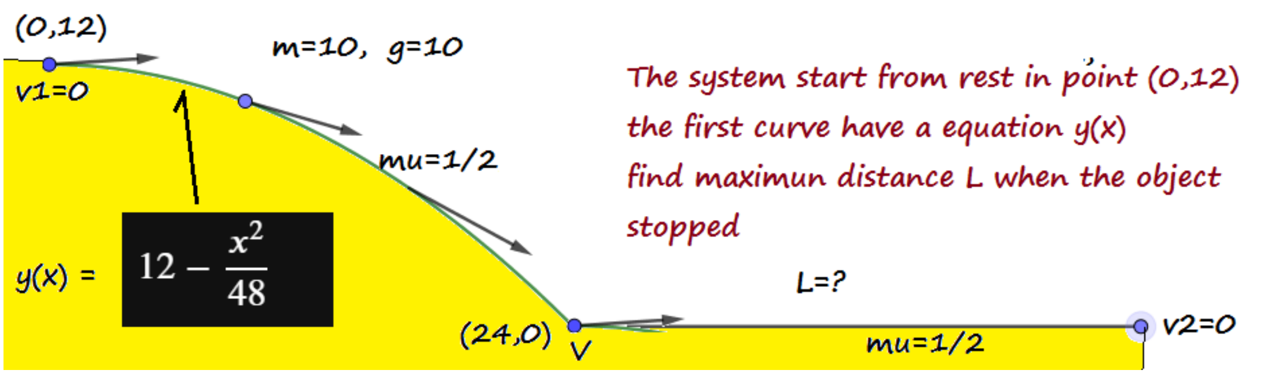
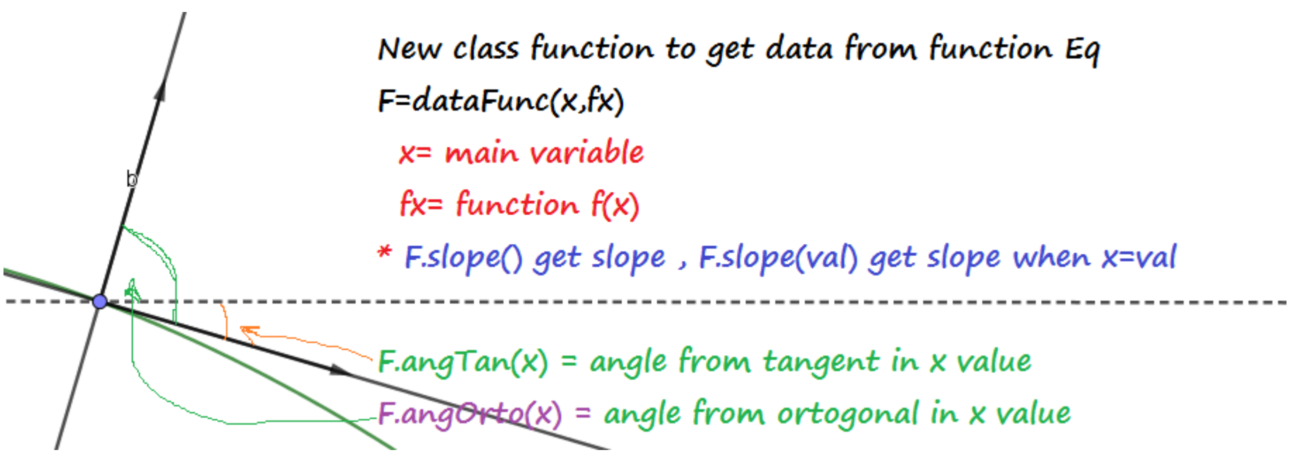
In [2]:
```
from sympy import *
from polyclass import *
from libaldo_math import *
from libaldo_show import *
from physic_lib import *
from IPython.display import display, Math
init_printing()
from IPython.display import Video
```



In [2]:
```
fx,x,alpha,m,g,F1,fr,mu,N1,V,L1=symbols('fx x alpha m g F1 fr mu N1 V L
1')
```

In [3]:
```
fx=12-x*x/48;fx
```

Out[3]: $12 - \frac{x^2}{48}$



In [4]:
```
F=dataFunc(x,fx)
```

In [5]:
```
F.slope()
```

Out[5]: $-\frac{x}{24}$

In [6]:
```
F.angTan()
```

In [6]:
```
F.angTan()
```

Out[6]: $-\left(\frac{x}{24}\right)$

In [7]:
```
F.angOrto()
```

Out[7]: $\left(\frac{24}{x}\right)$

In [8]:
```
# create object
P=mparticle(m=10,g=10,x1=0,x2=24,y1=12,y2=0,v1=0,v2=V)
```

