Modelagem Computacional: Simulação 01 - Lançamento Oblíquo

Alunos: Álvaro Cardoso Vicente de Souza 133536

Gabriel Angelo Cabral Neves 136124

Jhonatan Hiroo Eguchi 133691

Docente: Prof. Dr. Marcos Gonçalves Quiles

Universidade Federal de São Paulo - UNIFESP Instituto de Ciência e Tecnologia - Campus São José dos Campos

São José dos Campos - Brasil

Agosto de 2020

Descrição do modelo

O projeto propõe a simulação de um lançamento oblíquo de um objeto no ar levando em conta o atrito do mesmo. Para isso, foi usada uma implementação em Python 3 levando em conta as demais características que o percurso pode ter.

Utilizando o método de Euler e a base sobre o lançamento oblíquo, foi possível obter a velocidade e o deslocamento do corpo utilizado. Levando em consideração diversos casos e parâmetros distintos podemos analisar o percurso e suas peculiaridades.

Modelo Matemático

Para essa simulação consideramos o modelo matemático do lançamento oblíquo de um corpo, levando em conta os seguintes parâmetros:

- Massa do corpo - m;
- Velocidade inicial - v(0);
- Ângulo de lançamento - teta (θ);
- Altura inicial - s(0);
- Coeficiente de atrito - k;
- Aceleração gravitacional - g.

Seguindo o modelo da física newtoniana podemos descrever o seguinte modelo matemático fundamentado na segunda lei de Newton:

$$F = m \cdot a$$

Porém sabemos que a aceleração é a derivada da posição, logo:

$$a = \frac{dv}{dt} \Rightarrow F = m \cdot \frac{dv}{dt}$$

Nesta simulação em específico consideramos a força peso (*p*) e a força de atrito.

$$p = m \cdot g \qquad\qquad F_{atrito} = -k \cdot v$$

Assim podemos obter a força resultante apenas somando ambas as forças:

$$F_{resultante} = p + F_{atrito}$$

$$F_{resultante} = m \cdot g - k \cdot v \Rightarrow m \cdot \frac{dv}{dt} = m \cdot g - k \cdot v$$

Nesse ponto obtemos uma equação diferencial não separável, ou seja, uma equação com uma solução analítica que exige conhecimentos matemáticos mais rebuscados, porém, tendo em vista o objetivo da disciplina e o fato de que alguns problemas não podem ser modelados através de equações com soluções analíticas triviais, ou nem possuem tais soluções, utilizaremos uma ferramenta do cálculo numérico conhecida como método de Euler para realizar nossa simulação.

Agora, basta dividir a equação pela massa(*m*) e integrá-la numericamente aplicando o método de Euler e definindo o passo de integração para podermos visualizar a evolução da velocidade.

$$\frac{m \cdot \frac{dv}{dt}}{m} = \frac{m \cdot g - k \cdot v}{m} \Rightarrow \frac{dv}{dt} = g - \frac{k}{m} \cdot v$$

Ao aplicarmos o método de Euler obtemos a seguinte equação geral:

$$\Delta v = \left(g - \frac{k}{m} v\right)\Delta t \Rightarrow v(t + \Delta t) = v(t) + \left(g - \frac{k}{m} v\right)\Delta t$$

Agora devemos separar as componentes da velocidade no plano *x* e no plano *y,* pois existem forças diferentes atuando em cada plano.

$$v_x(0) = v(0) \cdot cos(\theta) \qquad v_y(0) = v(0) \cdot sen(\theta)$$

$$g_x = 0 \Rightarrow v_x(t + \Delta t) = v_x(t) - \left(\frac{k}{m} v_x\right)\Delta t$$

$$g_y = g \Rightarrow v_y(t + \Delta t) = v_y(t) + \left(g - \frac{k}{m} v_y\right)\Delta t$$

A partir da equação acima podemos determinar as equações da posição, desconsiderando o termo multiplicado por Δt^2, pois ele é muito pequeno e pode ser desconsiderado:

$$\frac{dx}{dt} = v\ (t + \Delta t) \Rightarrow \frac{\Delta x}{\Delta t} = v\ (t + \Delta t)$$

$$\Delta x = v\ (t + \Delta t)\Delta t$$

$$S_x(t + \Delta t) = S_x(t) + v_x(t)\Delta t$$

$$S_y(t + \Delta t) = S_y(t) + v_y(t)\Delta t$$

A energia cinética é definida pela fórmula:

$$Ec = \frac{mv^2}{2}$$

Energia potencial é definida pela fórmula:

$$Ep = mgy$$

Energia perdida em calor é definida pela fórmula:

$$Q = E - Ec - Ep$$

Energia total, nesse caso, sem perda:

