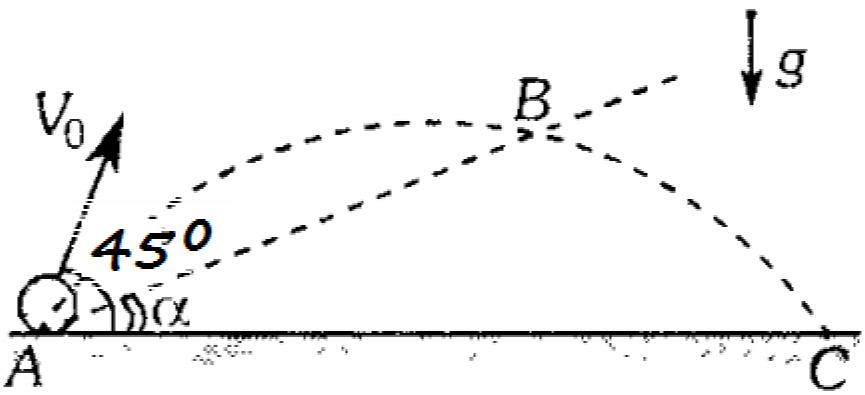
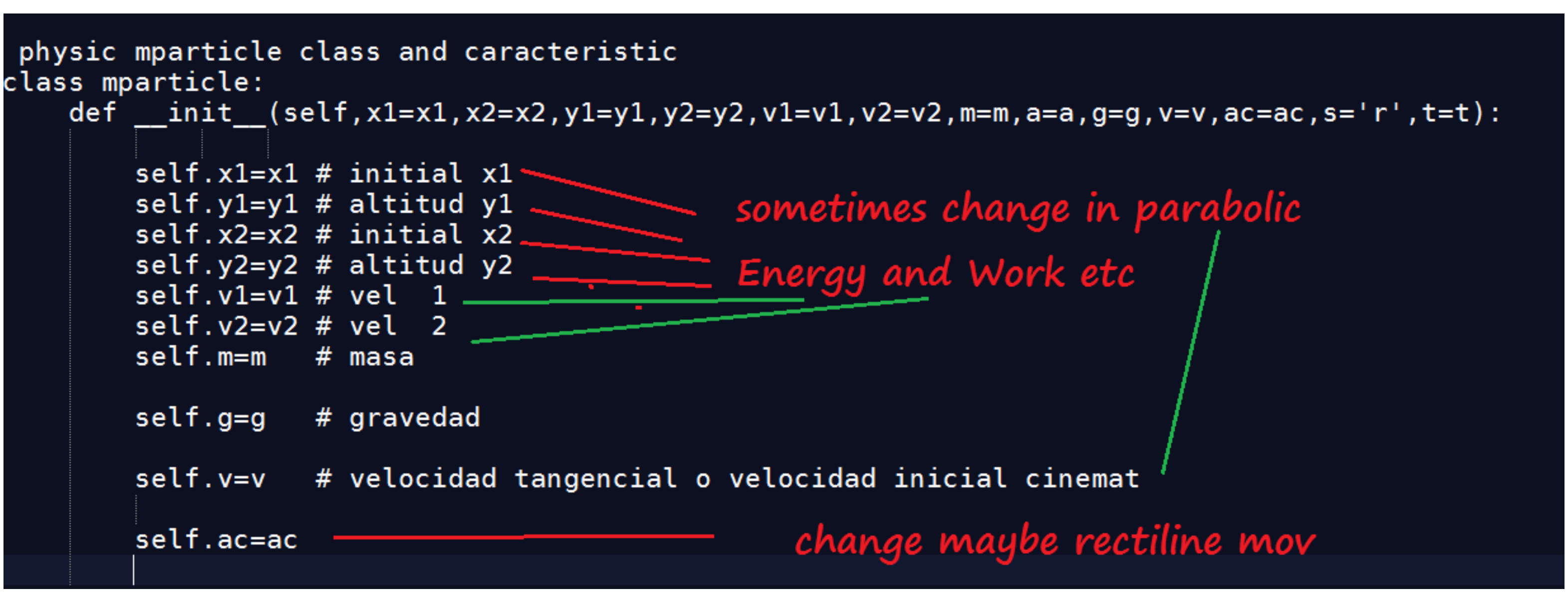
```
from sympy import *
from polyclass import *
from libaldo_math import *
from libaldo_show import *
from physic_lib import *
from IPython.display import display, Math
init_printing()
```