In [9]:
```
# refernece axis is tangent line over function
P.add_forza(100,-F.angTan(x)-pi/2) # angle change respect x pos
P.add_forza(N1,pi/2)
P.add_forza(fr,pi)
```

In [10]:
```
# reasing Nornal value in P
P.setValue(N1,csolve(P.y_res(),N1))
```

In [11]:
```
# reasing frozz value in P line Normal * mu
P.setValue(fr,N1/2)
```

In [12]:
```
# Total Work is sum F(x)*dx ,unisymbols only asure sympy identifique un
ic variable
W=opemat(integrate(unisymbols(P.x_res(kope='s')),(x,0,12)),'v');W
```

Out[12]: -294.172617071776

In [13]:
```
# Total work = Energy Total and find V when P are in floor
csolve(P.energia()-W,V,korden=1)
```

Out[13]: 13.4597725309771

In [14]:
```
P2=mparticle(m=10,g=10,x1=0,v1=13.45,v2=0,y1=0,y2=0)
```

In [15]:
```
P2.add_forza(50,pi) # Rozz force
```

In [16]:
```
#Work in x axis
P2.work_due_forza('x')
```

Out[16]: $-50x_2$

In [17]:
```
#Energia total
P2.energia()
```

Out[17]: -904.5125

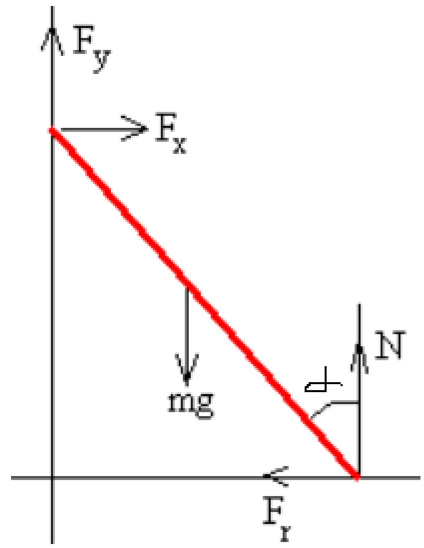
In [18]:
```
#find x2 if Wx=Etotal
L=csolve(P2.work_due_forza('x')-P2.energia(),x2,'L')
```

$L = 18.09025$

Static

Staircase supported by two perpendicular walls

which is the maximum value of alpha so that the ladder does not slip



In [19]:
```
Fx,Fy,m,g,N1,fr,mu,alpha,L=symbols('Fx Fy m g N1 fr mu alpha L',positiv
e=True)
```

In [20]:
```
# creating physical object P
P=mparticle()
# adding forces, angle , pos x, pos y  to every forces
P.add_forza(Fx,0,0,L*cos(alpha))
P.add_forza(m*g,-pi/2,L*sin(alpha)/2,L*cos(alpha)/2)
P.add_forza(fr,pi,L*sin(alpha),0)
P.add_forza(N1,pi/2,L*sin(alpha),0)
# Fy = cero...Ok
```

In [21]:
```
# get resultant forces in X equal cer0 then fr=Fx
P.x_res()
```

Out[21]: $Fx - fr$

In [22]:
```
# # get resultant forces in Y equal cer0 then Normal= weight
P.y_res()
```

Out[22]: $N_1 - gm$

In [23]:
```
# get torque in B = point(L*sin(a),0) = cero  deduced by geometry  Ok n
erd?
P.torque(L*sin(alpha),0)
```

Out[23]: $-FxL\cos(\alpha) + \frac{Lgm\sin(\alpha)}{2}$

In [24]:
```
# now we will change   fr and N1  value inside P whit info getting abov
e
P.setValue(N1,m*g)
P.setValue(fr,N1*mu)
```

In [25]:
```
# get torque in B but with new data
P.torque(L*sin(alpha),0)
```

Out[25]: $-FxL\cos(\alpha) + \frac{Lgm\sin(\alpha)}{2}$

In [26]:
```
# also we know that torque equal cero and take this to find Fx whit cso
lve math func
csolve(P.torque(L*sin(alpha),0),Fx)
```

Out[26]: $\frac{gm\tan(\alpha)}{2}$

In [27]:
```
# seting Fx in P
P.setValue(Fx,csolve(P.torque(L*sin(alpha),0),Fx))
```

How get info when we used store_val() or setValue() and use it for my convenience

