situation(A, B, C, P)).

B) (scalar) maximal rules
either forward or backward.

---

2) Need taxonomy of <situations>

Note that this occurs as a function in strategies

what are the possible values?

(WAIT)

1

u

100m

v

$v = u + at$

$s = ut + \frac{1}{2}at^2$

~~[illegible]~~

~~$s = ut + \frac{1}{2}at$~~

$$v^2 - u^2 = 2as$$

$v^2 = 2a100 - u^2$

$v = \sqrt{\quad}$

[v] [u, s]

2

$\cdot - \emptyset$

10m

$\cdot\ v$

[t, u] [v, s]

$v^2 - u^2 = 2as$

$-u^2 = -2a10$

3

3s

$u = 21$

$s = ut + \frac{1}{2}at^2$

[s] [t, u]

4

~~$v = u + at^2$~~

u 13 m s⁻¹

7

v

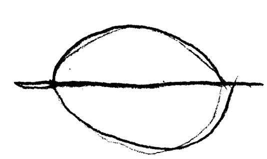
$v^2 - u^2 = 2as$

$v^2 = u^2 - 2g7$

[v]

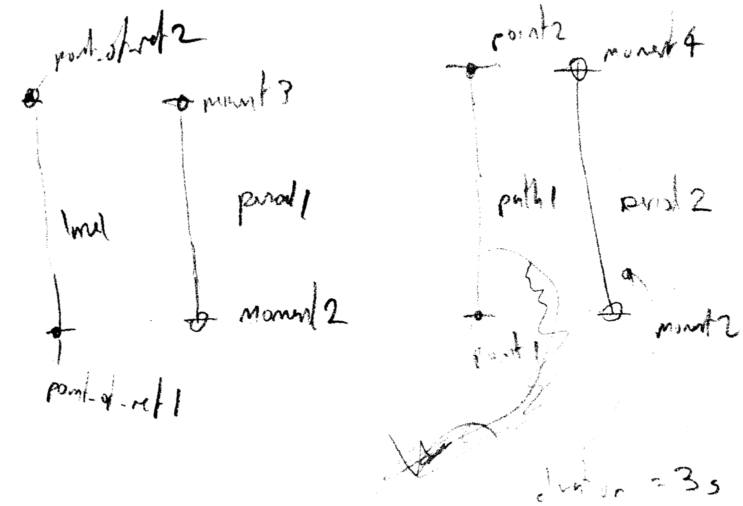
u

a



motion 2

unsupported(stone1, moment2)



velocity1 • stone1

scalar1 angle1

90°

21

dist_travelled(length1)

given(angle1)

given(duration)

~~sought(point2)~~ sought(length1)

U

T

S

A

$$s = ut + \frac{1}{2}at^2$$

---

[s] [t u]

s I ↑ u=21 t=3

[lengthsum, consvenergy, constvel, constacc-2, constacc-3, constacc-4]

I X — I I

MOTN1

```
isa(stone,stone1).
isa(cliff,cliff1).
isa(surface,sea).
isa(solid,sea).
isa(point,point1).
point_of(sea,point1).
isa(moment,moment1).
isa(moment,moment2).
consec_moments(moment1,moment2).
isa(separation,separation1).
isa(scalar,scalar1).
magnitude(separation1,scalar1).
isa(angle,angle1).
direction(separation1,angle1).
separation(cliff1,point1,separation1,moment2).
measure(scalar1,100,meters).
measure(angle1,90,degrees).
isa(velocity,velocity1).
vel(stone1,velocity1,moment2).
isa(scalar,scalar2).
magnitude(velocity1,scalar2).
easure(scalar2,0,arbs).
isa(angle,angle2).
direction(velocity1,angle2).
unsupported(stone1,moment2).
isa(point_of_ref,point_of_ref1).
pathat(stone1,point_of_ref1,moment2).
at(stone1,cliff1,moment2).
isa(solid,sea).
isa(moment,moment3).
consec_moments(moment2,moment3).
isa(system,system1).
motion(stone1,cliff1,point1,moment2,moment3,system1).
isa(motion,system1).
isa(period,period1).
period_of(system1,period1).
period(period1,moment2,moment3).
motion_of(stone1,period1,system1).
object_of(system1,stone1).
isa(path,path1).
ath_of(system1,path1).
isa(point,point2).
pathsys(path1,point_of_ref1,point2).
farend(path1,point_of_ref1,point2).
pathat(stone1,point2,moment3).
at(stone1,point1,moment3).
isa(velocity,velocity2).
vel(stone1,velocity2,moment3).
sought(velocity2).
given(scalar1).
given(angle1).
given(scalar2).
```

MOTN 3

```
isa(stone,stone1).
isa(moment,moment1).
isa(moment,moment2).
consec_moments(moment1,moment2).
isa(velocity,velocity1).
vel(stone1,velocity1,moment2).
isa(scalar,scalar1).
magnitude(velocity1,scalar1).
isa(angle,angle1).
direction(velocity1,angle1).
measure(angle1,90,degrees).
isa(point_of_ref,point_of_ref1).
pathat(stone1,point_of_ref1,moment2).
unsupported(stone1,moment2).
isa(system,system1).
maximal_motion(stone1,system1).
isa(motion,system1).
isa(period,period1).
period_of(system1,period1).
isa(moment,moment3).
period(period1,moment2,moment3).
isa(point_of_ref,point_of_ref2).
pathat(stone1,point_of_ref2,moment3).
isa(line,line1).
farend(line1,point_of_ref1,point_of_ref2).
path_of(system1,line1).
pathsys(line1,point_of_ref1,point_of_ref2).
isa(path,line1).
motion_of(stone1,period1,system1).
object_of(system1,stone1).
isa(moment,moment4).
isa(period,period2).
period(period2,moment2,moment4).
measure(period2,3,s).
isa(system,system2).
motion_of(stone1,period2,system2).
isa(motion,system2).
period_of(system2,period2).
object_of(system2,stone1).
isa(path,path1).
path_of(system2,path1).
isa(point,point1).
isa(point,point2).
pathsys(path1,point1,point2).
farend(path1,point1,point2).
pathat(stone1,point2,moment4).
pathat(stone1,point1,moment2).
isa(length,length1).
distance(stone1,length1,period2).
sought(length1).
given(angle1).
given(period2).
```

duration

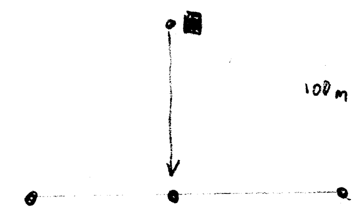
duration

MOTNI.PRB

```
isa(stone,stone1).
isa(cliff,cliff1).
isa(surface,sea).
isa(solid,sea).
isa(point,point1).
point_of(sea,point1).
isa(moment,moment1).
isa(moment,moment2).
consec_moments(moment1,moment2).
isa(separation,separation1).
isa(scalar,scalar1).
magnitude(separation1,scalar1).
isa(angle,angle1).
direction(separation1,angle1).
separation(cliff1,point1,separation1,moment2).
measure(scalar1,100,meters).
measure(angle1,90,degrees).
isa(velocity,velocity1).
vel(stone1,velocity1,moment2).
isa(scalar,scalar2).
magnitude(velocity1,scalar2).
measure(scalar2,0,arbs).
isa(angle,angle2).
direction(velocity1,angle2).
unsupported(stone1,moment2).
isa(point_of_ref,point_of_ref1).
pathat(stone1,point_of_ref1,moment2).
at(stone1,cliff1,moment2).
isa(solid,sea).
isa(moment,moment3).
consec_moments(moment2,moment3).
isa(system,system1).
motion(stone1,cliff1,point1,moment2,moment3,system1).
isa(motion,system1).
isa(period,period1).
period_of(system1,period1).
period(period1,moment2,moment3).
object_of(system1,stone1).
isa(path,path1).
path_of(system1,path1).
isa(point,point2).
pathsys(path1,point_of_ref1,point2).
farend(path1,point_of_ref1,point2).
pathat(stone1,point2,moment3).
at(stone1,point1,moment3).
isa(velocity,velocity2).
vel(stone1,velocity2,moment3).
sought(velocity2).
given(scalar1).
given(angle1).
given(scalar2).
```

Chris' motion problems

1)




```
cue linesys(period, period1, [start, finish])

cue linesys(surface, sea, [france, target, england])
cue linesys(path, drop, [top, bottom])
isa(particle, stone)

tangent(drop, 270, period1)              /* Infer these? */
concavity(drop, stline)

forced_contact(stone, top, start)        /* should be momentary contact of some sort */
forced_contact(stone, bottom, finish)
fixed_contact(bottom, target, period1)   /* different */

vel(stone, Ø, 270, start)                /* whence the angles? */
vel(stone, v1, 270, finish)

side(stone, drop, left, period1)         /* something like this seems to be required */

separation(top, target, dist, 270, period1)
measure(dist, 100, metres)
```

/* Probably also require something which says that motion actually occurs (can this be inferred?). Also the schema adds that the particle is alway in contact with the typical point of the path. (-see Alan's note on different kinds of typical point though). Saying that the occurring motion is on the path should be sufficient? */

```
/* Here is what my prosram sets for the first motion problem.
   If you have any specific objections to how it looks, please
   let me know.
   I have introduced most of the new notation for motion, etc
   that we discussed. As yet, the stuff that deals with vectors
   is not workins, and so the prosram thinks all quantities are
   scalars.
   Some of the output is a bit stranse, because I found it hard
   to ambush the risht places in LOGIC. For instance, before I
   edited this file, there were 2 copies of each 'isa' assertion
   (because 'isa' soes throush both of 'dbentry' and 'ccreate',
   each of which in seneral needs to be ambushed if one wants a
   record of every assertion produced) Basically, what you see
   here is a lot of thinss to be 'dbentry'ed

        - Chris
*/

isa(moment,mom1)
isa(stone,stone1)
isa(cliff,cliff1)
isa(point,point1)
point_of(surface,sea,point1)
property(point1,internal,0,movins)
typical_point(sea,point1)
isa(motion_start,motion_start1)
isa(moment,mom2)
consec_moments(mom1,mom2)
isa(lensth,lensth1)
isa(ansle,90)
separation(cliff1,point1,lensth1,90,mom2)
measure(lensth1,100,meters)
vel(stone1,vel1,mom2)
measure(vel1,0,arbs)
motion_start(stone1,cliff1,mom2)
at(stone1,cliff1,mom2)
isa(motion_point,motion_point1)
isa(moment,mom3)
consec_moments(mom2,mom3)
motion_point(stone1,point1,mom3)
farend(farend1,mom2,mom3)
isa(motion,motion1)
object_of(motion1,stone1)
period_of(motion1,farend1)
path_of(motion1,path_of1)
isa(path,path_of1)
farend(path_of1,farend2,farend3)
pathsys(path_of1,farend2,farend3)
pathat(stone1,farend2,mom2)
pathat(stone1,farend3,mom3)
at(stone1,point1,mom3)
vel(stone1,vel2,mom3)
sousht(vel2)
```

```
yes
| ?- input(forc1).

yes
| ?- sol(s-1).


Interpreting sentence s-1
Invoking semantics for verb 'rest'
Interpreting noun phrase np-1 (object)
Interpreting Adjectival Phrase (small)
Interpreting PP (of)
Interpreting indefinite measure np-2 (weight)
Interpreting measure pair [[10,n]]
Interpreting PP (in)
Interpreting PP (on)
Interpreting noun phrase np-4 (plane)
Interpreting Adjectival Phrase (rough)
     so that coeff(plane1,constant1)


Interpreting sentence s-2
Invoking semantics for verb 'incline'
Interpreting noun phrase np-6 (<trace>)
Interpreting noun phrase np-8 (horizontal)
     so that typical_point(x_axis,point1)
     so that tangent(plane1,angle1)
     so that typical_point(plane1,point2)
Interpreting measure pair [[30,degree]]

Finalising decisions for sentence s-1
     so that mass(object1,mass1,moment1)
     so that magnitude(velocity1,scalar1)
     so that vel(object1,velocity1,moment1)
     so that magnitude(acceleration1,scalar2)
     so that direction(acceleration1,angle2)
     so that accel(object1,acceleration1,moment1)

yes
| ?- display.
type(object,object1).
isa(particle,object1).
mass(object1,$3,$2).
measure($3,10,n).
vel(object1,$4,$2).
magnitude($4,$5).
measure($5,0,arbs).
accel(object1,$6,$2).
magnitude($6,$7).
direction($6,$8).
measure($7,0,arbs).
isa(plane,plane1).
isa(constant,constant1).
coeff(plane1,$9).
measure($9,mu,arbs).
isa(line,x_axis).
isa(point,point1).
tangent(x_axis,zero).
isa(angle,angle1).
```

```
isa(point,point2).
tangent(plane1,$11).
equation($13= $11+difference1).
measure(difference1,30,degree).
fixed_contact(object1,plane1,$2).
fixed_contact(plane1,object1,$2).
supports(plane1,object1).
isa(moment,moment1).
isa(mass,mass1).
isa(velocity,velocity1).
isa(scalar,scalar1).
isa(acceleration,acceleration1).
isa(scalar,scalar2).
isa(angle,angle2).
fixed_contact(object1,plane1,$2).
fixed_contact(plane1,object1,$2).
given($3).
given($5).
given($7).
given($9).
given(difference1).

yes
| ?- portray($2).
moment1
yes
| ?- in_dbase(record,X), print(X), put("."), nl, fail.
type(object,object1).
isa(particle,object1).
mass(object1,mass1,moment1).
measure(mass1,10,n).
vel(object1,velocity1,moment1).
magnitude(velocity1,scalar1).
measure(scalar1,0,arbs).
accel(object1,acceleration1,moment1).
magnitude(acceleration1,scalar2).
direction(acceleration1,angle2).
measure(scalar2,0,arbs).
isa(plane,plane1).
isa(constant,constant1).
coeff(plane1,constant1).
measure(constant1,mu,arbs).
isa(line,x_axis).
isa(point,point1).
tangent(x_axis,zero).
isa(angle,angle1).
isa(point,point2).
tangent(plane1,angle1).
equation(zero=angle1+difference1).
measure(difference1,30,degree).
fixed_contact(object1,plane1,moment1).
fixed_contact(plane1,object1,moment1).
supports(plane1,object1).
isa(moment,moment1).
isa(mass,mass1).
isa(velocity,velocity1).
isa(scalar,scalar1).
isa(acceleration,acceleration1).
isa(scalar,scalar2).
isa(angle,angle2).
```