Halle α si $t_{AB} = \dfrac{1}{2} t_{AC}$




```
# creating symbols variable
t,V_o,alpha,B_x,B_y,g=symbols('t V_o alpha B_x B_y g')
```

some explication how used mparticle class()



```
 physic mparticle class and caracteristic
class mparticle:
    def __init__(self,x1=x1,x2=x2,y1=y1,y2=y2,v1=v1,v2=v2,m=m,a=a,g=g,v=v,ac=ac,s='r',t=t):

        self.x1=x1 # initial x1
        self.y1=y1 # altitud y1
        self.x2=x2 # initial x2
        self.y2=y2 # altitud y2
        self.v1=v1 # vel  1
        self.v2=v2 # vel  2
        self.m=m   # masa

        self.g=g   # gravedad

        self.v=v   # velocidad tangencial o velocidad inicial cinemat

        self.ac=ac
```

sometimes change in parabolic

Energy and Work etc

change maybe rectiline mov

```
P=mparticle(x1=0,y1=0,v=V_o,ac=0,a=pi/4) # moss relevant in this case
```

remember functions parabolic

kinematic classic functions

```
# *********    kinematic      ***********

    def x_vel(self,t=t,kope='',keval=True):

    def y_vel(self,t=t,kope='',keval=True):

    def xy_vel(self,tt='',kope='',keval=True):

    def x_pos(self, t=t,kope='',keval=True):

    def y_pos(self, t=t,kope='',keval=True):

    def y_max(self,t=t,keval=True,kope='',krelative=True):

    def x_max(self,kope='',keval=True):

    def t_fly(self,kope='',keval=True):

    def tan_angle_in_t(self,t=t,keval=True,kope='',ktan=True):
```

```
P.t_fly() # total time from A to C
```

$$\frac{\sqrt{2}V_o}{g}$$

```
# T2= total time from A to B that is t_fly()/2
T2=P.t_fly()/2 ; T2
```

$$\frac{\sqrt{2}V_o}{2g}$$

```
B_x=P.x_pos(T2);B_x # find x pos of B in T function
```

$$\frac{V_o^2}{2g}$$

```
B_y=P.y_pos(T2);B_y # find y pos of B in T function
```

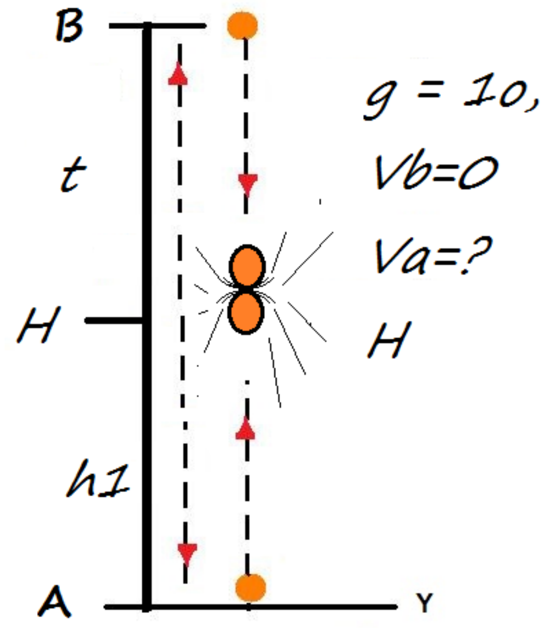
$$\frac{V_o^2}{4g}$$

```
Ta=B_y/B_x
sE([tan(alpha),'=',Ta])
```

$$\tan(\alpha) = \frac{1}{2}$$

Dear admirable and smart Nerd.... remeber that some time proccedure is to long but is only to learn, share and develop our passion ......

Mathematics is the reason for life



if you throw the ball from the ground with velocity Va the maximum height is H, where the balls collide if another ball is dropped whit Vb=0 at the same time

From now we will solve problems more fast and less lines script.. I hope dontmake confuse

I will try to introducce mpoliclass functions to manage severa equations

```
T,Va,H,h1,g=symbols('T Va H h1 g',positive=True) # we need positive=True due sqrt answers
```

```
# working whit Ball A alone
A=mparticle(x1=0,y1=0,y2=H,v=Va,g=g,a=pi/2,ac=0)
```

```
A.y_max() # that is equal to H.. and we use to find Va
```

$$\frac{Va^2}{2g}$$

```
csolve(A.y_max()-H,Va)
```

$$\sqrt{2}\sqrt{H}\sqrt{g}$$

Creating Equation No 1

```
e1=polyclass(Va,csolve(A.y_max()-H,Va));e1.Q
```

$$Va = \sqrt{2}\sqrt{H}\sqrt{g}$$

```
A.y_pos(T) # finding y pos in time equal T
```