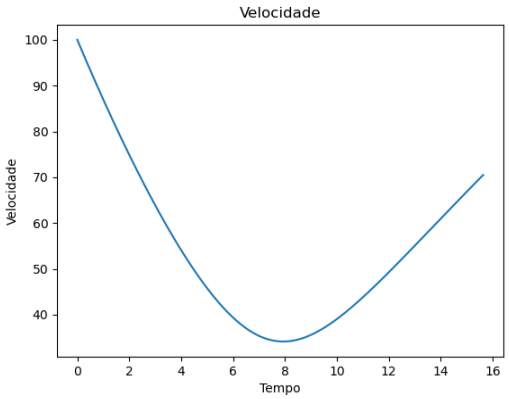
$$E = Q + Ec + Ep$$

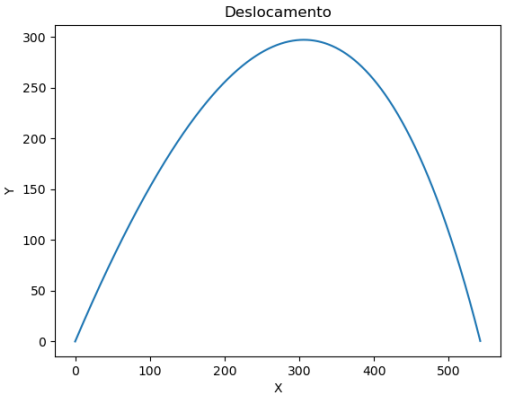
Resultados e Discussão

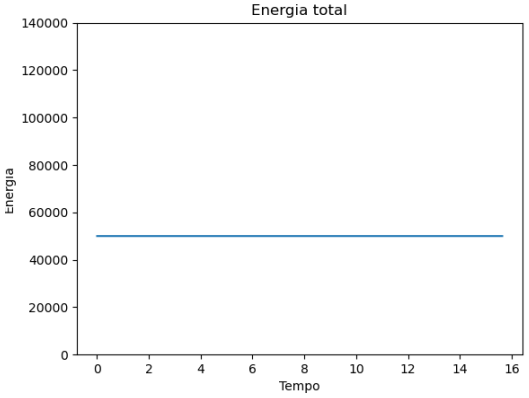
Caso 1: O objeto para ao tocar o chão

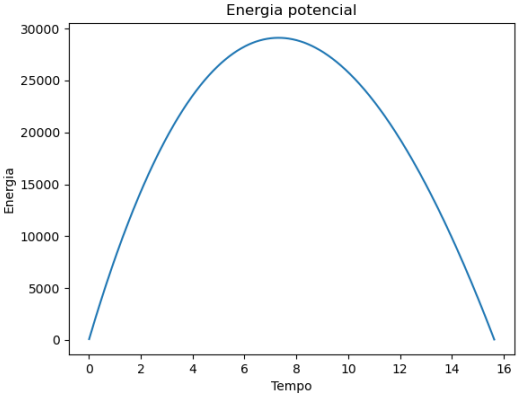
Parâmetros utilizados:

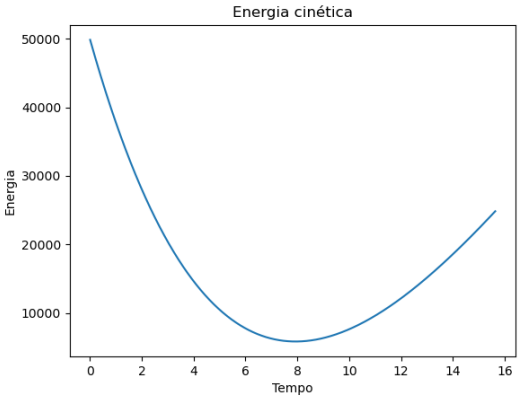
$$m = 10\,kg;\ \theta = 60°;\ v_0 = 100\,m/s;\ S_0(x, y) = (0, 0)$$