1 store_val = setValue, example...

```
P.setValue(N1,m*g)
P.setValue(fr,N1*mu)
P.setValue(Fx, bla..bla...)
```

3 get single value P.value (var)

```
P.value(Fx)
```

$\frac{gm\tan(\alpha)}{2}$

2. P.disp_solu() , whit this see value

```
P.disp_solu()
```

>>>

$$Fx = \frac{gm\tan(\alpha)}{2}$$

$$fr = gm\mu$$

$$N_1 = gm$$

Remmeber that we store value, internal algorith set whit last data value

In [28]:
```
P.disp_solu()
```

$$Fx = \frac{gm\tan(\alpha)}{2}$$

$$fr = gm\mu$$

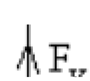
$$N_1 = gm$$

In [29]:
```
P.value(Fx)
```

Out[29]: $\frac{gm\tan(\alpha)}{2}$

In [30]:
```
# if Fx = fr then get alpha value
ta=csolve(P.value(Fx)-P.value(fr),tan(alpha),'Tg_a')
```

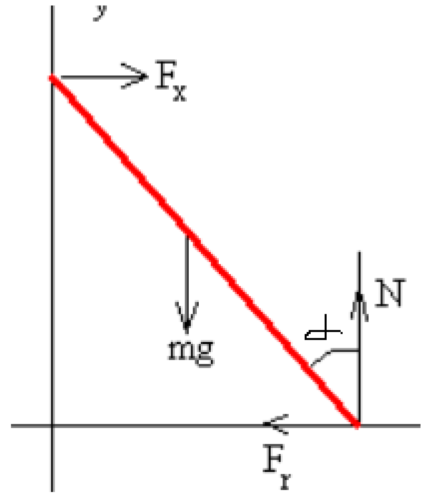
$$Tg_a = 2\mu$$



Static

Staircase supported by two perpendicular walls

which is the maximum value of alpha so that the ladder does not slip



In [31]:
```
Fx,Fy,m,g,N1,fr,mu,alpha,L,a_x,a_y,t,w_a=symbols('Fx Fy m g N1 fr mu alpha L a_x a_y t w_a',positive=True)
```

In [32]:
```
# Inercia
I_i=symbols('I_i')
```

In [33]:
```
# creating physical object P
P=mparticle()
# adding forces, angle , pos x, pos y  to every forces
P.add_forza(Fx,0,0,L*cos(alpha))
P.add_forza(m*g,-pi/2,L*sin(alpha)/2,L*cos(alpha)/2)
P.add_forza(fr,pi,L*sin(alpha),0)
P.add_forza(N1,pi/2,L*sin(alpha),0)
# Fy = cero...Ok
```

In [34]:
```
P.setValue(fr,N1*mu) # Now we will work whit m,g,Fx,N1 and mu for now
```

In [35]:
```
# Traslation X Equation , SumFx=mass*acc
Ex=polyclass(m*a_x,P.x_res());Ex.s()
```

Out[35]: $a_x m = Fx - N_1\mu$

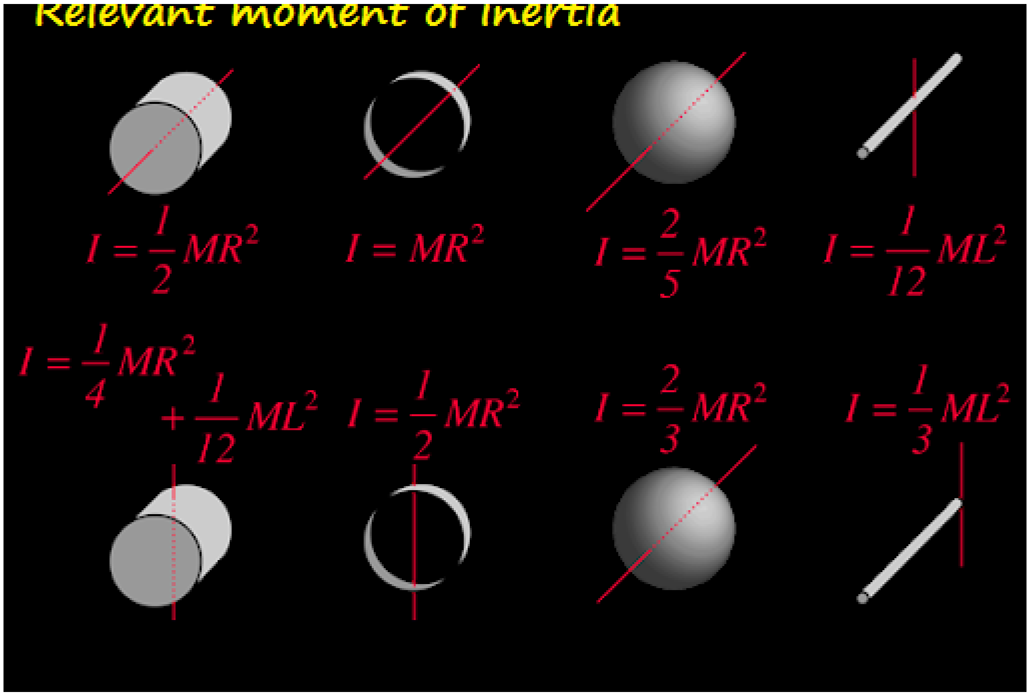
In [36]:
```
# Traslation Y Equation , SumFx=mass*acc
Ey=polyclass(m*a_y,P.y_res());Ey.s()
```