$$-\frac{T^2 g}{2} + TVa$$

Creating Equation No 2

```
e2=polyclass(h1,A.y_pos(T));e2.Q
```

$$h_1 = -\frac{T^2 g}{2} + TVa$$

```
# working whit Ball B
B=mparticle(x1=0,y1=H,y2=h1,v=0,g=g,a=-pi/2,ac=0)
```

```
B.y_pos(T) # finding y in T time
```

$$H - \frac{T^2 g}{2}$$

Creating Equation No 3

```
e3=polyclass(B.y_pos(T),h1);e3.Q
```

$$H - \frac{T^2 g}{2} = h_1$$

Creating Equation matrix to use in Mpolyclass

```
vec=[e1,e2,e3]
```

Creating super class of several polynomies Eq

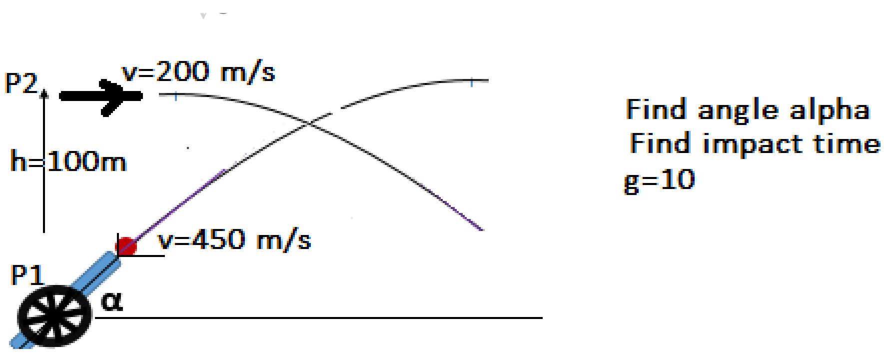
```
P=Mpolyclass(vec)
```

```
# what happend... what have..??
P.s()
```

$$\begin{aligned} Va &= \sqrt{2}\sqrt{H}\sqrt{g} \\ h_1 &= -\frac{T^2 g}{2} + TVa \\ H - \frac{T^2 g}{2} &= h_1 \end{aligned}$$

```
# magic solve alll that you want ..  if it possible... and Voalaaaa
P.matSolve([Va,h1,T],[0,1,2],kremp=True,kshow=False)
```

$$Va = \sqrt{2}\sqrt{H}\sqrt{g}$$
$$h_1 = \frac{3H}{4}$$




```
T,alpha=symbols('T alpha')
```

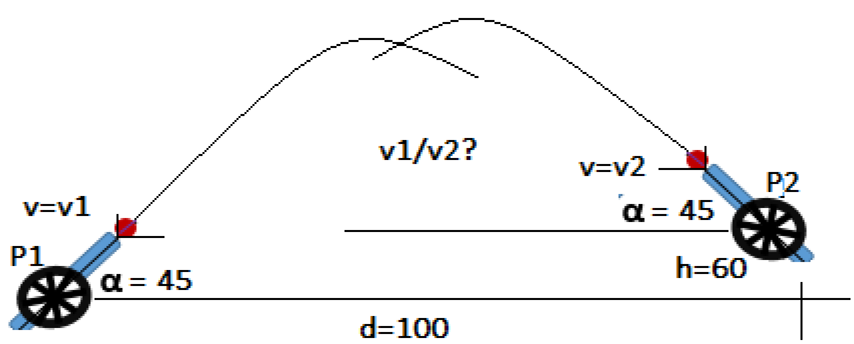
```
P1=mparticle(x1=0,y1=0,v=450,a=alpha,g=10,ac=0)
P2=mparticle(x1=0,y1=100,v=200,a=0,g=10,ac=0)
```

```
# creating e1 Eq to manage
e1=polyclass(P1.x_pos(T),P2.x_pos(T)) ; e1.Q
```

$$450T\cos(\alpha) = 200T$$

```
e1.solve(cos(alpha))
```

$$\frac{4}{9}$$




```
T,V1,V2=symbols('T V1 V2')
```

```
P1=mparticle(x1=0,y1=0,v=V1,a=pi/4,g=10,ac=0)
P2=mparticle(x1=0,y1=60,v=V2,a=pi/4,g=10,ac=0)
```

```
# creating e1 Eq to manage
```

```
e1=polyclass(P1.y_pos(T),P2.y_pos(T))
e2=polyclass(P1.x_pos(T)+P2.x_pos(T),100)
P=Mpolyclass([e1,e2])
P.s()
```

$$-5T^2 + \frac{\sqrt{2}TV_1}{2} = -5T^2 + \frac{\sqrt{2}TV_2}{2} + 60$$
$$\frac{\sqrt{2}TV_1}{2} + \frac{\sqrt{2}TV_2}{2} = 100$$