Motion

<motion_start>
<motion>

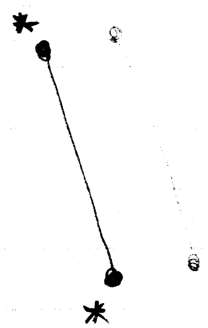
object_of (Motion, object)
period_of (Motion, Period)
path_of (Motion, Path)

motion_start(obj , Point, Moment)
motion_point(obj , (T) point , Moment)

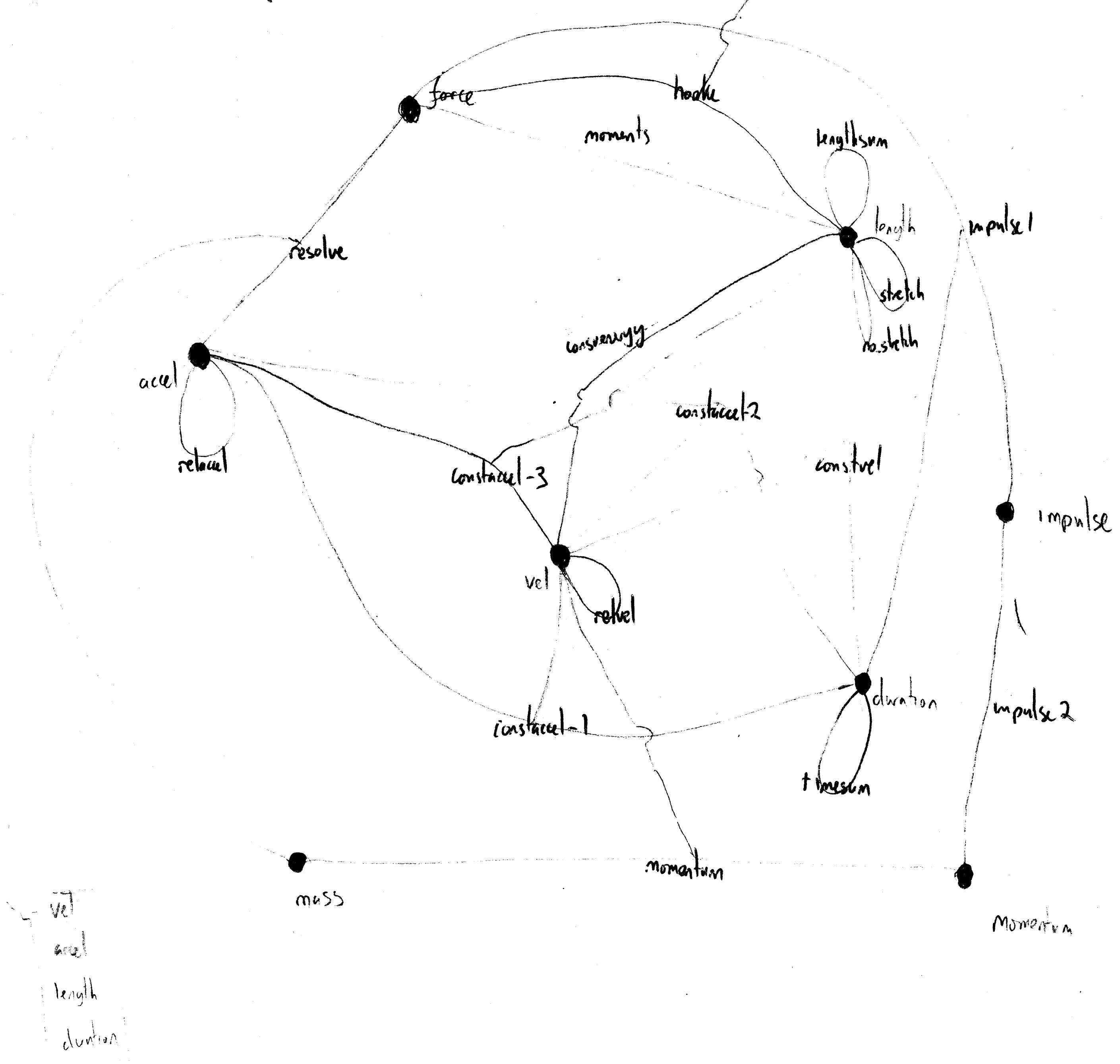
Vectors

magnitude (V , Magn)
direction (V, Dir)

at(ob, Pt, Moment)
pathat(ob, Pt, Moment)

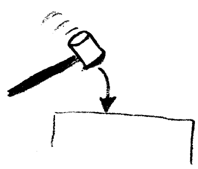


Collision Problems



CRASH

1)



•

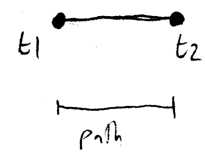
particle(hammer)

magnitude(v1, 15, ms-1)

vel(hammer, v1, t1)

vel(hammer, zero, t2)

mass(hammer, m1)



period(p1, t1, t2)

particle(nail)

impulse(nail, hammer, i1, p1)

given(v1)

{zero}

given(m1)

sought(i1)

↓ m1 v1

$Ft = mv - mu$

↓ ∅

~~prepare(impulse1, Q, impulse, impulse(X, Y, I, Period), situation(X, Y, Period, T1, T2))~~

:-

CRASH

ISFORMS

impulse	force
momentum	time
	mass
	vel

```
isform(impulse1, situation(X, Y, Period, T1, T2), Im = MV - MU,
        impulse(X, Y, Im, Period) &
        momentum(Y, MV, T2) &
        momentum(Y, MU, T1)).
```

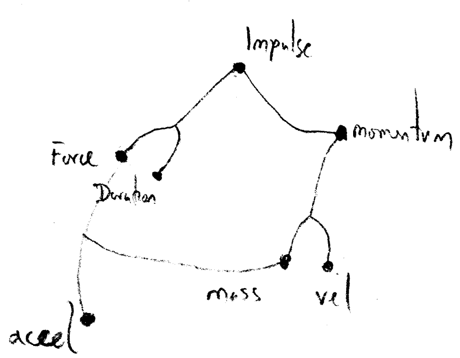
relates(impulse1, [impulse, momentum])

```
isform(impulse2, situation(X, Y, Period), Im = F * T,
        impulse(X, Y, Im, Period) & force(X, Y, F, Period) &
        duration(Period, T)).
```

relates(impulse2, [impulse, force, duration])

```
isform(momentum1, situation(X, T1), Mom = M * V,
        momentum(X, Mom, T1) &
        mass(X, M, T1) &
        vel(X, V, T1)     )
```

relates(momentum1, [momentum, mass, vel])



PREPARES

prepare(impulse1, Q, impulse, impulse(X, Y, Im, Per), situation(X, Y, Per, T1, T2))

← period(Per, T1, T2).

prepare(impulse2, Q, impulse, impulse(X, Y, Im, Per), situation(X, Y, Per)).

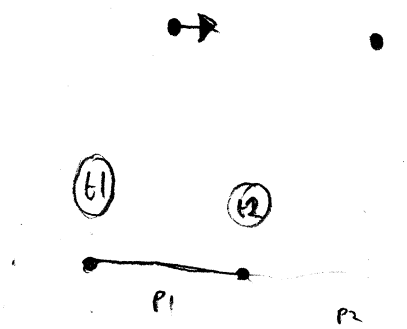
prepare(momentum1, Q, momentum, momentum(X, Mom, T1), situation(X, T1))

~ mass(X, M, T1) ~

~ vel(X, V, T1) ~

CRASH

2)



particle(truck1)
particle(truck2)

vel(truck1, v1, t1) ← magnitude(v1, 7, ms-1)
vel(truck2, zero, t1)

mass(truck1, m1) — magnitude(m1, 1200, kg)
mass(truck2, m2) — magnitude(m2, 1600, kg)

"coupled"

composite(cmp1)
part_of(cmp1, truck1)
part_of(cmp2, truck2)
exists(cmp1, p2)

vel(cmp1, v2,

period(p1, t1, t2)

"coupled"

∀v (vel(truck1, v, time) ⟺ vel(truck, v, time))

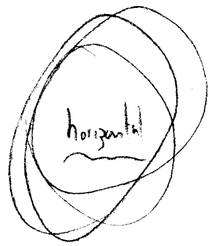
vel(truck1, v2, t2)
vel(truck2, v2, t2)

{ system over p1 = {truck1, truck2} }

given(v1)
(zero)
given(m1)
given(m2)
sought(v2)

"system"

CRASH



③

particle (gun) ?

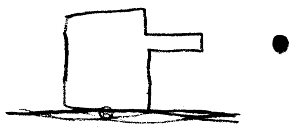
particle (shell)

mass (gun, m1) measure (m1, 750, kg)

mass (shell, m2) measure (m2, 12, kg)

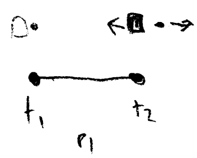
vel(gun, zero, t1)

vel(shell, zero, t1) ?

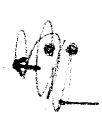


vel(gun, v1, t2) magnitude (v1, 3, ms-1)

vel(shell, v2, t2)



impulse(shell, gun, i1, p1)



given(m1)

given(m2)

given(v1) {zero}

idealising a system by enforcing constraints which allow elements to be ignored/removed.

sought(v2)

sought(i1)

m1 * zero + m2 * zero = m1 * v1 + m2 * v2 (conmom)

simplify this step.

requiring macros as inference?

i1 = m1 * v1 - m1 * zero (impulse1)

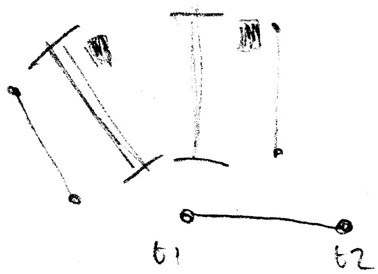
impulse of X on Y

Total impulse affecting Y

η?

Wild

motion(object, Start, Finish, T1, T2, motion)



collision(X, Y, Per) → ∃ I (impulse(X, Y, I, Per) &

impulse(Y, X, -I, Per))

impulse(X, Y, f(X, Y, Per), Per) ← collision(X, Y, Per).

-f(...) ←

Collision

Period

2 (or more) motions

Isform(collision_situation(X, Y, Per) I1 = -I2,

← impulse(X, Y, I1, Per) &

impulse(Y, X, I2, Per)

v2

mom77 = m1 * v2 (mom1)

= mom77 - mom78 (impulse1)

mom78 = m1 * v1 (mom1)

i79 = - i80 (collision)

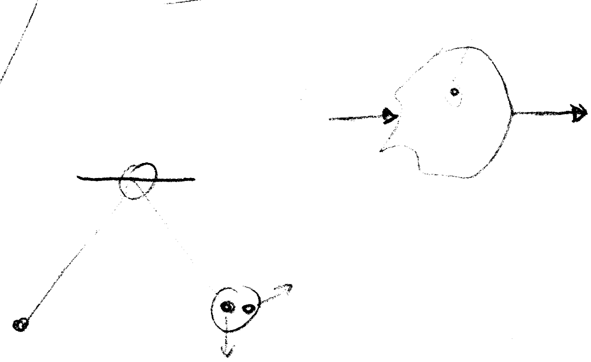
i80 = mom81 - mom82 (impulse1)

mom81 = m2 * v2 (mom1) mom82 = m2 * zero (mom1)

7 equations

System

member

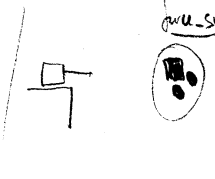


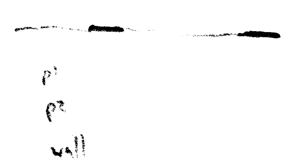
System = {collisions} + {objects} + {applied forces}

over some period

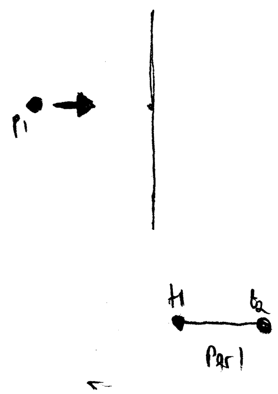
cf:

force_sum





⑦



surface(wall)

particle(p1)

vel(p1, v1, t1) ← magnitude(v1, 0.5, k_y) angle(—) ?

fixed(wall, Par1)

vel(p1, v2, t2) ← magnitude(v2, — angle(—).

period(p1, t1, t2)

collision(p1, wall, p1)

isform(restitution, situation(X, Y, Period, T1, T2), E = V/U,

coeff_restitution(X, Y, E, Period) &
relvel(X, Y, U, T1) &
relvel(Y, X, V, T2).

relates(restitution, [crest, vel])
?
=> constant

prepare

Forces revisited

force

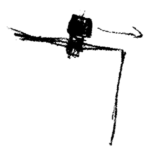
tension

friction

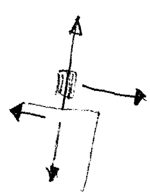
grav

reaction

tmoment



force1 = tension1 + reaction1 + friction1 + grav1



SYSTEMS

isform(consvmom , situation (System, T1 , T2), Msum1 = Msum2

momentum_sum (System, Msum1 , T1) &

momentum_sum (System , Msum2 , T2)

relates(consvmom, [momentum]).