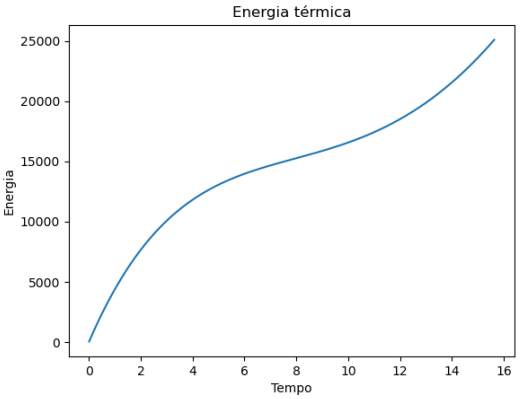








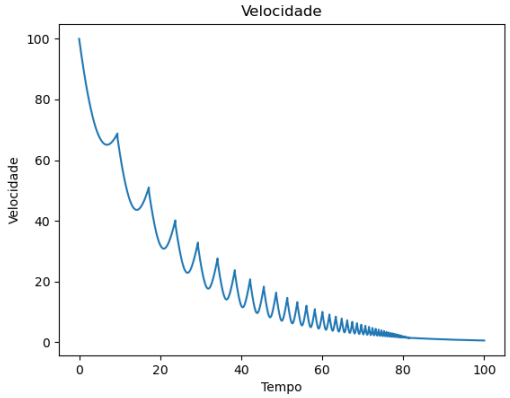


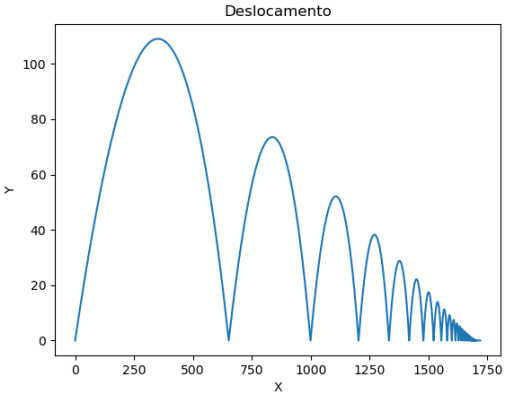


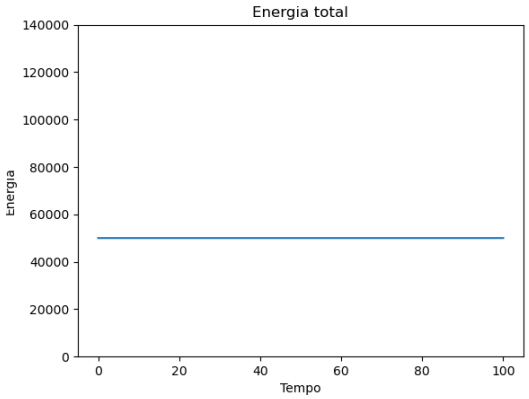
Caso 2 - O objeto quica ao tocar o chão e perde 5% de velocidade

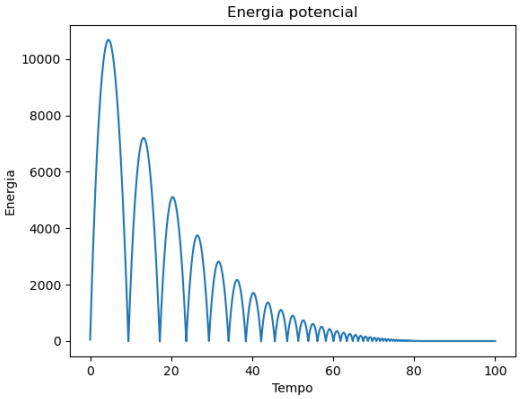
Parâmetros utilizados:

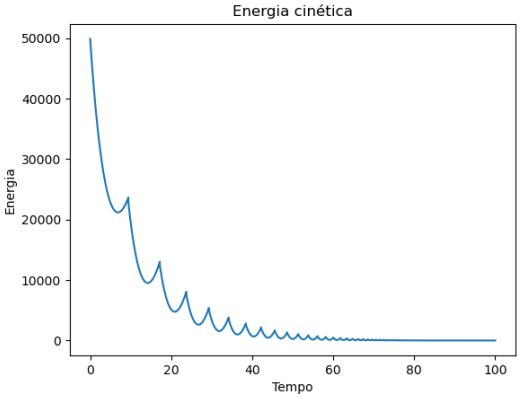
$$m = 10\,kg;\ \theta = 30°;\ v_0 = 100\,m/s;\ S_0(x, y) = (0, 0)$$