Out[36]: $a_y m = N_1 - gm$

In [37]:
```
# for geometry we know that mass center of P are in  (x1,y1) equal to.....
x1,y1=L*sin(alpha)/2,L*cos(alpha)/2
```

In [38]:
```
# get torque value in mass center for alpha value to use whit Inerce
P.torque(x1,y1)
```

Out[38]: $$-\frac{FxL\cos(\alpha)}{2} - \frac{LN_1\mu\cos(\alpha)}{2} + \frac{LN_1\sin(\alpha)}{2}$$



In [39]:
```
# InerciaMoment *AngulAcc=Torque then...
Ei=polyclass(I_i*w_a,P.torque(x1,y1));Ei.s()
```

Out[39]:
$$I_i w_a = -\frac{F x L\cos(\alpha)}{2} - \frac{L N_1 \mu\cos(\alpha)}{2} + \frac{L N_1 \sin(\alpha)}{2}$$

In [40]:
```
# but Ii=m*L*L/12 then redefine EqI
Ei=polyclass(m*L*L*w_a/12,P.torque(x1,y1));Ei.s()
```

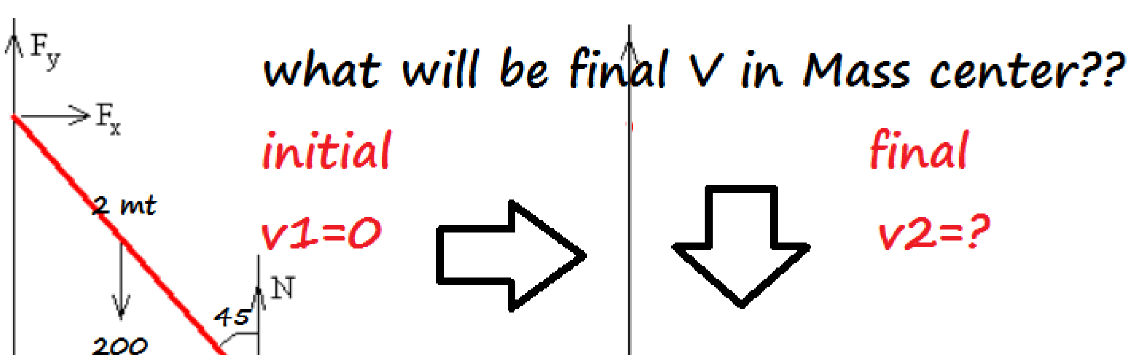
Out[40]:
$$\frac{L^2 m w_a}{12} = -\frac{F x L\cos(\alpha)}{2} - \frac{L N_1 \mu\cos(\alpha)}{2} + \frac{L N_1 \sin(\alpha)}{2}$$

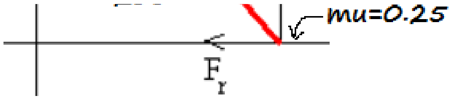
In [41]:
```
# 'M' = mult Ei by 2/L to reduce later aplied factor and simplidy... th
en...
Ei.psimplify('M',2/L,'fs')
```

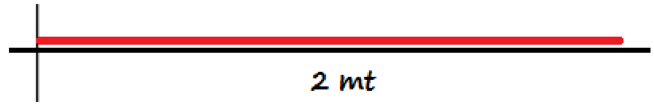
Out[41]:
$$\frac{L m w_a}{6} = -F x\cos(\alpha) - N_1 \mu\cos(\alpha) + N_1 \sin(\alpha)$$