{ vel (truck1 , v2, t2) }

prepare (consvmom , d, vel, vel(X, V, Time), situation(System

← ∃ System (collsys(system, T1, T2, objset, ~) &

member(x , objset) &

(coincident (Time , T1) v coincident(Time , T2)))

⇒ (m1*v1 + m2*zero) = (m1*v2) + (m2*v2)

⇒ mom92 + mom93 = mom94 + mom95

mom92 = ~

mom93 = ~

Again

cf with forces

Getting away with cantes in

drom tension

reaction

in statics over mass !

$F = MA$

$\Sigma F = \emptyset$

File: PULLEY

Connected motion Problems.

[PULL1]

Two particles of mass B and C are connected by a light string passing over a smooth pulley.
Find the acceleration of the particle of mass B.

[PULL2]

A particle of mass 4 kg rests on a smooth horizontal table.
It is connected by a light inextensible string passing over a smooth pulley at the edge of the table to a particle of mass 2 kg, which is hanging freely.
Find the acceleration of the system and the tension in the string.

(from Bostock and Chandler 1975)

[PULL3]

A particle of mass 5 kg rests on a rough horizontal table.
It is connected by a light inextensible string passing over a smooth pulley at the edge of the table to a particle of mass of 6 kg, which is hanging freely.
The coefficient of friction between the 5 kg mass and the table is 1/3.
Find the acceleration of the system and the tension in the string.

(from Bostock and Chandler 1975)

[PULL4]

Two particles of mass 3 kg and 4 kg are connected by a light inextensible string passing over a smooth fixed pulley.
The system is released from rest with the string taut and both particles at a height of 2 m above the ground.
Find the velocity of the 3 kg mass when the 4 kg mass reaches the ground.

(from Bostock and Chandler 1975)

[PULL5]

A string passing over a smooth fixed pulley supports at its two ends smooth moveable pulleys of masses 5 lb and 6 lb respectively.
Over the first of the moveable pulleys passes a string having masses of 3 lb and 4 lb at its ends, and over the second a string having masses of 2 lb and 3 lb at its ends.
Find the acceleration of the moveable pulleys and of each of the masses.

(from Humphrey 1930)

[PULL6]

Two particles of mass 3 kg and 5 kg are connected by a light inextensible string passing over a smooth pulley which is fixed to the ceiling of a lift.
Find the tension in the string when the system is moving freely

and the lift has a downward acceleration G ms-2.

(from Bostock and Chandler 1975)

[PULL7]

Two particles of masses M and 4M are connected by a light inextensible string which passes over a pulley of radius A.
The pulley is free to turn in a vertical plane without friction about a horizontal axis through its centre and the moment of inertia about this axis is MA^2.
The system is released from rest and the string does not slip on the pulley.
Find the accelerations of the particles and the distance each moves in time T.
Also find the tensions in the string.

(A-level exam: AEB)

[PULL8]

A particle of mass M1 is in contact with the smooth sloping face of a wedge which is itself standing on a smooth horizontal surface.
If the mass of the wedge is M2 and the sloping face of the wedge is inclined at an angle 30 degrees to the horizontal find the acceleration of the wedge in terms of M1 and M2.

(from Bostock and Chandler 1975)

[PULL9]

Particles of mass M and 2M are connected by a light string which passes over a pulley at the vertex of a wedge shaped block, one particle resting on each of the faces which are smooth.
The mass of the wedge being N, and the inclination of the faces to the horizontal being Alpha, find the acceleration of the wedge and the particles when the wedge is placed on a smooth horizontal table.

(from Humphrey 1930)

[PULL10]

A wedge of mass N, whose section ABC is a triangle right angled at A, is placed with the face BC on a smooth horizontal table.
The faces AB and AC are rough, the coefficient of friction being Mu.
Two masses M1 and M2, connected by a light inextensible string passing over a light frictionless pulley at A, rest on the faces AB and AC respectively.
M1 moves down AB with acceleration F1.
Find F1, and the acceleration F2 of the wedge.

(from Humphrey 1930)

```
/* PULLEY problem  -  for MECHO 10.1 */
```

This was actually output from chris!

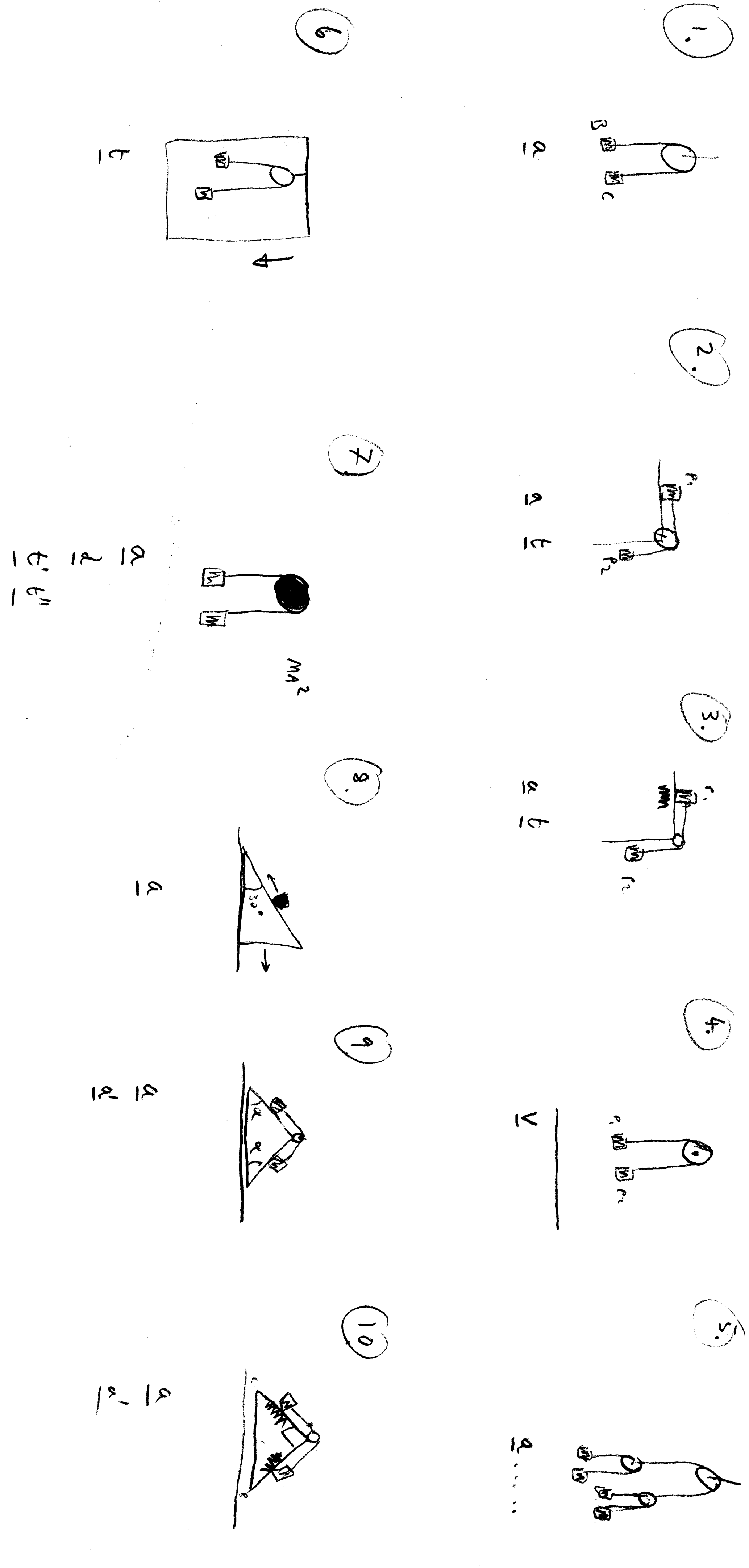
```
problem(pulley,'Simple pulley problem.\n%r==\n\n',[=====,4]).

isa(period,period0).
isa(particle,particle00).
isa(particle,particle01).
mass(particle00,mass1,period0).
mass(particle01,mass2,period0).
accel(particle00,acceleration0,90,period0).

measure(mass1,10,arbs).
measure(mass2,12,arbs).

given(mass1).
given(mass2).
sought(acceleration0).

cue pullsys_stan(sys,pulley0,string0,particle00,particle01,period0).
```

* See also WP50 (appendix) for more problems. (I will get these of archive when FMS comes back)

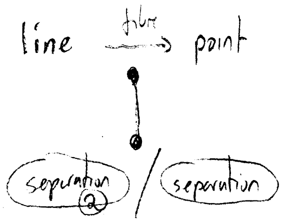
Very OLD problem

```
/* PULLEY PROBLEM WITH P1 ON TABLE AND P2 HANGING OVER EDGE. */


  steady :-      problem,

                 checklist( dbentry,[ isa(period,period1),
                                      isa(particle,p1),
                                      isa(particle,p2),
                                      mass(p1,bq,period1),
                                      mass(p2,cq,period1),
                                      accel(p1,a1,0,period1),
                                      incline(table,0,lefttop),
                                      solid(table),
                                      rough(table),
                                      coeff(table,mu),

                                      quantity(bq,b,arbs),
                                      quantity(cq,c,arbs)
                                    ]  ),

                 checklist( asserta,[ given(bq),
                                      given(cq),
                                      given(mu),
                                      sought(a1)
                                    ]  ),

                 nl,

                 cue(pullsys_maj(sys,pull,str,p1,0,p2,270,period1)),
                 cue(pathsys(table,lefttop,edge,hor,stline)),
                 cue(line_motion(p1,table,period1)).

  problem :- writef('\n\tPULLEY ON TABLE PROBLEM.\n\t%r\n\n',[-,24]).
```

Pulley problems

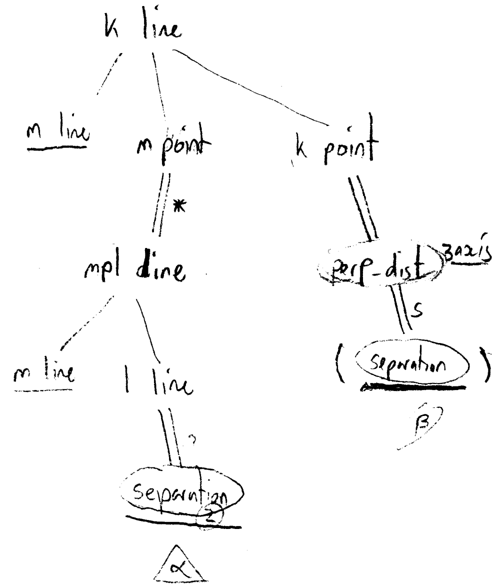


MOFI 1



on !!

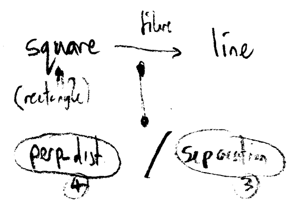
other meets !!

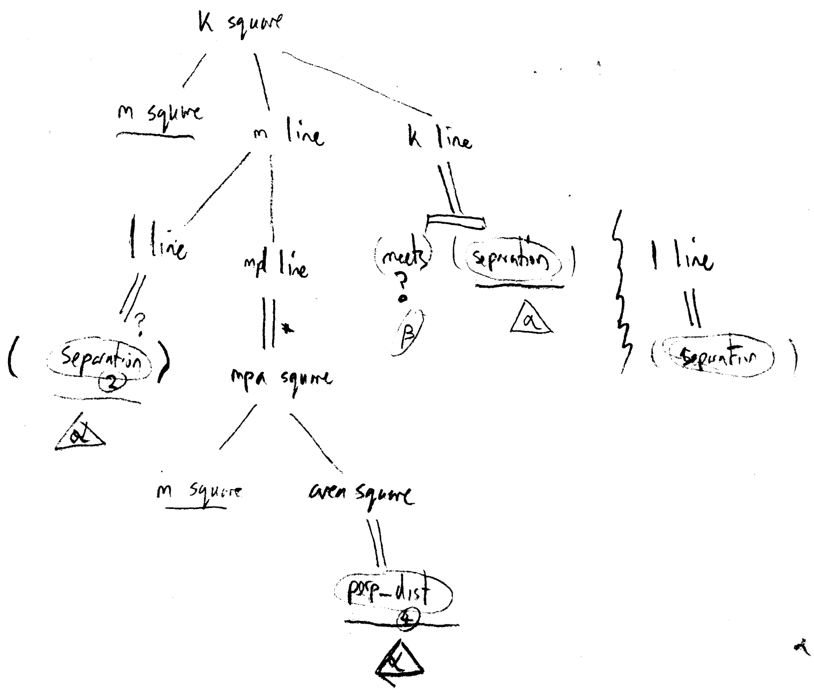


①

②

12

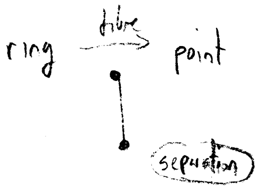


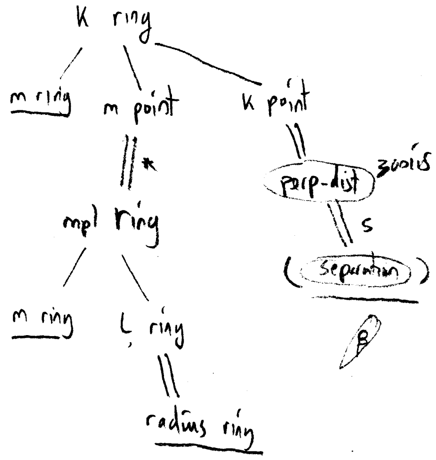


- perp-dist ⇔ separation ?