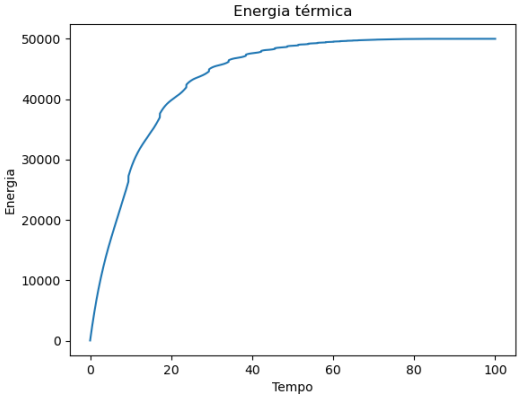








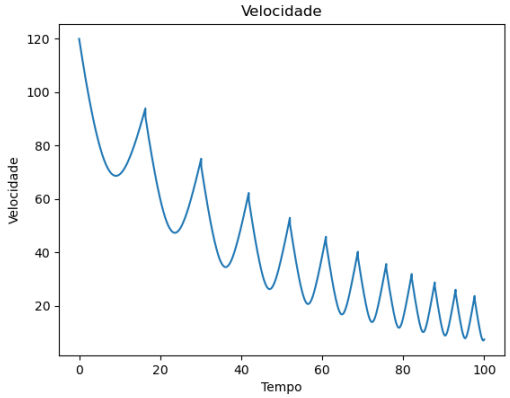


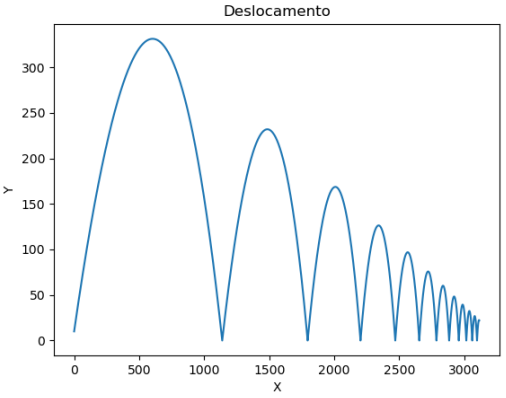


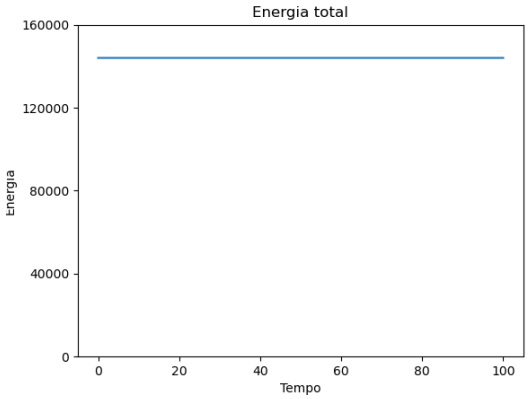
Caso 3 - O objeto quica ao tocar o chão e perde 5% de velocidade

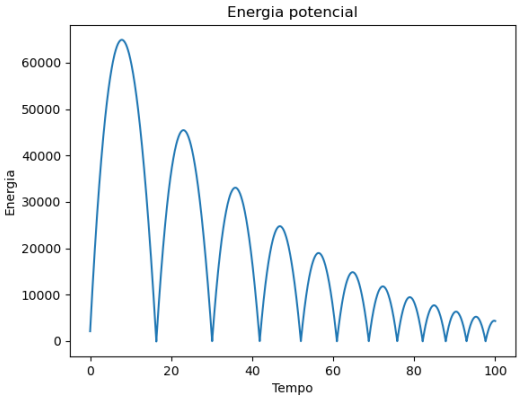
Parâmetros utilizados:

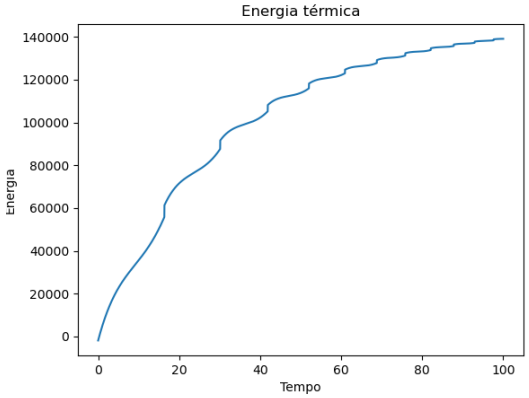
$$m = 20\,kg;\ \theta = 45°;\ v_0 = 120\,m/s;\ S_0(x, y) = (0, 0)$$