In [42]:
```
Ei.solve(w_a)
```

Out[42]:
$$\frac{6(-F x\cos(\alpha) - N_1 \mu\cos(\alpha) + N_1 \sin(\alpha))}{L m}$$





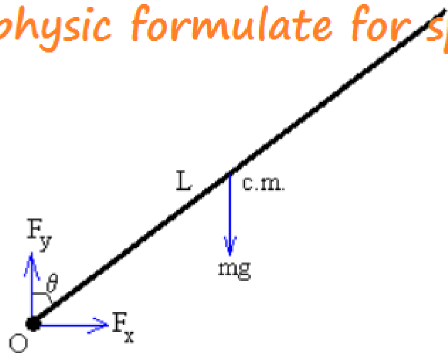


I'm not sure if solution are good, please send me your method to aldotb@gmail.com

In [43]:
```
Fx,Fy,N1,fr,alpha,t,x,V=symbols('Fx Fy N1 fr alpha t x V',positive=True
)
```

In [44]:
```
m=2
g=10
L=2
a1=pi/4
xx1,yy1,xx2,yy2=sin(pi/4),cos(pi/4),1,0
P=mparticle(m=2,g=10,x1=xx1,x2=xx2,y1=yy1,y2=yy2,v1=0,v2=V)
```

physic formulate for special problems....



Aplicando la segunda ley de Newton escribimos

$m(a_t \cdot \cos\theta - a_n \cdot \text{sen}\theta) = F_x$
$m(a_n \cdot \cos\theta + a_t \cdot \text{sen}\theta) = mg - F_y$

$$a_t = \alpha \frac{L}{2} = \frac{3g}{4} \text{sen}\,\theta$$
$$a_n = \omega^2 \frac{L}{2} = \frac{3}{2} g(\cos\theta_0 - \cos\theta)$$

Dado el ángulo θ, despejamos F_x y F_y del sistema de ecuaciones.

$$F_x = \frac{3mg}{4} \text{sen}\,\theta(3\cos\theta - 2\cos\theta_0)$$
$$F_y = mg - \frac{3mg}{4}(1 + 2\cos\theta\cos\theta_0 - 3\cos^2\theta)$$

In [45]:
```
Fx=3*m*g*sin(alpha)*(3*cos(a1)-cos(alpha))
Fy=m*g-3*m*g*(1+2*cos(alpha)*cos(a1)-3*kpow(cos(alpha),2))/4
```

In [46]:
```
x1=5*sin(alpha) # x pos in alpha function..
y1=5*cos(alpha) # y pos in alpha function..
```

In [47]:
```
# WorxX= Fx*dx = Fx(a)*dx/d(alpha)
Wx=opemat(integrate(Fx*x1,(alpha,pi/4,pi/2)),'v');Wx # integrate give m
e num value
```

Out[47]: 344.366530097709

In [48]:
```
Wy=opemat(integrate(Fy*y1,(alpha,pi/4,pi/2)),'v');Wy
```

Out[48]: 0.6942857471105

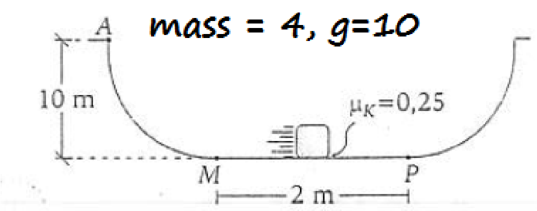
Out[48]: 9.6042857471195

In [49]:
```
Wt=Wx+Wy;Wt
```

Out[49]: 353.970815844828

In [50]:
```
# Energy in tha system is equal to Total Work.. ??? humm let me if are
 good
V=csolve(P.energia()-Wx-Wy,V,'V_f',korden=1)
```

$V_f = 19.1862698685429$



How many times will pass the block from plane section if is dropped from A with vel=0 and end in plane zone

In [51]:
```
m,g,nx = symbols('m g nx',positive=True)
```

In [52]:
```
m=4
g=10
d=2
mu=1/4
N1=m*g
fr=N1*mu
```

In [53]:
```
# Work when pass one time
W=fr*d;W
```

Out[53]: 20.0

In [54]:
```
# Work when pass nx time
Wn=W*nx;Wn
```

Out[54]: $20.0nx$

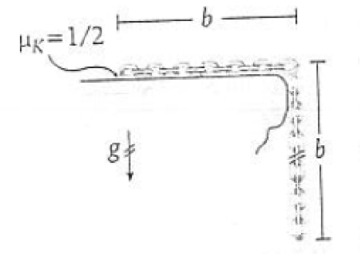
In [55]:
```
P=mparticle(m=4,g=10,y1=10,y2=0) # only y pos data is relevant
```