— Midpt discovery

MOFI 3

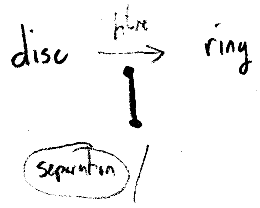


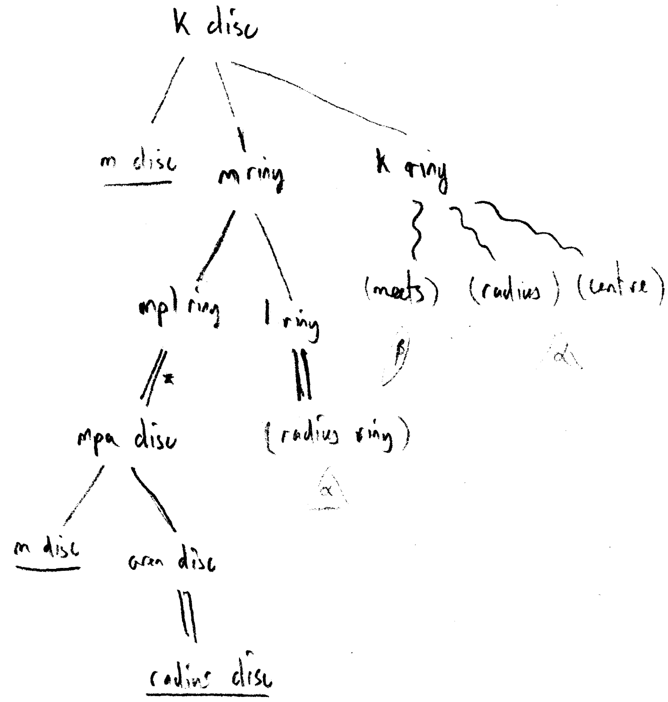


α) Inferring dimensions of objects

β) Inferring axis ~~meeting~~ interception and perp-dists from axis

MOF14

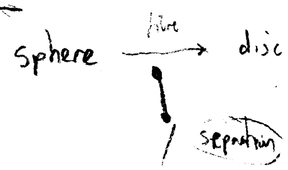


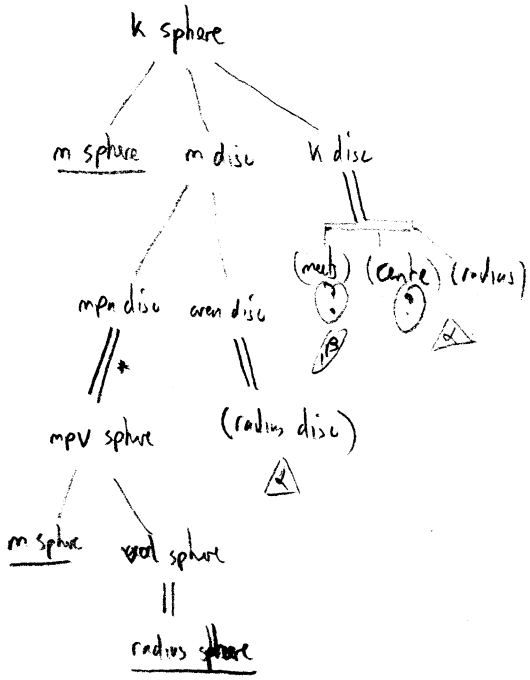


1)
2)
3)

checking axis meets at centre - relies on separation(origin, origin, v(Z, 0, 90))
+ ordering
ø

MOF15





1)
2)
3)

connection between sphere & disc - relies on separation(origin, C, v(ø, ø, 90))
+ ordering
origin

{is_delta}
IR

Mofi

current difficulties

1) Midpt problem

general) goals of the form: sep(pt, X, vec, time)

+ function properties?

- i) existence properties for points
- ii) function properties for <classes>
- iii) different preds

sep, over absolute space-points
sep₂ over all point like objects — space-points + line-points +

2) Vectors

- -evaluation vs- individual reps] can we canonicalise?
- partial specification
- relationship between vectors and their internal quantities
  re: Problem solving, {soughts - givens}

Mofi problems

Major

1) Vectors

- extracting bits properly
- getting definitions etc of contained symbolic quantities.

[TAKING THIS INTO ACCOUNT?] *

1 - nfrec2 merge_sep error ✓ (trail(sep, ...))

2 - Defn bodies (functional props vs ∃ ~)

3 - length/2 clash with evaluable ✓ (length → llength

- cartesian coord for line has midpt problem

Incidental

- Nature of cont_meas wrt creation of a new fibre
  - Object creation (note)
  - NOT constrained by cc flag except inasmuch as conditions on defn require creation

- checklist cc's on unused is_defn's is pretty weird.

① Line
② Square
③ Ring
④ Disc
⑤ Sphere

mofi1 - ok but for ridiculous
+ midpt thrash time waste

mofi2 - Blows up on midpt problem (nr fine fibre)

mofi3 - vecadd runs out of core during merge.

vecadd([a, theta2, ∅], [0,0,0], X)

From Bundy Hps[400,405] on April 2, 1981 at 2:51 PM
Lawrence

I take it you did not manage to sort out separation last night, since it still seems to blow up.

I haadded 3 more problems: the solid cylinder and cone and the hollow sphere. Mecho has proved all 10 problems at some time, although it is currently limited to 9. Adapting it to prove mofi4 screwed it for mofi3. I think the problem is with separation: it blows up with pages of algebra to tidy (and I mean PAGES!) and it gets a variable in vecadd.

All the problems create non-fatal errors when run.

(a) Error messages after fn_dbentry of on(.,.,.,.).

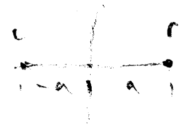
(b) Error messages on Types after go.

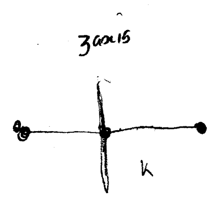
(c) Unnecessarily long separation outputs.

I shall leave you to fix these. Please do it with high priority so I can get back and finish off the mofi problems.

Alan

```
/* MOFI1.PRB : A moment of inertia problem */

problem(mofi1,'Radius of Gyration of a line\n\n',[]).

line(l1).
point(l).
point(r).
line_sys(l1,l,r).
separation(origin,l,-a,[0,0]).
separation(origin,r,a,[0,0]).
meets(zaxis,l1,origin).
mass(l1,m).

rad_of_gyr(l1,zaxis,k).

given(a).
given(m).
sought(k).
```





1) $m k^2 = \int d(r1) * mpl1 * r1^2$ [-a, a, r1] | moment_of_inertia

mass rofg

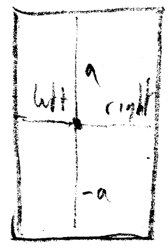
mass of fibre via hack rule + creation on mpl1

rofg of fibre via rule via perp dist rule

2) $m = 2a * mpl1$ | mass_per_area

(must use spherical polar at the moment?)

```
/* MOFI2.PRB : 2nd Moments of Inertia problem */

problem(mofi2,'Radius of Gyration of a square\n\n',[]).


square(sq).
line(top).
line(bot).
line(left).
line(right).
quad_sys(sq,top,bot,left,right).
perp_dist(origin,top,a,[90,0]).
perp_dist(origin,bot,-a,[90,0]).
perp_dist(origin,left,-a,[0,0]).
perp_dist(origin,right,a,[0,0]).
meets(yaxis,sq,yaxis).

mass(sq,m).

rad_of_gyr(sq,yaxis,k).

given(a).
given(m).
sought(k).

                              % radius of gyration of a line

rad_of_gyr(Lne,Ax,A/sqrt(3))
    <-- line_sys(Lne,Lend,Rend) &
        meets(Ax,Lne,Pt) &
        separation(Pt,Rend,A,[T,P]) &
        separation(Pt,Lend,-A,[T,P]).
```



) width

```
/* MOFI3.PRB : 3rd Moments of Inertia problem */

problem(mofi3,'Radius of Gyration of a ring\n\n',[]).


ring(ring1).
centre(ring1,origin).
radius(ring1,a).
meets(zaxis,ring1,origin).

mass(ring1,m).

rad_of_gyr(ring1,zaxis,k).

given(a).
given(m).
sought(k).
```

```
/* MOFI4.PRB : 4th Moments of Inertia problem */

problem(mofi4,'Radius of Gyration of a Disc\n\n',[]).


disc(disc1).
centre(disc1,origin).
radius(disc1,a).
meets(zaxis,disc1,origin).

mass(disc1,m).

rad_of_gyr(disc1,zaxis,k).

given(a).
given(m).
sought(k).


                              % The radius of gyration of a ring about a
                              % perpendicular axis through its centre is its radius

rad_of_gyr(Ring,Axis,R)
    <-- ring(Ring) & centre(Ring,C) &
        meets(Axis,Ring,C) & radius(Ring,R).
```

meets

needs sep~(origin, origin

due to offset

```
/* MOFI5.PRB : 5th Moments of Inertia problem */

problem(mofi5,'Radius of Gyration of a sphere\n\n',[]).


sphere(sphere1).
centre(sphere1,origin).
radius(sphere1,a).
meets(zaxis,sphere1,zaxis).

mass(sphere1,m).

rad_of_gyr(sphere1,zaxis,k).

given(a).
given(m).
sought(k).


                          % The radius of gyration of a disc about a
                          % perpendicular axis through its centre is its radius
                          % divided by root 2.

rad_of_gyr(Disc,Axis,R/sqrt(2))
    <-- disc(Disc) & centre(Disc,C) &
        meets(Axis,Disc,C) & radius(Disc,R).
```

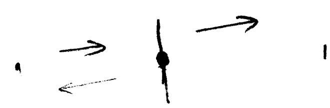
nab ↗ ?

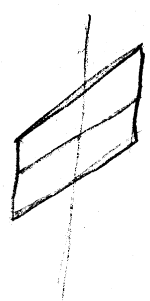
```
/* MOFI1.PRB : A moment of inertia problem */

problem(mofi1,'Radius of Gyration of a line\n\n',[]).


line(l1).
point(l).
point(r).
line_sys(l1,l,r).
separation(origin,l,-a,[0,0]).
separation(origin,r,a,[0,0]).
meets(yaxis,l1,origin).
mass(l1,m).

rad_of_gyr(l1,yaxis,k).

given(a).
given(m).
sought(k).
```

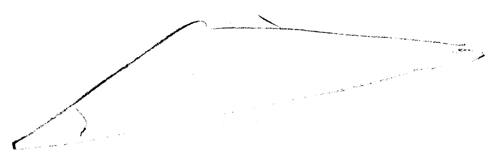
```
/* MOFI2.PRB : 2nd Moments of Inertia problem */

problem(mofi2,'Radius of Gyration of a rectangle\n\n',[]).


rectangle(rect).
line(top).
line(bot).
line(left).
line(right).
quad_sys(rect,top,bot,left,right).
perp_dist(origin,top,b,[90,0]).
perp_dist(origin,bot,-b,[90,0]).
perp_dist(origin,left,-a,[0,0]).
perp_dist(origin,right,a,[0,0]).
meets(yaxis,rect,yaxis).

mass(rect,m).

rad_of_gyr(rect,yaxis,k).

given(a).
given(m).
sought(k).

                              % radius of gyration of a line

rad_of_gyr(Lne,yaxis,A/sqrt(3))
    <-- line_sys(Lne,Lend,Rend) &
        meets(yaxis,Lne,Pt) &
        separation(Pt,Rend,A,[0,0]) &
        separation(Pt,Lend,-A,[0,0]).
```

```
/* MOFI3.PRB : 3rd Moment of Inertia problem */

problem(mofi3,'Radius of Gyration of an inclined line\n\n',[]).


line(l1).
point(l).
point(r).
line_sys(l1,l,r).
separation(origin,l,-a*cos(alpha),[alpha,0]).
separation(origin,r,a*cos(alpha),[alpha,0]).
meets(yaxis,l1,origin).
mass(l1,m).

rad_of_gyr(l1,yaxis,k).

given(a).
given(m).
sought(k).
```

```
/* MOFI4.PRB : 4th Moments of Inertia problem */

problem(mofi4,'Radius of Gyration of a parallelogram\n\n',[]).


parallelogram(pagrm).
line(top).
line(bot).
line(left).
line(right).
quad_sys(pagrm,top,bot,left,right).
perp_dist(origin,top,b*cos(alpha),[90+alpha,0]).
perp_dist(origin,bot,-b*cos(alpha),[90+alpha,0]).
perp_dist(origin,left,-a*cos(alpha),[0,0]).
perp_dist(origin,right,a*cos(alpha),[0,0]).
meets(yaxis,pagrm,yaxis).

mass(pagrm,m).

rad_of_gyr(pagrm,yaxis,k).

given(a).
given(m).
sought(k).

                                % radius of gyration of an inclined line

rad_of_gyr(Lne,Ax,A*sin(T2-T1)/sqrt(3))
     <-- line_sys(Lne,Lend,Rend) &
         meets(Ax,Lne,Pt) &
         incline(Ax,[T2,P],Pt) &
         separation(Pt,Rend,A,[T1,P]) &
         separation(Pt,Lend,-A,[T1,P]).
```







```
/* MOFI5.PRB : 5th Moments of Inertia problem */

problem(mofi5,'Radius of Gyration of a ring\n\n',[]).


ring(ring1).
centre(ring1,origin).
radius(ring1,a).
meets(zaxis,ring1,origin).

mass(ring1,m).

rad_of_gyr(ring1,zaxis,k).

given(a).
given(m).
sought(k).
```

```
/* MOFI6.PRB : 6th Moments of Inertia problem */

problem(mofi6,'Radius of Gyration of a Disc\n\n',[]).


disc(disc1).
centre(disc1,origin).
radius(disc1,a).
meets(zaxis,disc1,origin).

mass(disc1,m).

rad_of_gyr(disc1,zaxis,k).

given(a).
given(m).
sought(k).


                               % The radius of gyration of a ring about a
                               % perpendicular axis through its centre is its radius

rad_of_gyr(Ring,Axis,R)
     <-- ring(Ring) & centre(Ring,C) &
         meets(Axis,Ring,C) & radius(Ring,R).
```

```
/* MOFI7.PRB : 7th Moments of Inertia problem */

problem(mofi7,'Radius of Gyration of a sphere\n\n',[]).


sphere(sphere1).
centre(sphere1,origin).
radius(sphere1,a).
meets(zaxis,sphere1,zaxis).

mass(sphere1,m).

rad_of_gyr(sphere1,zaxis,k).

given(a).
given(m).
sought(k).


                              % The radius of gyration of a disc about a
                              % perpendicular axis through its centre is its radius
                              % divided by root 2.

rad_of_gyr(Disc,Axis,R/sqrt(2))
    <-- disc(Disc) & centre(Disc,C) &
        meets(Axis,Disc,C) & radius(Disc,R).
```