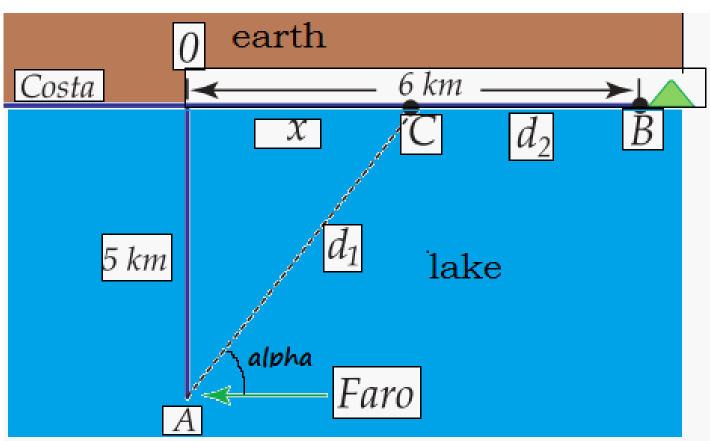
```
P.matSolve([V1,V2],[0,1],kremp=True,kshow=False)
```

$$V_1 = \frac{80\sqrt{2}}{T}$$
$$V_2 = \frac{20\sqrt{2}}{T}$$

```
sE([V1/V2,'=',4])
```

$$\frac{V_1}{V_2} = 4$$

```
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()
```



<- Store

velocity in lake 2 km/h
velocity in earth 4 km/h
find C point to go from A..C..B
in less Time

```
L,alpha,X,t1,t2=symbols('L alpha X t1 t2',positive=True)
A=mparticle(x1=0,y1=-5,y2=0,x2=X,v=2,a=alpha,g=0,ac=0)
```

```
t1=A.displacement()/2;t1 # t1= distan(A-C)/vel_lake = A.displacement(A to C)/2
```

$$\frac{\sqrt{X^2 + 25}}{2}$$

```
t2=(6-X)/4;t2 # t2= dist(C-B)/vel_earth
```

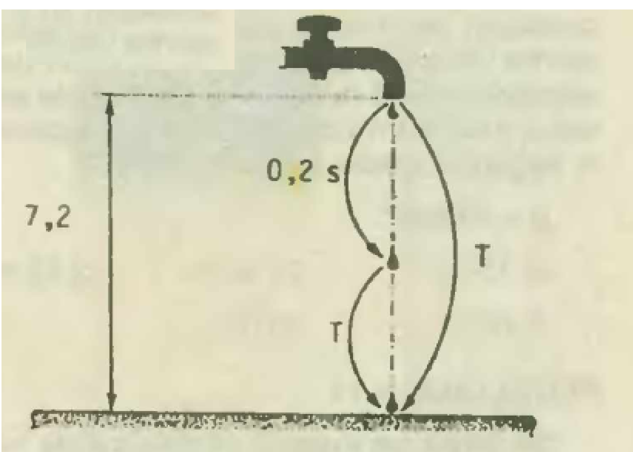
$$\frac{3}{2} - \frac{X}{4}$$

```
T_t=t1+t2;T_t # T_t=Tptal travel Time
```

$$-\frac{X}{4} + \frac{\sqrt{X^2 + 25}}{2} + \frac{3}{2}$$

```
# minimun time is when diff of time(X) equalcero and use csolve to find X same time
T_t.subs(X,csolve(diff(T_t,X),X))
```

$$\frac{3}{2} + \frac{5\sqrt{3}}{4}$$



A drop falls from a pipe every 0.1 second, if when the third drop is about to fall, the tap is opened and a jet of water comes out, with what speed must said jet come out so that it reaches the first drop, just when it arrives to the ground?

The spout is at a height of 7.2 meters from the floor.

g = 1 0 m /s

```
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()
```

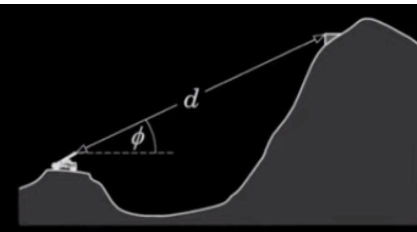
```
V,T=symbols('V T',positive=True)
A=mparticle(x1=0,x2=0,y1=7.2,g=10,v=0,a=-pi/2)
T_t=csolve(A.y_pos(T),T,'Total_t',korden=1) # findin   first drop time
```

$$Total_t = 1.2$$

```
# Find Vel from y_pos=0 and time is tota_t minus 0.4
B=mparticle(x1=0,x2=0,y1=7.2,y2=0,g=10,a=-pi/2,v=V)
V=csolve(B.y_pos(T_t-0.2),V,'V')
```