In [56]:
```
# Total work will be equal to Potential Energy Total
P.energia('P')
```

Out[56]: 400

In [57]:
```
nn=csolve(Wn-P.energia('P'),nx,'n_t')
```

$n_t = 20.0$



We have a chain that is abandoned from rest as show it in the sample. What is the velocity of A in the moment that it is placed totally vertical

In [34]:
```
m,g,Fx,fr,N1,b,mu,x,V=symbols('m g Fx fr N1 b mu x V',positive=True)
```

In [35]:
```
# Total work generate by fricction
mu=frs(1,2)
N1=m*(b-x)*g
fr=N1*mu
W=integrate(fr,(x,b,0))
show_res(W,'W_t')
```

$$W_t = -\frac{b^2 g m}{4}$$

In [36]:
```
# a little trick to solve quicky
def Tenergy(m,g,h,v):
    Ek=m*v*v/2
    Ep=m*g*h
    kres=Ek+Ep
    show_res(kres,'E_t')
    return Ek+Ep
```

In [37]:
```
# Total Energy when is droped b long
E1=Tenergy(m*b,g,-b/2,0)
```

$$E_t = -\frac{b^2 g m}{2}$$

In [38]:
```
# Total Energy when is droped 2*b long
E2=Tenergy(m*b*2,g,-b,V)
```

$$E_t = V^2 b m - 2b^2 g m$$

In [39]:
```
show_res(E2-E1,'final_e')
```

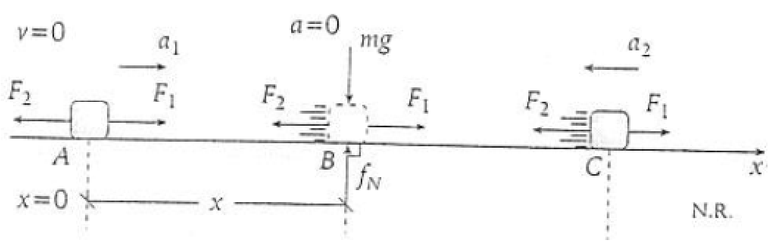
$$final_e = V^2 b m - \frac{3b^2 g m}{2}$$

In [40]:
```
#$create Eq to solve , Et = Wt, find V
e1=polyclass(E2-E1,W);e1.s()
```

Out[40]:
$$V^2 b m - \frac{3b^2 g m}{2} = -\frac{b^2 g m}{4}$$

In [41]:
```
V=e1.solve(V,'V',korden=1)
```

$$V = \frac{\sqrt{5}\sqrt{bg}}{2}$$



mass= 2   F1=15
g =10   F2=2*x+5
if start from rest , what will be x value when V is maximun

In [2]:
```
m,g,F1,F2,x,V=symbols('m g F1 F2 x V',positive=True)
```

In [3]:
```
F1=15
F2=(2*x+5)
P=mparticle(m=2,g=10,x1=0,x2=x,y1=0,y2=0,v1=0,v2=V)
```