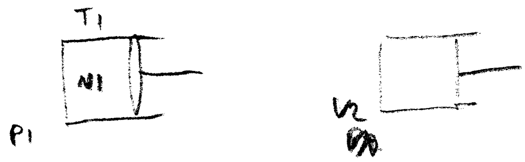
```
/* PR1.PRB : First Thermo Problem

                                                          Wouter
                                                          Updated: 11 December 81
*/


/* An ideal gas is contained in a container. The container is closed
by a piston. The volume of the gas is 2 litres and the and the
pressure is 120 kPa. By moving the piston slowly outward, the volume
is increased till 3 litres, while the temperature remains constant.
What is the new pressure of the gas?
*/
/*
contain(container1,gas1).
ideal_gas(gas1).
closed_by(container1,piston1).
piston(piston1).
volume_gas(gas1,volume1,time1).
measure(volume1,3,litres).
pressure_gas(gas1,pressure1,time1).
measure(pressure1,120,kPa).
move(piston1,outward,period).
period(period,time1,time2).
volume_gas(gas1,volume2,time2).
measure(volume2,3,litres).
temperature_gas(gas1,period).
constant(temp1,time1,time2).
pressure_gas(gas1,pressure2,time2).
sought(pressure2).
*/


state(state1).
state(state2).
system(observed_substance1,system1).
ideal_gas(gas1).
container(container1).

observed_substance(system1,gas1).
reversible(system1,st_change).
closed(system1).
state_change(system1,st_change,state1,state2).
% contains(container1,system1,state1).

volume(system1,vol1,state1).
measure(vol1,2,liter).
pressure(system1,pres1,state1).
measure(pres1,120,kPa).

volume(system1,vol2,state2).
measure(vol2,3,liter).
pressure(system1,pres2,state2).
temperature(system1,temp1,state1).
const(temp1,st_change,system1).

given(vol1).
given(vol2).
given(pres1).
sought(pres2).
```

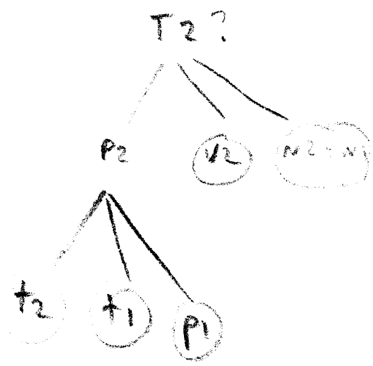
```
/* PR1B.PRB : Version 2 of prb1(dT instead of constant temp.

                                                    Wouter
                                                    Updated: 11 December 81
*/



/* An ideal gas is contained in a container. The container is closed
by a piston. The volume of the gas is 2 litres and the and the
pressure is 120 kPa. By moving the piston slowly outward, the volume
is increased till 3 litres, while the temperature remains constant.
What is the new pressure of the gas?
*/
/*
contain(container1,gas1).
ideal_gas(gas1).
closed_by(container1,piston1).
piston(piston1).
volume_gas(gas1,volume1,time1).
measure(volume1,3,litres).
pressure_gas(gas1,pressure1,time1).
 easure(pressure1,120,kPa).
move(piston1,outward,period).
period(period,time1,time2).
volume_gas(gas1,volume2,time2).
measure(volume2,3,litres).
temperature_gas(gas1,period).
constant(temp1,time1,time2).
pressure_gas(gas1,pressure2,time2).
sought(pressure2).
*/


state(state1).
state(state2).
system(observed_substance1,system1).
ideal_gas(gas1).
container(container1).

observed_substance(system1,gas1).
 eversible(system1,st_change).
closed(system1).
state_change(system1,st_change,state1,state2).
% contains(container1,system1,state1).

volume(system1,vol1,state1).
measure(vol1,2,liter).
pressure(system1,pres1,state1).
measure(pres1,120,kPa).

volume(system1,vol2,state2).
measure(vol2,3,liter).
pressure(system1,pres2,state2).
temperature(system1,t1,state1).          % Seems superfluous, but it needs
                                         % to know about a temperature !
dT(st_change,dT).
measure(dT,0,celsius).
given(vol1).
given(vol2).
```

```
given(pres1).
sought(pres2).
given(dT).
```

```
/* PR2.PRB : Second thermo problem

                                                    Wouter
                                                    Updated: 15 December 81
*/


/* A cylinder is closed by a frictionless piston and contains
argon. It expands adiabatic from a pressure of 300 kPa to 100 kPa.
The initial temperature is 590 kelvin. What is the final temperature?.
*/

/*
closed_by(container1,piston1).
piston(piston1).
frictionless(piston1).
contain(container1,gas1).
argon(gas1).
expansion(adiabatic,period).
period(period,time1,time2).
pressure_gas(gas1,p1,time1).
measure(p1,300,kPa).
pressure_gas(gas1,p2,time2).
measure(p2,100,kPa).
temp_gas(gas1,t1,time1).
measure(t1,590,kelvin).
temp_gas(gas1,t2,time2).
sought(t2).
*/


state(state1).
state(state2).
argon(gas1).
ideal_gas(gas1).

observed_substance(system1,gas1).
system(observed_substance1,system1).
reversible(system1,st_change).
closed(system1).
adiabatic(system1,st_change).
state_change(system1,st_change,state1,state2).
pressure(system1,pres1,state1).
pressure(system1,pres2,state2).
temperature(system1,temp1,state1).
temperature(system1,temp2,state2).
given(pres1).
given(pres2).
given(temp1).
sought(temp2).
```

```
/* PR3.PRB : Third thermo problem.

                                                        Wouter
                                                        Updated: 11 December 81
*/

/* A cylinder is closed by a frictionless piston. It contains 10 mol
helium-gas. The pressure is 120 kPa and the temperature 40 Celsius.
There is an adiabatic expansion till a volume of 300 dm^3. Compute
the new temperature.
*/

/*
closed_by(container1,piston1).
piston(piston1).
frictionless(piston1).
contain(container1,gas1).
helium(gas1).
amount_of_gas(gas1,n1,time1).
measure(n1,10,mol).
pressure_gas(gas1,p1,time1).
measure(p1,120,kPa).
temperature_gas(gas1,t1,time1).
measure(t1,40,celsius).
period(period,time1,time2).
expansion(adiabatic,period).
volume_gas(gas1,v2,time2).
measure(v2,300,'dm^3').
temperature_gas(gas1,t2,time2).
sought(t2).

*/
```






```
state(state1).
state(state2).
system(observed_substance1,system1).
state_change(system1,st_change,state1,state2).
observed_substance(system1,gas1).
reversible(system1,st_change).
closed(system1).
adiabatic(system1,st_change).
helium(gas1).
ideal_gas(gas1).
amount_of_substance(system1,n1,state1).
pressure(system1,p1,state1).
temperature(system1,t1,state1).
volume(system1,v2,state2).
temperature(system1,t2,state2).
given(n1).
given(p1).
given(t1).
given(v2).
sought(t2).
```

```
/* PR4.PRB : Thermo problem 4 (Number 32 from progress report).

                                                    Wouter
                                                    Updated: 14 December 81
*/

/* A closed container contains 1 mol ideal gas. An increase in temperature
with 30 Celsius results in an increase of the pressure with 100 kPa.
What is the volume of the container.
*/

/*
container(container1).
closed(container1).                 % different meaning of closed ?
contain(container1,gas1).
ideal_gas(gas1).
amount_of_gas(gas1,n,time1).
measure(n,1,mol).
period(period,time1,time2).
increase(temperature_gas,incr1,period).
measure(incr1,30,celsius).
increase(pressure_gas,incr2,period).
measure(incr2,100,kPa).
volume_container(v_cont,container1).
sought(v_cont).
*/

state(state1).
state(state2).
system(observed_substance1,system1).
state_change(system1,st_change,state1,state2).
observed_substance(system1,gas1).
reversible(system1,st_change).
closed(system1).
ideal_gas(gas1).
amount_of_substance(system1,n,state1).
dT(st_change,dT).
dP(st_change,dP).
volume(system1,vol,state1).         % Seems superfluous but marple seems
                                    % to need a vol of a state.
                                    % If this statement comes AFTER the
                                    % next const then marple takes the
                                    % wrong definition(vol,X,Y) !!!
const(vol,st_change,system1).
given(n).
given(dT).
given(dP).
sought(vol).
```

```
/* PR5.PRB : Thermo problem 5. (Number 33 from progress report).

                                                    Wouter
                                                    Updated: 11 December 81
*/

/*      An amount of argon is adiabatic expanded till three times
        it's original volume. What will the new temperature be if
        it is given that the temperature decreases by 400 Kelvin.
*/
/*
amount_of_gas(gas1,n1,time1).
argon(gas1).
expansion(adiabatic,period).
period(period,time1,time2).
volume_gas(gas1,vol,time1).
volume_gas(gas1,3*vol,time2).
temperature_gas(gas1,temp,time2).
decrease(temperature_gas,decr,period).
measure(decr,400,kelvin).
sought(temp).
*/

state(state1).
state(state2).
observed_substance(system1,gas1).
system(observed_substance1,system1).
state_change(system1,st_change,state1,state2).
reversible(system1,st_change).
closed(system1).
argon(gas1).
ideal_gas(gas1).
adiabatic(system1,st_change).
volume(system1,  vol,state1).
volume(system1,3*vol,state2).
dT(st_change,dT).
temperature(system1,temp,state2).
given(dT).
sought(temp).
```

```
/* PR6.PRB : Thermo problem 6 (Mettes & Pilot blz. 111).

                                                         Wouter
                                                         Updated: 16 December 81
*/

/*      1 mol ideal gas is contained in a cylinder under a weightless
        and frictionless piston. The cylinder and the piston are
        thermically isolated. The temperature of the gas is 27 Celsius,
        the pressure is 1 atm.
        Suddenly one places a weight on the piston; the weight is
        equal to a pressure of 7/3 atm.
        What is the temperature of the gas after the process ?
        Cv = 2,5 R ; Cp = 3,5 R.
*/
/*
amount_of_gas(gas1,n1,time1).
ideal_gas(gas1).
cylinder(cylinder1).
contain(cylinder1,gas1).
closed_by(cylinder1,piston1).
weightless(piston1).
 frictionless(piston1).
isolated(cylinder1).
isolated(piston1).
temperature_gas(gas1,temp1,time1).
measure(temp1,27,celsius).
pressure_gas(gas1,p1,time1).
measure(p1,1,atm).
place(weight,on,piston1,suddenly).
measure(pressure1,7/2,atm).             % ???????
equal(weight,pressure1).                % ???????????
temperature_gas(gas1,temp2,time2).
process(process1,time1,time2).
sought(temp2).
relate(cv,r,cv=(5/2)*r).
relate(cp,r,cp=(7/2)*r).
*/

state(state1).
state(state2).
observed_substance(system1,gas1).
system(observed_substance1,system1).
environment(environment,system1).
state_change(system1,st_change,state1,state2).
closed(system1).
isolated(system1).
adiabatic(system1,st_change).           % Can be infered from previous
                                        % statement !
irreversible(system1,st_change).
ideal_gas(gas1).
amount_of_substance(system1,n,state1).
temperature(system1,temp1,state1).
pressure(system1,p1,state1).
pressure(system1,p2,state2).
const(p2,st_change,environment).        % This one changed with the
                                        % previous one doesn't work !
                                        % wrong definition(p2,X,Y).
dP(st_change,dP).
temperature(system1,temp2,state2).
```

```
cv_gas(gas1,cp).
cv_gas(gas1,cv).
given(n).
given(temp1).
given(p1).
given(dP).
given(cv).
given(cp).
sought(temp2).
deltaQ(st_change,dQ).                        % Has to be inferred
given(dQ).                                   % from 'adiabatic'
```

```
/* PR7.PRB : Thermo problem 7. (Number 35 from progress report).

                                                    Wouter
                                                    Updated: 15 December 81
*/

/*      A container is closed by a piston and contains 1 mol ideal
        gas. By heating the gas the temperature raises 30 Celsius
        and the pressure 20 kPa. During the heating the piston is
        fixed.
        Next the piston is released and is moved outwards till the
        volume is increased by 5 dm^3. This process is adiabatic.
        Compute the new pressure, when it is given that the temperature
        decreased by 100 Celsius.
*/

state(state1).
state(state2).
state(state3).
observed_substance(system1,gas1).
system(obs_subst1,system1).
ideal_gas(gas1).
 losed(system1).
state_change(system1,st_change1,state1,state2).
state_change(system1,st_change2,state2,state3).
adiabatic(system1,st_change2).
reversible(system1,st_change2).
amount_of_substance(system1,n1,state1).
measure(n1,1,mol).
volume(system1,vol1,state1).
const(vol1,st_change1,system1).
dP(st_change1,dP_1).
measure(dP_1,20,kPa).
dT(st_change1,dT_1).
measure(dT_1,30,celsius).
dV(st_change2,dV_2).
measure(dV_2,5,'dm^3').
dT(st_change2,dT_2).
measure(dT_2,-100,celsius).
pressure(system1,pres3,state3).
given(dP_1).
 iven(dT_1).
given(dV_2).
given(dT_2).
given(n1).
sought(pres3).
```

31 JAN 80

Mecho Problem Areas

Pulleys

Roller Coaster

Constant Acceleration

Levers

other Statics

Springs

- Projectiles
- Impulses / collisions

Relative velocity

(Moments of Inertia)

Hydrostatics

Thermodynamics

[May 80]

Mechanics problems to be solved

Recover old problems — Pulleys
Simple motion
DeKleer motion-on-complex paths ✓

Work problems

Do All of Novaks examples (jump up statics)

Do All of Larkins+ problems

Thermodynamics (Simon & Bhasker) ✓

elastic collisions (for fun)

Problems

<u>People</u>

(Bobrow)		
Charniak		rate problems.
Novak	(20)	Lever type statics problems.
De Kleer	(3)	Motion on smooth paths.
Larkin		statics problems.
		some motion problems

Mecho Problem Areas
==================

1) Forces in Equilibrium.