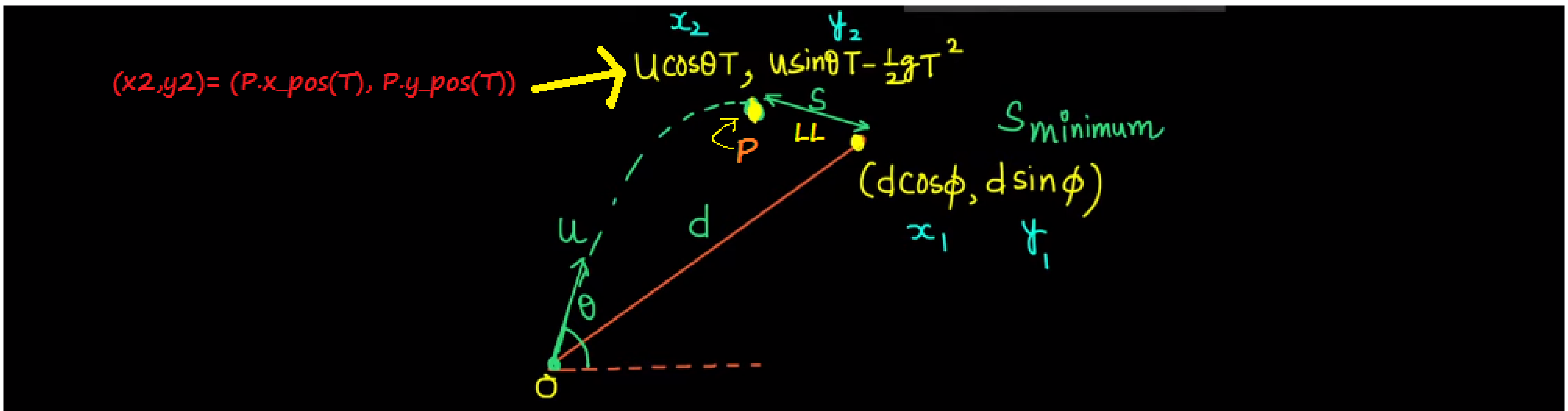
$V = 2.2$

A cannon installed at the top of a hill can fire shells in all directions. There is an enemy bunker at an angle of elevation ϕ and a distance d from the cannon. All the shells fired explode in air in time T before they reach the bunker. At what angle to the horizontal, should a shell be fired with a speed u to explode closest to the bunker? Acceleration due to gravity is g.




```
alpha,theta,d,T,u,g = symbols('alpha theta d T u g',positive=True)
```

```
P=mparticle(x1=0,y1=0,a=alpha,v=u,g=g,ac=0)
```



LL= $\frac{T^4 g^2}{4} - T^3 g u \sin(\alpha) + T^2 d g \sin(\theta) + T^2 u^2 \sin^2(\alpha) + T^2 u^2 \cos^2(\alpha) - 2 T d u \cos(\alpha - \theta) + d^2 \sin^2(\theta) + d^2 \cos^2$

```
Lx=d*cos(theta)-P.x_pos(T)
Ly=d*sin(theta)-P.y_pos(T)
LL= opemat(Lx*Lx+Ly*Ly ,'es');LL
```

$$\frac{T^4 g^2}{4} - T^3 g u \sin(\alpha) + T^2 d g \sin(\theta) + T^2 u^2 \sin^2(\alpha) + T^2 u^2 \cos^2(\alpha) - 2 T d u \cos(\alpha - \theta) + d^2 \sin^2(\theta) + d^2 \cos^2(\theta)$$

```
LL2=fpoly(LL,'filt',alpha);LL2 # filt = pick all monomies that contain alpha....
```

$$-T^3 g u \sin(\alpha) + T^2 u^2 \sin^2(\alpha) + T^2 u^2 \cos^2(\alpha) - 2 T d u \cos(\alpha - \theta)$$

```
fpoly(LL2,'list') # list = what is order of monomies to later create my diferent Eq whit this data
```

$$\left[T^2 u^2 \cos^2(\alpha),\ T^2 u^2 \sin^2(\alpha),\ -T^3 g u \sin(\alpha),\ -2 T d u \cos(\alpha - \theta)\right]$$

```
LL3=1+fpoly(LL2,'get',2)+fpoly(LL2,'get',3);LL3 # sum of squared sin cos equal 1
```

$$-T^3 gu\sin(\alpha) - 2Tdu\cos(\alpha-\theta) + 1$$

```
LL4=diff(LL3,alpha);LL4 # Now applied diferential to get minimum of LL4(theta) and equal to cero
```

$$-T^3 gu\cos(\alpha) + 2Tdu\sin(\alpha-\theta)$$

```
e1=polyclass(fpoly(LL4,'get',1), -1*(fpoly(LL4,'get',0)));e1.s() # get,n= giveme the polynomoe number n
```