In [4]:
```
P.add_forza(F1,0)
P.add_forza(F2,pi)
```

In [5]:
```
W=kintegrate(P.x_res(),x);W
```

Out[5]: $-x^2 + 10x$

In [6]:
```
P.x_res()
```

Out[6]: $10 - 2x$

In [7]:
```
V=csolve(P.energia()-W,V,'V',korden=0)
```

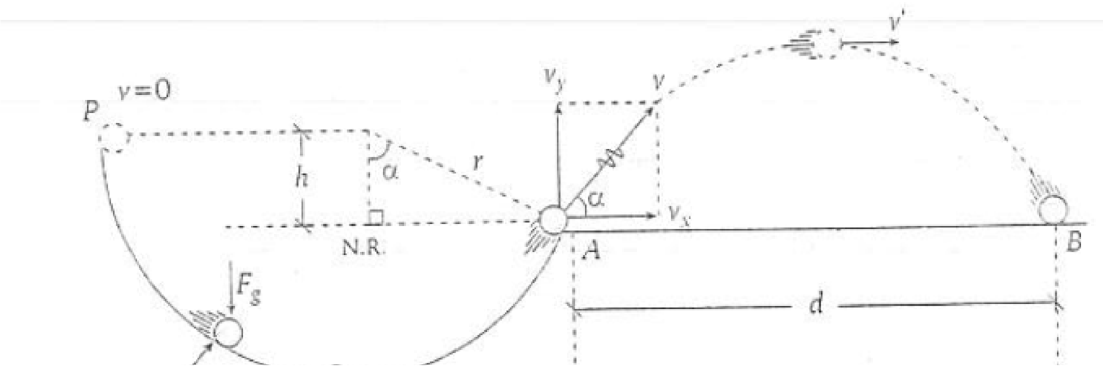
$$V = \sqrt{x(10 - x)}$$

In [9]:
```
Vm=kdiff(V,x);Vm
```

Out[9]: $$\frac{\sqrt{x(10 - x)}(5 - x)}{x(10 - x)}$$

In [11]:
```
X=csolve(Vm,x,'X_v')
```

$$X_v = 5$$



what should be alpha value to d will be maximum

mu=0

In [3]:
```
m,g,r,alpha,V=symbols('m g r alpha V',positive=True)
```

In [4]:
```
# Part one.. On circle....
P=mparticle(m=m,g=g,x1=-r,y1=0,x2=r*sin(alpha),y2=-r*cos(alpha),v1=0,v2
=V)
```

In [5]:
```
# Total Energy on circle equal cero, find out_V and store
V=csolve(P.energia(),V,'V',korden=1)
```

$$V = \sqrt{2}\sqrt{gr\cos(\alpha)}$$

In [6]:
```
# Part two fly Ball fly..... kinematic seting
B=mparticle(v=V,x1=0,y1=0,a=alpha,y2=0,g=g)
```

In [7]:
```
L=B.x_max();L
```

Out[7]:
$$\frac{4gr\sin(\alpha)\cos(\alpha)\cos(\alpha)}{g}$$

In [8]:
```
# Equation of d in alpha function
e1=polyclass(kdiff(L,alpha),0);e1.s()
```

Out[8]:
$$-8r\sin^2(\alpha)\cos(\alpha) + 4r\cos^3(\alpha) = 0$$

In [9]:
```
e1.psimplify('D',4*r,'sf') # take e1, divide by 4*r and simpli.. and fa
ctor
```

Out[9]:
$$(1 - 3\sin^2(\alpha))\cos(\alpha) = 0$$

In [10]:
```
e1.psimplify('D',cos(alpha),'sf') # del cos(al....) whit divide
```

Out[10]:
$$1 - 3\sin^2(\alpha) = 0$$

In [18]:
```
kr=e1.solve(sin(alpha),'asin(alpha)' ,korden=1)
```

$$asin(alpha) = \frac{\sqrt{3}}{3}$$

In [19]:
```
# ksubs is like subs in sympy but I dont know why sympy use double valu
e and defini..
# see whit sympy
L.subs(alpha,asin(kr))
# ES UNA MIERDAAAAA...#  for this reazon I create and use... this...
```

Out[19]:
$$\frac{4\sqrt{2}gr\cos(\alpha)}{3g}$$

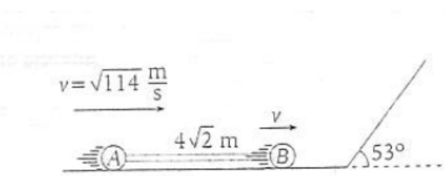
ksubs(Equation to change, what value, new value, opt opemat)

In [20]:
```
L #remember that L are in alpha funcction
```
Out[20]:

$$\frac{4gr\sin(\alpha)\cos(\alpha)\cos(\alpha)}{g}$$

In [21]:
```
ksubs(L,alpha,asin(kr)) # change alpha by new  angle... and voaaaLLaaa
```
Out[21]:

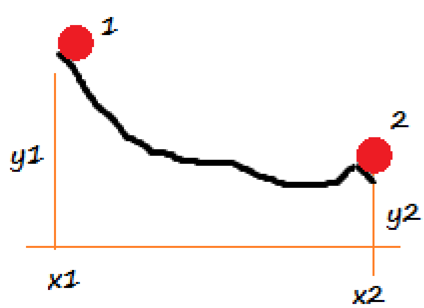
$$\frac{8\sqrt{3}r}{9}$$



Two small spheres are joined as shown, the bar is rigid of negligible weight. How fast will the side be when B has ascended 4 meters,

In [39]:
```
V,V1,V2,alpha,L,m,g,h,d1,d2=symbols('V,V1 V2 alpha L m,g h d1 d2',posit
ive=True)
g=10
h=4
m=1
V=rpow(144,2)
Pa=mparticle(m=m,g=g,x1=0,x2=d1,y1=0,y2=0,v1=V,v2=V1)
Pb=mparticle(m=m,g=g,x1=L,x2=d2,y1=0,y2=h,v1=V,v2=V2)
```