Statics problems involving just resolution of forces. Includes masses, strings and rods, with frictional and contact forces with surfaces.

File: FORCES
Problems: FORC1-FORC10

2) Rigid Body Equilibrium.

Statics problems with lever principles involving turning moments and resolution of forces.

File: NOVAK
Problems: NOVK1-NOVK25 (take a selection)

3) Connected Motion.

Motion problems involving pulleys, wedges, and particles on surfaces.

File: PULLEY
Problems: PULL1-PULL10

4) Constant Acceleration.

Distance/rate/time problems involving objects in motion vertically or horizontally.

File: MOTION
Problems: MOTN1-MOTN10

5) Relative Motion.

Problems involving interception (etc) of several objects in motion, including effects due to currents, winds etc.

File: RELTIV
Problems: RELT1-RELT10

6) Projectiles.

Particles in flight with both vertical and horizontal acceleration/velocity components.

File: PROJCT
Problems: PROJ1-PROJ10

7) Motion on Complex Paths.

The deKleer problems.

8) Collision and Impact.

Elastic and inelastic collisions involving momentum, impulses, and the Law of Restitution.

File: CRASH
Problems: CRSH1-CRSH10

9) Springs and Energy.

Springs and elastic strings, using force and energy principles.

File: SPRING
Problems: SRNG1-SRNG10

10) Work and Power.

Objects in simple motion under forces.

File: WORK
Problems: WORK1-WORK10

11) Moments of Inertia.

Alan's "fibre" work.

File: FORCES

Forces in Equilibrium Problems.
===============================

[FORC1]

A small object of weight 10 N rests in equilibrium on a rough plane inclined at 30 degrees to the horizontal.
Calculate the magnitude of the frictional force.

(from Bostock and Chandler 1975)

[FORC2]

A particle of weight W rests on a smooth plane inclined at 30 degrees to the horizontal and is held in equilibrium by a string inclined at 30 degrees to the plane.
Find, in terms of W, the tension in the string.

(from Bostock and Chandler 1975)

[FORC3]

A weight W is suspended by two ropes which make 30 degrees and 60 degrees with the horizontal.
If the tension in the first rope is 20 N, find the tension in the other and the value of W.

(from Bostock and Chandler 1975)

[FORC4]

A sledge whose weight is 4000 N is pulled at constant speed along level ground by a rope held at 30 degrees to the ground.
If mu = 1/4, find the pulling force required.

(from Bostock and Chandler 1975)

[FORC5]

A uniform rod AB of weight W rests in equilibrium with the end A in contact with a smooth vertical wall and the end B in contact with a smooth plane inclined at 45 degrees to the wall.
Find the reactions at A and B in terms of W.

(from Bostock and Chandler 1975)

[FORC6]

A uniform rod AB of weight 120 N and length 4 m is in equilibrium with the end A on a rough horizontal floor at an angle of theta degrees to the horizontal.
The rod is resting against a smooth fixed peg at the point C in the rod such that AC = 3 m.
Find mu, the coefficient of friction between the rod and the floor.

(A-level exam adaptation: AEB)

[FORC7]

A uniform rod AB of length 2a and mass M is freely hinged to a fixed point A and is held inclined at an angle of 30 degrees to the horizontal (with B above A) by a light inextensible string attached to the mid-point of the rod and to a point C vertically above A, where AC = a.
Find the magnitude and direction of the reaction at A.
Find also the tension in the string.

(A-level exam: U of L)

File: NOVAK

Rigid Body Equilibrium Problems

Levers etc.
Taken from Novak 1976.

[NOVK1]

A lever 10 ft long is pinned at its left end.
The lever is supported by a spring with a constant
of 40 lb/ft.
The spring is attached 6 ft from the left end of the lever.
A weight of 20 lb is attached at the other end of the lever.
The weight of the lever is 8 lb.
How much is the spring stretched?

[NOVK2]

Where must a weight be hung on a pole, of negligible weight,
so that the boy at one end supports 1/3 as much as the man at
the other end?

(from Schaum 1961)

[NOVK3]

A scaffold 10 ft long is supported by ropes attached at each end.
The scaffold weighs 100 lb.
One painter weighing 150 lb stands on the scaffold 4 ft from one
end while a second painter weighing 175 lb stands on the scaffold
2 ft from the other end.
What is the tension on each of the ropes supporting the scaffold?

(from Dull, Metcalfe and Williams 1964)

[NOVK4]

A horizontal uniform bar 10 m long is supported by two ropes
attached at its ends.
The rope on the left end makes an angle of 45 degrees with the
horizontal, while the rope on the right end makes an angle of
60 degrees with the horizontal.
A weight of 100 nt is attached 2 m from the right end of the bar.
What is the weight of the bar?

[NOVK5]

A uniform scaffold 12 ft long and weighing 100 lb is supported
horizontally by two vertical ropes hung from its ends.
Find the tension in each rope when a 180 lb painter stands
4 ft from one end.

(from Schaum 1961)

[NOVK6]

A uniform bar A-B is 100 cm long and weighs 50 lb.
The bar is to be supported at ends A and B.
An upward force of 40 lb is applied 80 cm from A.

Compute the forces on the supports.

(from Schaum 1961)

[NOVK7]

A uniform pole 20 ft long and weighing 30 lb is supported by
a boy 3 ft from one end and a man 6 ft from the other end.
At what point must a 150 lb weight be attached so that the man
supports twice as much as the boy?

(from Schaum 1961)

[NOVK8]

The foot of a ladder rests against a vertical wall and on a
horizontal floor.
The top of the ladder is supported from the wall by a horizontal
rope 30 ft long.
The ladder is 50 ft long, weighs 100 lb with its center of gravity
20 ft from the foot, and a 150 lb man is 10 ft from the top.
Determine the tension in the rope.

(from Schaum 1961)

[NOVK9]

The hinges of a door weighing 20 lb are 12 ft apart, and the door
is 3 ft wide.
The weight of the door is supported by the upper hinge.
Determine the forces exerted on the door at the hinges.

(from Schaum 1961)

[NOVK10]

A steel beam of uniform cross section weighs 250 megadynes.
If it is 500 cm long, what force is needed to lift one end
of it?

(from Dull, Metcalfe and Williams 1964)

[NOVK11]

A tapering wooden telegraph pole is 15 ft long and its center
of gravity is 6 ft from one end.
It weighs 400 lb.
What force is required to lift each end?

(from Dull, Metcalfe and Williams 1964)

[NOVK12]

A bar 4 ft long weighs 400 nt.
Its center of gravity is 1.5 m from one end.
If a weight of 300 nt is attached at the heavy end and a weight
of 500 nt is attached at the light end, what are the magnitude,
direction, and point of application of the equilibrant?

(from Dull, Metcalfe and Williams 1964)

[NOVK13]

A bridge 60 ft long is supported by a pier at each end.
The bridge weighs 50 tons.
If a load of 7.5 tons is located 15 ft from one end, what load does each pier support?

(from Dull, Metcalfe and Williams 1964)

[NOVK14]

A bar 5 m long has its center of gravity 1.5 m from the heavy end.
If it is placed on the edge of a block 1.5 m from the light end and a weight of 750 nt is added to the light end, it will be balanced.
What is the weight of the bar?

(from Dull, Metcalfe and Williams 1964)

[NOVK15]

A uniform bar 25 m long weighs 10000 nt.
From end A a weight of 2500 nt is hung.
At B, the other end of the pole, there is a weight of 3500 nt.
An upward force of 3000 nt is exerted 4 m from B, while an upward force of 4000 nt is exerted 8 m from A.
Determine the magnitude, direction, and point of application of the equilibrant.

(from Dull, Metcalfe and Williams 1964)

[NOVK16]

A beam 4 m long is supported at both ends.
A weight of 500 nt is attached 1 m from end A, a weight of 800 nt is attached 1.5 m from end A, and a weight of 300 nt is attached 1.25 m from end B.
Calculate the forces exerted by the supports at each end of the beam.

(from Dull, Metcalfe and Williams 1964)

[NOVK17]

Paul and Henry carry a sack weighing 600 nt on a pole between them.
If the pole is 2 m long and the load is .5 m from Paul, what force does each boy exert?

(from Dull, Metcalfe and Williams 1964)

[NOVK18]

A bridge is 80 ft long.
What force must the pier at each end of the bridge exert to support an automobile weighing 2 tons which is 30 ft from one end of the bridge?

(from Dull, Metcalfe and Williams 1964)

[NOVK19]

A painter weighing 900 nt stands on a plank 3 m long, which is supported at each end by a stepladder.
If he stands 1 m from one end of the plank, what force is exerted by each stepladder?

(from Dull, Metcalfe and Williams 1964)

[NOVK20]

Two boys, weighing 100 lb and 125 lb respectively, wish to balance on a seesaw.
If the 100 lb boy sits 5 ft from the center, how far from the center must the 125 lb boy sit?

(from Dull, Metcalfe and Williams 1964)

Additional problems considered by Novak but not solved.

[NOVK21]

A uniform steel meter bar rests on two scales at its ends.
The bar weighs 4 lb.
Find the readings on the scales.

(from Halliday and Resnick 1967)

[NOVK22]

A 60 ft ladder weighing 100 lb rests against a wall at a point 48 ft above the ground.
The center of gravity of the ladder is one third the way up.
A 160 lb man climbs halfway up the ladder.
Assuming that the wall is frictionless, find the forces exerted by the system on the ground and the wall.

(from Halliday and Resnick 1967)

[NOVK23]

A uniform beam is hinged at the wall.
A wire connected to the wall a distance D above the hinge is attached to the other end of the beam.
The beam makes an angle of 30 deg with the horizontal when a weight W1 is hung from a string fastened to the end of the beam.
If the beam has a weight W2 and a length L, find the tension in the wire and the forces exerted by the hinge on the beam.

(from Halliday and Resnick 1967)

[NOVK24]

A door 7 ft high and 3 ft wide weighs 60 lb.
A hinge 1 ft from the top and another 1 ft from the bottom each support half the doors weight.
Assume that the center of gravity is at the geometrical center of the door and determine the horizontal and vertical force components exerted by each hinge on the door.

(from Halliday and Resnick 1967)

[NOVK25]

An automobile weighing 3000 lb has a wheel base of 120 in.
Its center of gravity is located 70 in behind the front axle.

Determine the force exerted on each of the front wheels (assumed the same) and the force exerted on each of the back wheels (assumed the same) by the level ground.

(from Halliday and Resnick 1967)

File: PULLEY

Connected motion Problems.

[PULL1]

Two particles of mass B and C are connected by a light string passing over a smooth pulley.
Find the acceleration of the particle of mass B.

[PULL2]

A particle of mass 4 kg rests on a smooth horizontal table.
It is connected by a light inextensible string passing over a smooth pulley at the edge of the table to a particle of mass 2 kg, which is hanging freely.
Find the acceleration of the system and the tension in the string.

(from Bostock and Chandler 1975)

[PULL3]

A particle of mass 5 kg rests on a rough horizontal table.
It is connected by a light inextensible string passing over a smooth pulley at the edge of the table, to a particle of mass of 6 kg, which is hanging freely.
The coefficient of friction between the 5 kg mass and the table is 1/3
Find the acceleration of the system and the tension in the string.

(from Bostock and Chandler 1975)

[PULL4]

Two particles of mass 3 kg and 4 kg are connected by a light inextensible string passing over a smooth fixed pulley.
The system is released from rest with the string taut and both particles at a height of 2 m above the ground.
Find the velocity of the 3 kg mass when the 4 kg mass reaches the ground.

(from Bostock and Chandler 1975)

[PULL5]

A string passing over a smooth fixed pulley supports at its two ends smooth moveable pulleys of masses 5 lb and 6 lb respectively.
Over the first of the moveable pulleys passes a string having masses of 3 lb and 4 lb at its ends, and over the second a string having masses of 2 lb and 3 lb at its ends.
Find the acceleration of the moveable pulleys and of each of the masses.