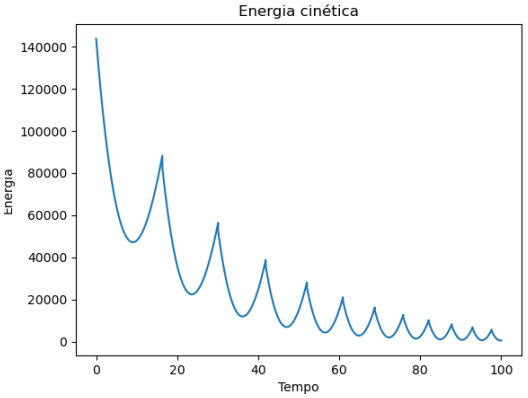








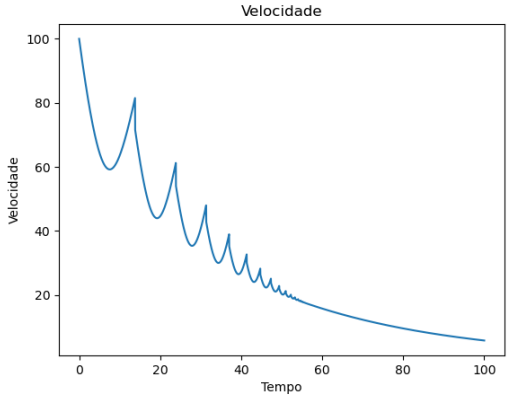


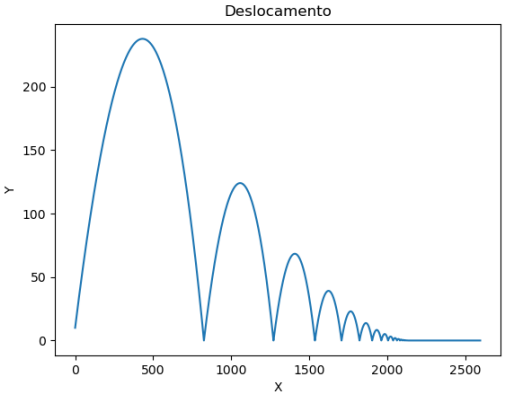


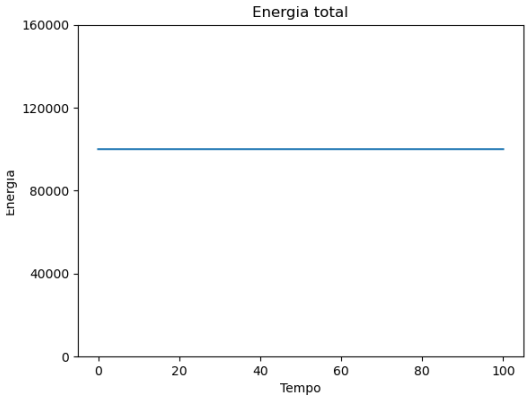
Caso 4 - O objeto quica ao tocar o chão e perde 20% de velocidade

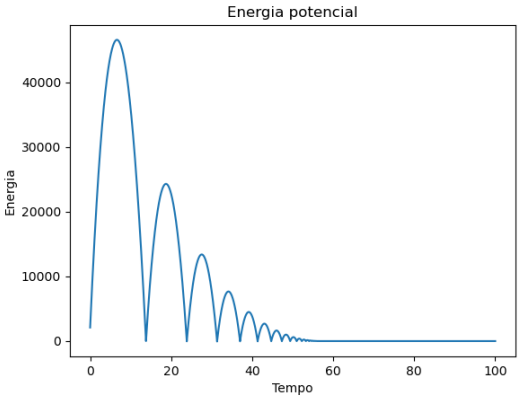
Parâmetros utilizados:

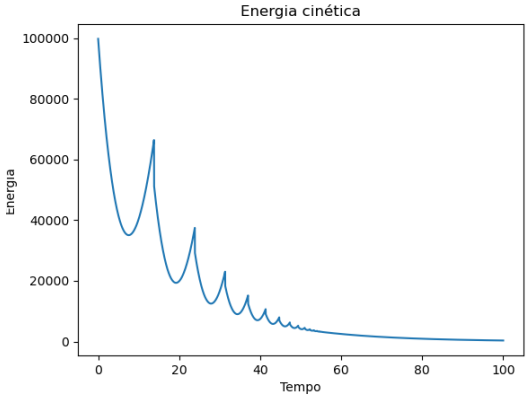
$$m = 20\,kg;\ \theta = 45°;\ v_0 = 100\,m/s;\ S_0(x, y) = (0, 10)$$