$$2Tdu\sin(\alpha-\theta) = T^3 gu\cos(\alpha)$$

```
e1.ope4('D',2*T*d*u,1,'s') # 'D' = divide e1 by 2*T*d*y, 1= update new Eq,'s' = before update simplifiq
```

$$\sin(\alpha-\theta) = \frac{T^2 g\cos(\alpha)}{2d}$$

```
e1.opemat('x',1) #  'x' = expand trigonometric or... separate alpha and theta
```

$$\sin(\alpha)\cos(\theta) - \sin(\theta)\cos(\alpha) = \frac{T^2 g\cos(\alpha)}{2d}$$

```
e1.ope4('D',cos(alpha),1,'e') # 'D' = divide by cos(alpha), 1 = update, 'e' = expand
```

$$\frac{\sin(\alpha)\cos(\theta)}{\cos(\alpha)} - \sin(\theta) = \frac{T^2 g}{2d}$$

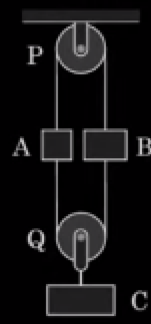
```
e1.ope4('S',sin(theta),1,'s') # 'S'=sum sin(theta ) to Eq ,1=..bla ...hope you getit if not ... search
```

$$\cos(\theta)\tan(\alpha) = \frac{T^2 g}{2d} + \sin(\theta)$$

```
# Answer to FIND alpha.. if yoy want applied atan() to get alpha
e1.ope4('D',cos(theta),1,'s')
```

$$\tan(\alpha) = \frac{T^2 g}{2d\cos(\theta)} + \tan(\theta)$$

Two blocks A and B of masses 3 kg and 6 kg respectively are suspended at the ends of a light inextensible cord. The cord passes over an ideal pulley P fixed to the ceiling. Ends of another light inextensible cord are attached at the bottoms of the blocks. This cord supports another ideal pulley Q from which a block C of mass 4 kg is suspended as shown in the figure. Initially the system is held motionless and then released. Which of the following conclusions can you make? Acceleration due to gravity is 10 m/s².



(a) All the blocks remain motionless.
(b) Acceleration magnitudes of blocks A and B is 10/3 m/s² while C remains motionless.
(c) Tensile forces in the upper and the lower cords are 40 N and 20 N respectively.
(d) Tensile forces in the upper and the lower cords are 60 N and 20 N respectively.

```
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()
```

```
T1,T2,T3,m,g,t=symbols('T1 T2 T3 m g t')
```

```
# A...
# Initialize A whit all vec forces
A=mparticle(m=3,g=10);A.add_forza(T1,pi/2);A.add_forza(30,-pi/2);A.add_forza(T2,-pi/2)
```

```
# get acc A in Y
A.simple_ac('y')
```

Mostrar resultado oculto

```
# B...
# Initialize B whit all vec forces
B=mparticle(m=6,g=10);B.add_forza(T1,pi/2);B.add_forza(60,-pi/2);B.add_forza(T2,-pi/2)
```

```
# get acc B in Y
B.simple_ac('y')
```

$$\frac{T_1}{6} - \frac{T_2}{6} - 10$$

```
# C...
# Initialize C whit all vec forces
C=mparticle(m=4,g=10);C.add_forza(2*T2,pi/2);C.add_forza(40,-pi/2)
```

```
# get acc C in Y  take note is cero and find T2
csolve(C.simple_ac('y'),T2,'T2')
```

$$T2 = 20$$

$$20$$

```
 # store new T2 in A y B object
A.store_val(T2,20);B.store_val(T2,20)
```