(from Humphrey 1930)

[PULL6]

Two particles of mass 3 kg and 5 kg are connected by a light inextensible string passing over a smooth pulley which is fixed to the ceiling of a lift.
Find the tension in the string when the system is moving freely

and the lift has a downward acceleration G ms-2.

(from Bostock and Chandler 1975)

[PULL7]

Two particles of masses M and 4M are connected by a light inextensible string which passes over a pulley of radius A.
The pulley is free to turn in a vertical plane without friction about a horizontal axis through its centre and the moment of inertia about this axis is MA^2.
The system is released from rest and the string does not slip on the pulley.
Find the accelerations of the particles and the distance each moves in time T.
Also find the tensions in the string.

(A-level exam: AEB)

[PULL8]

A particle of mass M2 is in contact with the smooth sloping face of a wedge which is itself standing on a smooth horizontal surface.
If the mass of the wedge is M2 and the sloping face of the wedge is inclined at an angle 30 degrees to the horizontal find the acceleration of the wedge in terms of M1 and M2.

(from Bostock and Chandler 1975)

[PULL9]

Particles of mass M and 2M are connected by a light string which passes over a pulley at the vertex of a wedge shaped block, one particle resting on each of the faces which are smooth.
The mass of the wedge being N, and the inclination of the faces to the horizontal being Alpha, find the acceleration of the wedge and the particles when the wedge is placed on a smooth horizontal table.

(from Humphrey 1930)

[PULL10]

A wedge of mass N, whose section ABC is a triangle right angled at A, is placed with the face BC on a smooth horizontal table.
The faces AB and AC are rough, the coefficient of friction being Mu.
Two masses M1 and M2, connected by a light inextensible string passin over a light frictionless pulley at A, rest on the faces AB and AC respectively.
M1 moves down AB with acceleration F1.
Find F1, and the acceleration F2 of the wedge.

(from Humphrey 1930)

File: MOTION

Constant Acceleration Problems.

[MOTN1]

A stone is dropped from a cliff 100 m above the sea.
Find the speed with which it hits the sea.

(from Bostock and Chandler 1975)

[MOTN2]

A ball is thrown vertically upward to a height of 10 m.
Find the time taken to reach this height and the initial speed of the ball.

(from Bostock and Chandler 1975)

[MOTN3]

A stone is projected vertically upward with a speed of 21 ms-1.
Find the distance travelled by the stone in the first 3 s of its motion.

(from Bostock and Chandler 1975)

[MOTN4]

A ball is thrown vertically upward with a speed of 15 ms-1 from a point which is 0.7 m above ground level.
Find the speed with which the ball hits the ground.

(from Bostock and Chandler 1975)

[MOTN5]

A stone is dropped from the top of a tower.
In the last second of its motion it falls through a distance which is 1/5 of the height of the tower.
Find the height of the tower.

(from Bostock and Chandler 1975)

[MOTN6]

A stone is dropped from the top of a building 20 m high.
A second stone is dropped from a point half-way up the same building.
Find the time that should elapse between the release of the two stones if they are to reach the ground at the same time.

(from Bostock and Chandler 1975)

[MOTN7]

A particle which is moving in a straight line with constant acceleration takes 3 s and 5 s to cover two succesive distances of 1 m.
Find the acceleration.

(from Bostock and Chandler 1975)

[MOTN8]

A toy train is moving along a straight length of track.
It accelerates uniformly from rest to a velocity of 0.5 ms-1 and maintains this velocity for a time before decelerating uniformly to rest again.
If the time taken for this journey is 2 seconds and it moves a distance of 0.8 m along the track, find the time for which the speed of the train is uniform.

(from Bostock and Chandler 1975)

[MOTN9]

A bus moves away from rest at a bus stop with an acceleration of 1 ms-2.
As the bus starts to move a man who is 4 m behind the stop runs with a constant speed after the bus.
If he just manages to catch the bus, find his speed.

(from Bostock and Chandler 1975)

[MOTN10]

A road runs due north through a point A.
A lorry travelling north at a constant speed of 54 km/h passes A, and immediately a car leaves A moving from rest with a constant acceleration of 2 ms-2.
Find the distance from A (in meters) where the car overtakes the lorry and the time (in seconds) for this to occur.

(A-level exam: AEB)

(pwt)

File: RELTIV

Relative Motion Problems.

[RELT1]

The driver of a car travelling due East on a straight road at 40 kmh-1 is watching a train moving due North at 75 kmh-1. What is the apparent speed and direction of motion of the train?

(from Bostock and Chandler 1975)

[RELT2]

A steamship is travelling north at the rate of 10 mph, and there is a north-east wind blowing at the rate of 20 mph.
In what direction will the smoke from the funnel appear to move to an observer on the ship?

(from Humphrey 1930)

[RELT3]

A man with a boat wishes to cross a stream of width A from a point O to a point P on the opposite bank at a distance B downstream.
If the stream flows with velocity U, what is the least speed with which the man must be able to row through still water if he is to reach P?
In what direction must he row, and how long will he take?

(from Palmer and Snell 1935)

[RELT4]

An aircraft P is 800 m due North of another aircraft Q.
Both are flying at the same height with constant velocities 150 ms-1 due West and 200 ms-1 N 30` W.
After what time will the aircraft be closest together and how far apart will they then be?

(from Bostock and Chandler 1975)

[RELT5]

A speedboat travelling due East at 100 kmh-1 is 500 m due North of a launch when the launch sets off to try to catch the speedboat.
If the speed of the launch is 60 kmh-1, show that the launch cannot get closer to the speedboat than 400 m.

(from Bostock and Chandler 1975)

[RELT6]

A and B are two ships which, at 1200 hours, are at P and Q respectively, where PQ = 26 nautical miles.
A is steaming at 30 knots in a direction perpendicular to PQ and B is steaming on a straight course at 20 knots in such a direction as to approach A as closely as possible.

Show that B steams at an angle of arcsin(2/3) with PQ.
Find when the ships are closest together.

(from Bostock and Chandler 1975)

[RELT7]

A cruiser is travelling due East at 15 knots.
At 1200 hours a destroyer which is 12 nautical miles South West of the cruiser sets off at 20 knots to intercept the cruiser.
At what time will the interception occur and on what bearing should the destoyer travel?

(from Bostock and Chandler 1975)

[RELT8]

A destroyer moving N 30` E at 50 kmh-1 observes at noon a cruiser travelling due North at 20 kmh-1.
If the destroyer overtakes the cruiser one hour later find the distance and bearing of the cruiser from the destroyer at noon.

(from Bostock and Chandler 1975)

[RELT9]

Two aircraft are in horizontal flight at the same altitude.
One is flying due north at 500 kmh-1 whilst the other is flying due west at 600 kmh-1.
Realising that they are on a collision course, the pilots take action simultaneously when the aircraft are 10 km apart.
The pilot of the first plane changes his course to N 15` W maintaining his speed of 500 kmh-1 and the pilot of the second plane maintains his course but increases his speed to V kmh-1.
Find the value of V if the aircraft are still on collision courses.

(A-level exam: AEB)

[RELT10]

A port X is 18 nautical miles due north of another port Y.
Steamers A, B leave X, Y respectively at the same time, A travelling at 12 knots due east and B at 8 knots in a direction arcsin(1/3) east of north.
Find in magnitude and direction the velocity of B relative to A.
Prove that subsequently the shortest distance between A and B is 14 nautical miles and find the time taken to reach this position.
If when the steamers are in this position a boat leaves A and travels due west so as to intercept B, find at what speed the boat must travel

(A-level exam: JMB)

File: PROJCT

Projectile Problems.

[PROJ1]

A stone is thrown from the top of a cliff 70 m high at an angle of 30` below the horizontal and hits the sea 20 m from the bottom of the cliff.
Find the initial speed of the stone and the direction in which it hits the sea.

(from Bostock and Chandler 1975)

[PROJ2]

A ball is thrown from ground level with a velocity of 15 ms-1 at an angle of 60` to the horizontal.
Find when the ball hits the ground and the time at which it reaches its greatest height above the point of projection.

(from Bostock and Chandler 1975)

[PROJ3]

A gun has a maximum range of 200 m on the horizontal.
Find the velocity of a shell as it leaves the muzzle of the gun.

(from Bostock and Chandler 1975)

[PROJ4]

The greatest range of a particle, with a given velocity of projection, on a horizontal plane is 3000 metres.
Find the greatest range up a plane inclined at 30` to the horizontal.

(adapted from Humphrey 1930)

[PROJ5]

A ball is projected upwards at an angle of 30` to the horizontal, with a speed of 60 m/s.
The ball reaches a maximum height of H metres vertically above the horizontal plane containing the point of projection.
Calculate the value of H, the range on the horizontal plane, and the direction of motion of the ball 4 s after projection.

(O(A)-level exam: AEB)

[PROJ6]

A ball is thrown with a speed of 10 m/s from the top of a tower which is 15 m high.
The ball strikes the ground at the same level as the base of the tower and at a horizontal distance of 20 m from the point where the ball was thrown.
Calculate the angle of elevation at which the ball was thrown, the time of flight of the ball, and the direction in which the ball is moving at the instant when it strikes the ground.

(A-level exam (part): AEB)

[PROJ7]

A tile slides down a roof inclined at 20` to the horizontal starting 3 m from the edge of the roof.
Assuming that the roof is smooth, find the horizontal distance from the edge of the roof that the tile hits the ground if the edge of the roof is 8 m above ground level.

(from Bostock and Chandler 1975)

[PROJ8]

A gun with a muzzle velocity of 100 ms-1 is fired from the floor of a tunnel which is 4 m high.
Find the maximum angle of projection possible if the bullet is not to hit the roof, and the range of the gun with this angle of projection.

(from Bostock and Chandler 1975)

[PROJ9]

Two particles P and Q are fired simultaneously from two points A and B on level ground with speeds of projection U and 2U respectively.
If AB is the maximum range of P, and the particles collide when both are moving horizontally, find the angle of projection of Q.

(from Bostock and Chandler 1975)

[PROJ10]

Two particles are projected with the same sppeed from the same point.
The angles of projection are 2A and A and a time T elapses between the instants of projection.
If the particles collide in flight, find the speed of projection in terms of T and A.
If the collision occurs when one of the particles is at its greatest height, show that A is given by 4cos(A)^4 - cos(A)^2 - 1 = 0.

(A-level exam: AEB)

File: CRASH

Collision and Impact problems.
==============================

[CRSH1]

A hammer of mass 1.2 kg travelling at 15 ms-1 is brought to rest when it strikes a nail.
What impulse acts on the hammer?

(from Bostock and Chandler 1975)

[CRSH2]

A truck of mass 1200 kg is moving with a speed of 7 ms-1 when it collides with a second truck of mass 1600 kg which is stationary.
If the two trucks are automatically coupled together at impact, with what speed do they move on together?

(from Bostock and Chandler 1975)

[CRSH3]

A shell of mass 12 kg is fired horizontally from a gun of mass 750 kg, which stands at rest on horizontal ground and is free to move.
If the initial velocity of recoil of the gun is 3 ms-1, calculate the initial velocity of the shell and the magnitude of the impulse which acts on the gun.

(O(A)-level exam: AEB)

[CRSH4]

A gun of mass K M fires a shell of mass M
The barrel of the gun is elevated at an angle A and the gun recoils horizontally.
Find the angle at which the shell leaves the barrel.

(adapted from Bostock and Chandler 1975)

[CRSH5]

Two particles of masses 3 oz and 5 oz are connected by an inextensible string of length 14 feet which passes over a small at a height of 10 feet above a table on which the heavier particle rests, vertically beneath the pulley.
The other particle is raised to the pulley and allowed to fall.
Find the velocity of the system after the jerk, and the time at which it will first come to rest.

(from Humphrey 1930)

[CRSH6]

A particle of mass 4 kg is attached to one end X of a light inextensible string which passes over a smooth light pulley and supports particles of masses 2 kg and 3 kg at the other end Y.
The end X is held in contact with a horizontal table at a depth

6 m below the pulley, both portions of the string being vertical and the particles at Y hanging freely.
The system is released from rest.
When Y has descended a distance of 2.5 m, the particle of mass 2 kg is disconnected and begins to fall freely.
Calculate the greatest height reached by X above the table and the momentum of the 4 kg particle when it strikes the table.

(A-level exam: U of L)

[CRSH7]

A smooth sphere of mass 0.5 kg moving with a horizontal speed of 3 ms-1 strikes at right angles a vertical wall and bounces off the wall with horizontal speed of 2 ms-1.
Find the coefficient of restitution between the sphere and the wall and the impulse exerted on the wall at impact.

(from Bostock and Chandler 1975)

[CRSH8]

A smooth sphere A of mass 2 M, moving on a horizontal plane with speed U, collides directly with another smooth sphere B of equal radius and of mass M, which is at rest.
If the coefficient of restitution between the spheres is E, find their speeds after impact.

(A-level exam (part): U of L)

[CRSH9]

A smooth uniform sphere A of mass M, sliding with speed U on a horizontal table, collides obliquely with a smooth uniform sphere B, of mass 4 M and of equal radius, which is at rest on the table.
At the moment of impact, the path of A makes an angle Alpha with the line of centres of the spheres.
If, after impact, B moves with speed (3 U cos Alpha)/10, find the coefficient of restitution between the spheres.
If the kinetic energy of the spheres after the impact are equal, find tan Alpha.