P.energia(option)
Energy Tot =P.energia()
Energy Kinetic P.energia('K')
Tot Pot = P.energia('P')
Pot in 1= P.energia('p1')
Pot in 2 = P.energia('p2')

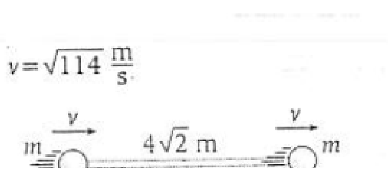


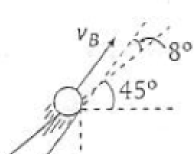
kinet Tot = P.energia('K')
Kinet in 1 = P.energia('k1')
kinet in 2 = P.energia('k2')
E tot in 1 = P.energia('1')
E tot in 2 = P.energia('2')

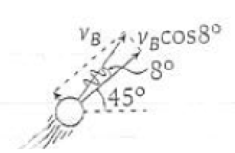
In [40]:
```
# E1 = Energy Total Initial
E1=Pa.energia('1')+Pb.energia('1')
```
Out[40]: 144

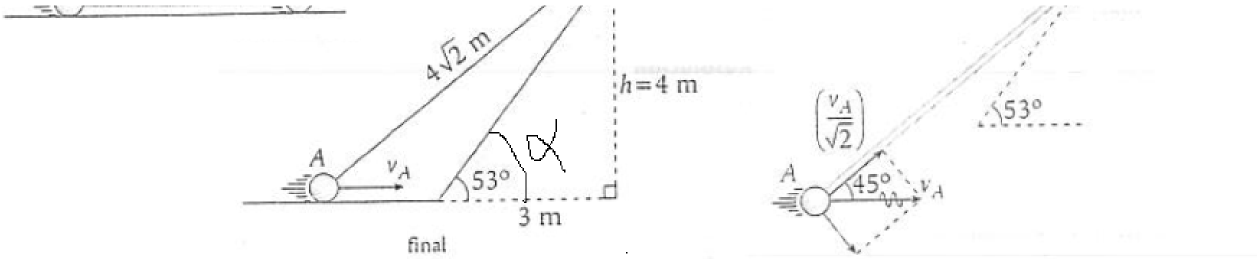
In [27]:
```
# E1 = Energy Total final
E2=Pa.energia('2')+Pb.energia('2')
```
Out[27]:

$$\frac{V_1^2}{2}+\frac{V_2^2}{2}+40$$









In [28]:
```
# create Equa Velocity same direcction are equal module
e2=polyclass(V1*cos(pi/4),V2*cos(alpha-pi/4));e2.s()
```

Out[28]:
$$\frac{\sqrt{2}V_1}{2} = V_2\sin\left(\alpha + \frac{\pi}{4}\right)$$

In [29]:
```
e2.psimplify('M',1,'x') # e2*1 and trig expand
```

Out[29]:
$$\frac{\sqrt{2}V_1}{2} = V_2\left(\frac{\sqrt{2}\sin(\alpha)}{2} + \frac{\sqrt{2}\cos(\alpha)}{2}\right)$$

In [30]:
```
e2.setValue(sin(alpha),frs(3,5)) # e2 change alpha = 53grad
```

Out[30]:
$$\frac{\sqrt{2}V_1}{2} = V_2\left(\frac{\sqrt{2}\cos(\alpha)}{2} + \frac{3\sqrt{2}}{10}\right)$$

In [31]:
```
e2.setValue(cos(alpha),frs(4,5))
```

Out[31]:
$$\frac{\sqrt{2}V_1}{2} = \frac{7\sqrt{2}V_2}{10}$$

In [32]:
```
v22=e2.solve(V2,'V2')
```

$$V2 = \frac{5V_1}{7}$$

In [33]:
```
e3=polyclass(E1,E2);e3.s()
```

Out[33]:
$$144 = \frac{V_1^2}{2} + \frac{V_2^2}{2} + 40$$

In [34]:
```
e3.setValue(V2,v22)
```

Out[34]:
$$144 = \frac{37V_1^2}{49} + 40$$

In [35]:
```
v11=e3.solve(V1,'V1',korden=1)
```

$$V1 = \frac{14\sqrt{962}}{37}$$

In [19]:
```
nsimplify(e3.solve(V1,korden=1))
```

Out[19]:
$$\frac{14\sqrt{962}}{37}$$