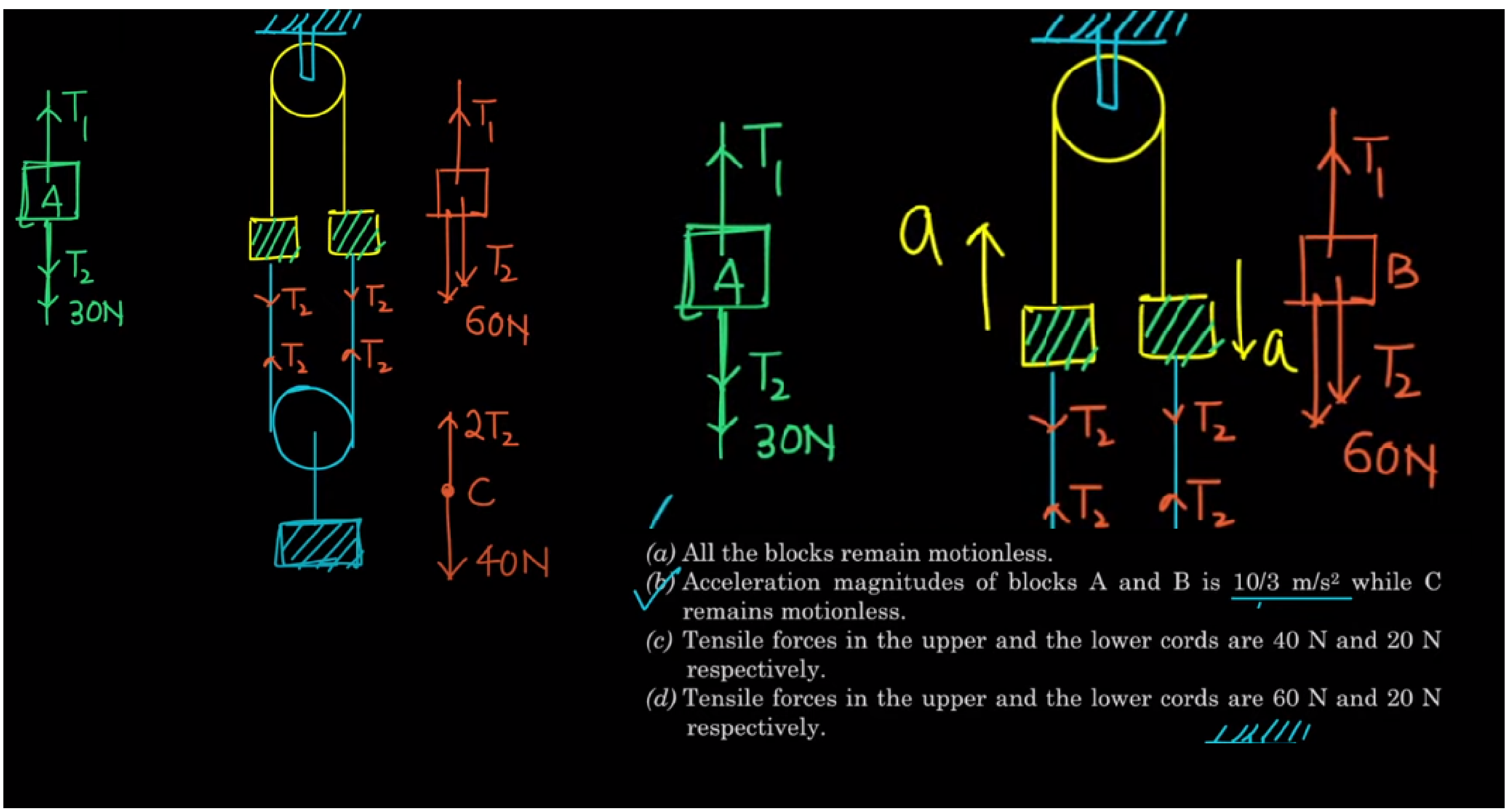
```
# get acc A in Y
csolve(A.simple_ac('y')+B.simple_ac('y'),T1)
```

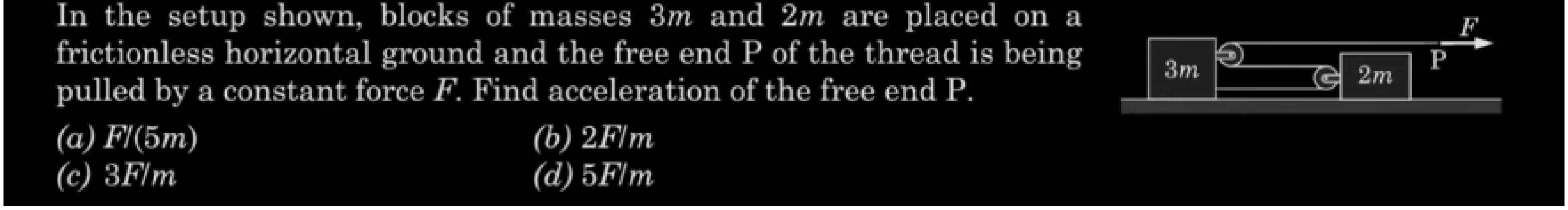
$$60$$

```
 # store new T2 in A y B object
A.store_val(T1,60);B.store_val(T1,60)
```

```
# get acc B in Y
A.simple_ac('y')
```