(A-level exam: AEB)

[CRSH10]

Three small spheres A, B and C, of equal radii and masses 2M, 3M and 4M respectively, lie at rest on a frictionless horizontal plane with their centres in a straight line.
The sphere A is made to move towards B with a speed 2U and after the collision B begins to move towards C with speed U.
The sphere B is brought to rest by its collision with C.
Calculate the coefficients of restitution between A and B and between B and C.
Show that, in all, three collisions only take place and find the final speeds of each sphere in terms of U.

(A-level exam: AEB)

File: SPRING

Spring and Energy problems.

[SRNG1]

A particle of mass M1 is suspended from the end of a spring of length L1 and elasticity E.
A second spring with length L2 and elasticity 2E is attached to the first particle, and another particle of mass M2 is suspended from the second spring.
Find the extension of each spring.

[SRNG2]

An elastic string of natural length 4L and modulus of elasticity 4Mg is stretched between two points A and B which are on the same level, where AB = 4L.
A particle attached to the midpoint of the string hangs in equilibrium with both portions of string making 30` with AB.
What is the mass of the particle?

(from Bostock and Chandler 1975)

[SRNG3]

Two springs AB BC are joined together end to end to form one long spri
The natural lengths of the separate spings are 1.6 m and 1.4 m and their moduli of elasticity are 20 N and 28 N respectively.
Find the tension in the combined spring if it is stretched between two points 4 m apart.

(from Bostock and Chandler 1975)

[SRNG4]

If the work done in halving the length of a spring of modulus 4 N is 1.2 Joule what is the natural length?

(from Bostock and Chandler 1975)

[SRNG5]

An elastic string whose modulus is 4 N is stretched from 3 m to 4 m in length.
What is its increase in energy if its natural length was 2 m?

(from Bostock and Chandler 1975)

[SRNG6]

A light elastic string, of unstretched length A and modulus of elasticity W, is fixed at one end to a point on the ceiling of a room.
To the other end of the string is attached a particle of weight W.
A horizontal force P is applied to the particle and in equilibrium it is found that the string is stretched to three times its natural

length.
Calculate the angle the string makes with the horizontal, and the value of P in terms of W.

(A-level exam (part): U of L)

[SRNG7]

Two light elastic strings AB and CD each have natural length L and an extension of L/2 is produced in each string by tensions Mg and 2Mg respectively.
The strings are joined at their ends B and C and the end A is fastened to a fixed point.
From the end D is hung a particle of mass M.
Show that, when the mass M hangs at rest vertically below A, the total extension in the combined string ABCD is 3L/4.

(A-level exam (part): AEB)

[SRNG8]

A particle of mass M is suspended from a fixed point A by a light elastic string of natural length L and modulus of elesticity 4 Mg.
The particle is pulled down from its equilibrium position a distance D and then released.
If the particle just reaches the height of A, find D.

(from Bostock and Chandler 1975)

[SRNG9]

One end of an elastic string is fixed to a point A on a horizontal table.
The other end is attached to a heavy particle P of mass M.
The particle is pulled away from A until AP is of length 3L/2 and is then released.
If the natural length of the string is L and its modulus of elasticity is Mg find the velocity of the particle when the string reaches its natural length.

(adapted from Bostock and Chandler 1975)

[SRNG10]

Two particles A and B are connected by a light inelastic string which passes over a smooth pulley.
A is of mass M and B is of mass 2M.
Initially both particles are at rest at a depth 2L below the pulley.
If they are released from rest find their velocity when each has moved a distance L.

(from Bostock and Chandler 1975)

File: WORK

Work and Power problems.
========================

[WORK1]

A block of mass 500 kg is raised a height of 10 m by a crane.
Find the work done by the crane against gravity.

(from Bostock and Chandler 1975)

[WORK2]

A block is pulled at a constant speed of 5 ms-1 along a horizontal surface by a horizontal string.
If the tension in the string is 5 N, find the work done by the string in ten seconds.

(from Bostock and Chandler 1975)

[WORK3]

A block is pulled along a rough horizontal surface by a horizontal string.
If the string pulls the block at a steady speed and does work of 100 J in moving the block a distance of 5 m, find the tension in the string.

(from Bostock and Chandler 1975)

[WORK4]

A particle of mass 5 kg is pulled up a rough plane by a string parallel to the plane.
If the plane is inclined at 30 degrees to the horizontal, and if the work done by the tension in the string in moving the block a distance of 3 m at a steady speed is 90 J, find the coefficient of friction between the block and the plane.

(from Bostock and Chandler 1975)

[WORK5]

A train has a maximum speed of 80 kmh-1 on the level against resistance of 50000 N.
Find the power of the engine.

(from Bostock and Chandler 1975)

[WORK6]

A car has a maximum speed of 100 kmh-1 on the level with the engine working at 50 kW.
Find the resistance to motion.

(from Bostock and Chandler 1975)

[WORK7]

A lorry of mass 10000 kg has a maximum speed of 24 km/h up

a slope of 1 in 10 against a resistance of 1200 newtons.
Find the effective power of the engine in kilowatts.

(A-level exam (part): U of L)

[WORK8]

A car of mass 1500 kg tows another car of mass 1000 kg up a hill inclined at arcsin 1/10 to the horizontal.
The resistance to motion of the cars is 0.5 N per kg.
Find the tension in the tow rope at the instant when their speed is 10 ms-1 and the power output of the towing car is 150 kW.

(from Bostock and Chandler 1975)

[WORK9]

At the instant a car of mass 840 kg passes a sign post on a level road its speed is 90 km/h and its engine is working at 70 kW.
If the total resistance is constant and equal to 2100 N, find the acceleration of the car in m/s2 at the instant it passes the sign post.
Calculate the maximum speed in km/h at which this car could travel up an incline of arcsin(1/10) against the same resistance with the engine working at the same rate.

(A-level exam: AEB)

[WORK10]

A car of mass 1000 kg has a maximum speed of 15 m/s up a slope inclined at an angle theta to the horizontal where sin theta = 0.2.
There is a constant frictional resistance equal to one tenth of the weight of the car.
Find the maximum speed of the car on a level road.
If the car descends the same slope with its engine working at half its maximum power, find the acceleration of the car at the moment when its speed is 30 m/s.

(A-level exam: U of L)

*
* Rob's 30 problems for the parser
*
*

* [CRSH1]

A hammer of mass 2 kg travelling at 15 ms-1 is brought to rest
when it strikes a nail.
What impulse acts on the hammer?

* was 1.2 kg (from Bostock and Chandler 1975)

* [FORC1]

A small object of weight 10 N rests in equilibrium on a rough
plane inclined at 30 degrees to the horizontal.
Calculate the magnitude of the frictional force.

* (from Bostock and Chandler 1975)

* [MOTN1]

A stone is dropped from a cliff 100 m above the sea.
Find the speed with which it hits the sea.

* (from Bostock and Chandler 1975)

* [MOTN2]

A ball is thrown vertically upward to a height of 10 m.
Find the time taken to reach this height and the initial speed
of the ball.
* (from Bostock and Chandler 1975)

* [MOTN3]

A stone is projected vertically upward with a speed of 21 ms-1.
Find the distance travelled by the stone in the first 3 s of its
motion.
* (from Bostock and Chandler 1975)

* [MOTN4]

A ball is thrown vertically upward with a speed of 15 ms-1 from
a point which is 1 m above ground level.
Find the speed with which the ball hits the ground.

* was 0.7 meters
* (from Bostock and Chandler 1975)

* [MOTN5]

A stone is dropped from the top of a tower.
In the last second of its motion it falls through a distance
which is 1/5 of the height of the tower.
Find the height of the tower.

* (from Bostock and Chandler 1975)

*   [MOTN6]

A stone is dropped from the top of a building 20 m high.
A second stone is dropped from a point half-way up the same building.
Find the time that should elapse between the release of the two stones if they are to reach the ground at the same time.

*   (from Bostock and Chandler 1975)

*   [MOTN7]

A particle which is moving in a straight line with constant acceleration takes 3 s and 5 s to cover two succesive distances of 1 m.
Find the acceleration.

*   (from Bostock and Chandler 1975)

*   [NOVK1]

A lever 10 ft long is pinned at its left end.
The lever is supported by a spring with a constant of 40 lb/ft.
The spring is attached 6 ft from the left end of the lever.
A weight of 20 lb is attached at the other end of the lever.
The weight of the lever is 8 lb.
How much is the spring stretched?

*   [NOVK2]

Where must a weight be hung on a pole, of negligible weight, so that the boy at one end supports 1/3 as much as the man at the other end?

*   (from Schaum 1961)

*   [NOVK3]

A scaffold 10 ft long is supported by ropes attached at each end.
The scaffold weighs 100 lb.
One painter weighing 150 lb stands on the scaffold 4 ft from one end, while a second painter weighing 175 lb stands on the scaffold 2 ft from the other end.
What is the tension on each of the ropes supporting the scaffold?

*   (from Dull, Metcalfe and Williams 1964)

*   [NOVK4]

A horizontal uniform bar 10 m long is supported by two ropes attached at its ends.
The rope on the left end makes an angle of 45 degrees with the horizontal, while the rope on the right end makes an angle of 60 degrees with the horizontal.
A weight of 100 nt is attached 2 m from the right end of the bar.
What is the weight of the bar?

*   [NOVK5]

A uniform scaffold 12 ft long and weighing 100 lb is supported horizontally by two vertical ropes hung from its ends.

Find the tension in each rope when a 180 lb painter stands
4 ft from one end.

* (from Schaum 1961)

* [NOVK6]

A uniform bar B-C is 100 cm long and weighs 50 lb.
The bar is to be supported at ends B and C.
An upward force of 40 lb is applied 80 cm from B.
Compute the forces on the supports.

* (from Schaum 1961)

* [NOVK7]

A uniform pole 20 ft long and weighing 30 lb is supported by
a boy 3 ft from one end and a man 6 ft from the other end.
At what point must a 150 lb weight be attached so that the man
supports twice as much as the boy?

(from Schaum 1961)

* [NOVK9]

The hinges of a door weighing 20 lb are 12 ft apart, and the door
is 3 ft wide.
The weight of the door is supported by the upper hinge.
Determine the forces exerted on the door at the hinges.

* (from Schaum 1961)

* [NOVK18]

A bridge is 80 ft long.
What force must the pier at each end of the bridge exert to
support an automobile weighing 2 tons which is 30 ft from one
end of the bridge?

* (from Dull, Metcalfe and Williams 1964)

* [PROJ3]

A gun has a maximum range of 200 m on the horizontal.
Find the velocity of a shell as it leaves the muzzle of the gun.

* (from Bostock and Chandler 1975)

* [PROJ4]

The greatest range of a particle, with a given velocity
of projection, on a horizontal plane is 3000 metres.
Find the greatest range up a plane inclined at 30 degrees to
the horizontal.

* (adapted from Humphrey 1930)

* [PULL1]

Two particles of mass B and C are connected by a light string
passing over a smooth pulley.
Find the acceleration of the particle of mass B.

* [PULL2]

A particle of mass 4 kg rests on a smooth horizontal table.
It is connected by a light inextensible string passing over a smooth pulley at the edge of the table to a particle of mass 2 kg, which is hanging freely.
Find the acceleration of the system and the tension in the string.

* (from Bostock and Chandler 1975)

* [PULL3]

A particle of mass 5 kg rests on a rough horizontal table.
It is connected by a light inextensible string passing over a smooth pulley at the edge of the table to a particle of mass of 6 kg, which is hanging freely.
The coefficient of friction between the 5 kg mass and the table is 1/3.
Find the acceleration of the system and the tension in the string.

* (from Bostock and Chandler 1975)

[PULL4]

Two particles of mass 3 kg and 4 kg are connected by a light inextensible string passing over a smooth fixed pulley.
The system is released from rest with the string taunt and both particles at a height of 2 m above the ground.
Find the velocity of the 3 kg mass when the 4 kg mass reaches the ground.

* (from Bostock and Chandler 1975)

* [PULL6]

Two particles of mass 3 kg and 5 kg are connected by a light inextensible string passing over a smooth pulley which is fixed to the ceiling of a lift.
Find the tension in the string when the system is moving freely, and the lift has a downward acceleration G ms-2.

(from Bostock and Chandler 1975)

* [RELT1]

The driver of a car travelling due East on a straight road at 40 kmh-1 is watching a train moving due North at 75 kmh-1.
What is the apparent speed and direction of motion of the train?

* (from Bostock and Chandler 1975)

* [SRNG1]

A particle of mass M1 is suspended from the end of a spring of length L1 and elasticity E.
A second spring with length L2 and elasticity 2E is attached to the first particle, and another particle of mass M2 is suspended from the second spring.
Find the extension of each spring.

* [SRNG6]

A light elastic string of unstretched length A and modulus of
elasticity W, is fixed at one end to a point on the ceiling
of a room.
To the other end of the string is attached a particle of weight W.
A horizontal force P is applied to the particle and in equilibrium
it is found that the string is stretched to three times its natural
length.
Calculate the angle the string makes with the horizontal, and the
value of P in terms of W.

* (A-level exam (part): U of L)

* [WORK1]

A block of mass 500 kg is raised a height of 10 m by a crane.
Find the work done by the crane against gravity.

* (from Bostock and Chandler 1975)

[WORK5]

A train has a maximum speed of 80 kmh-1 on the level against
resistance of 50000 N.
Find the power of the engine.

* (from Bostock and Chandler 1975)

* [WORK6]

A car has a maximum speed of 100 kmh-1 on the level with the
engine working at 50 kW.
Find the resistance to motion.

* (from Bostock and Chandler 1975)

* [WORK7]

A lorry of mass 10000 kg has a maximum speed of 24 km/h up
a slope of 1 in 10 against a resistance of 1200 newtons.
Find the effective power of the engine in kilowatts.

* (A-level exam (part): U of L)

end.