$$\frac{10}{3}$$





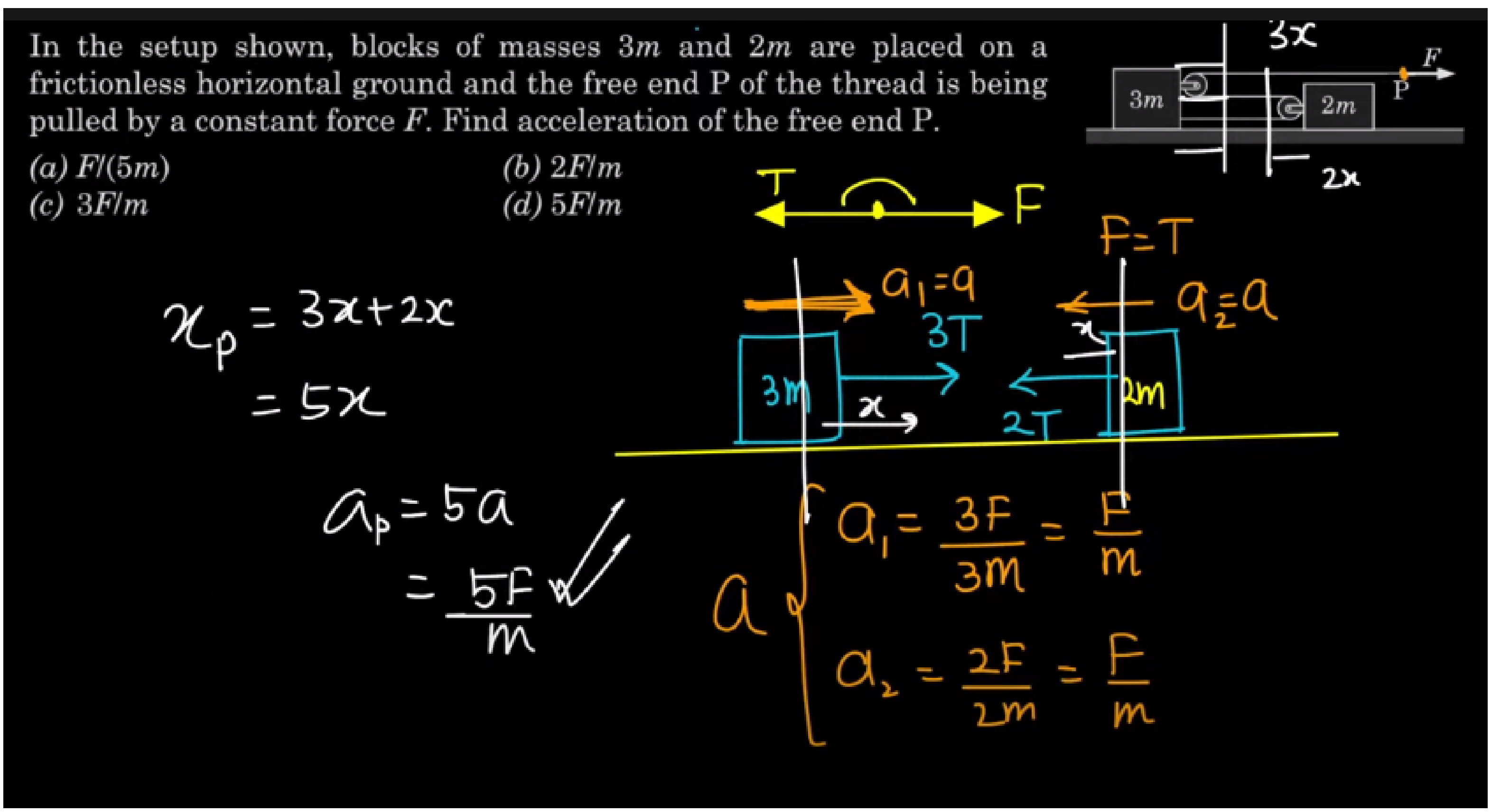

```
T1,F1,m,g=symbols('T1 F1 m g ')
```

```
# A...
# Initialize A whit all vec forces
A=mparticle(m=3*m,g=10);A.add_forza(3*F1,0)
A.simple_ac('x')
```

$$\frac{F_1}{m}$$

```
# B...
# Initialize B whit all vec forces
B=mparticle(m=2*m,g=10);B.add_forza(2*F1,0)
B.simple_ac('x')
```

$$\frac{F_1}{m}$$




```
#Set new data
A=mparticle(m=3*m,g=10,v=0,x1=0,a=0,ac=F1/m);A.add_forza(3*F1,0)
B=mparticle(m=2*m,g=10,v=0,x1=0,a=0,ac=F1/m);B.add_forza(2*F1,0)
```

```
# Desplazament of F is 2 Des B+3Des A then
X=3*A.x_pos(1)+2*B.x_pos(1);X
```

$$\frac{5F_1}{2m}$$

```
aC=symbols('aC')
csolve(X-aC/2,aC)
```

$$\frac{5F_1}{m}$$