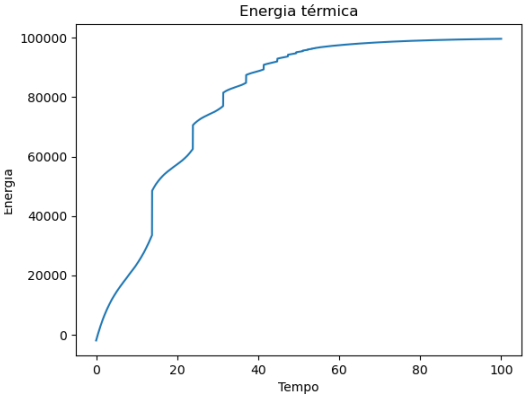








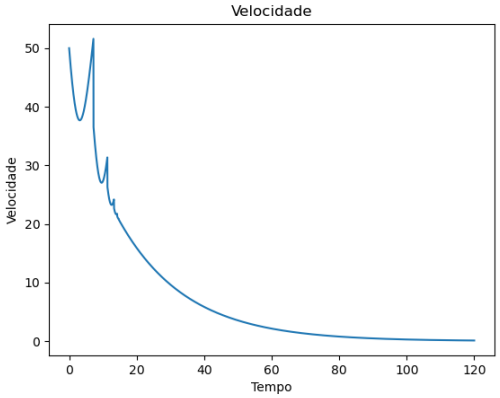


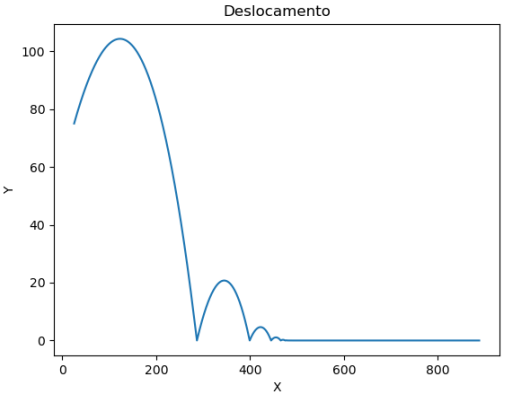


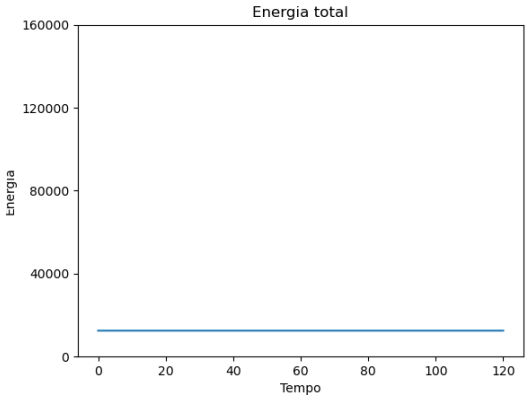
Caso 5 - O objeto quica ao tocar o chão e perde 50% de velocidade

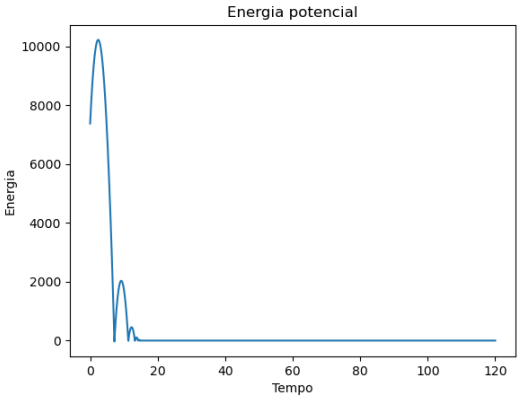
Parâmetros utilizados:

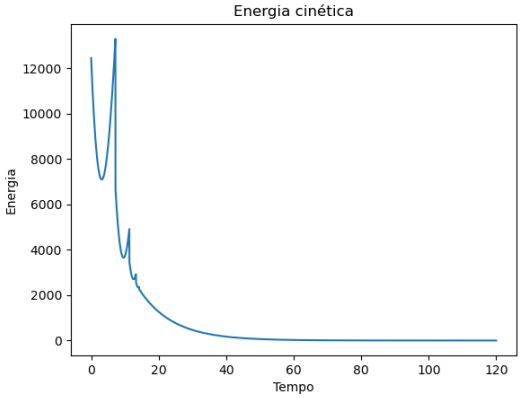
$$m = 10\,kg;\ \theta = 30°;\ v_0 = 50\,m/s;\ S_0(x, y) = (25, 75)$$

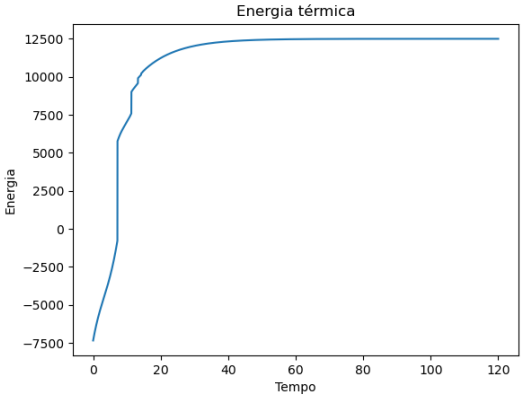












Anexo 1 - Implementação em Python 3

```
import numpy as np
import matplotlib.pyplot as plt
import math

DT = 0.01 #Delta T
TMax = 120 #Tempo de simulação
v0 = 50 #Velocidade Inicial
y0 = 75 #Posição Inicial em Y
x0 = 25 #Posição Inicial em X
k = 0.5 #Coeficiente de Atrito
g = -9.8 #Constante Gravitacional
theta = (math.pi)/6 #Ângulo de Lançamento
m = 10 #Massa do Corpo

vx = v0*(math.cos(theta)) #Velocidade Inicial em X
vy = v0*(math.sin(theta)) #Velocidade Inicial em Y
vx_array = [vx] #Vetor Velocidade em X
vy_array = [vy] #Vetor Velocidade em Y
v_array = [v0] #Vetor Velocidade em Total
Sy_array = [y0] #Vetor Posição em X
Sx_array = [x0] #Vetor Posição em Y
t_array = [0] #Vetor do tempo
Et_array = [None] #Vetor da Energia Total
Ep_array = [None] #Vetor da Energia Potencial
Ec_array = [None] #Vetor da Energia Total
Q_array = [None] #Vetor da Energia Térmica
Ec = 0.5*m*math.pow(v0,2)#calcula energia cinética
Et = Ec #Temos somente energia cinética no começo do lançamento
y = y0 # Atribuir posição inicial
x = x0 # Atribuir posição inicial
```

```
#Loop do tempo
for t in np.arange(DT, TMax, DT):
    # Velocidade em x
    vx = vx+((-k/m)*vx)*DT

    # Velocidade em y
    vy = vy+(g-((k/m)*vy))*DT

    # Velocidade Total
    v = math.sqrt(math.pow(vx,2) + math.pow(vy,2))

    # Deslocamento em y
    y = y + vy*(DT)

    # Deslocamento em x
    x = x + vx*(DT)

    # Energia cinética
    Ec = 0.5*m*math.pow(v,2)

    # Energia potencial
    Ep = m*(-g)*y #modulo de g

    # Energia térmica(calor)/Energia perdida
    Q = Et - (Ec+Ep)

    # Energia total
    Et =  Q + Ec + Ep

    #Condicional para rebater o corpo quando atinge o chão
    if y <= 0:
        vy = -vy*0.5 # Escolha da perda de velocidade
        y = 0

    #if y==0:
        #break
```

```
#Atribuição de valores nos vetores
    Sx_array.append(x)
    Sy_array.append(y)
    vx_array.append(vx)
    vy_array.append(vy)
    Et_array.append(Et)
    Ep_array.append(Ep)
    Ec_array.append(Ec)
    Q_array.append(Q)
    v_array.append(v)
    t_array.append(t)

#Funções Gráfico
plt.plot(t_array, v_array)
plt.title('Velocidade')
plt.xlabel('Tempo')
plt.ylabel('Velocidade')
plt.show()
plt.plot(Sx_array, Sy_array)
plt.title('Deslocamento')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
plt.plot(t_array, Et_array)
plt.title('Energia total')
plt.xlabel('Tempo')
plt.ylabel('Energia')
plt.yticks (np.arange(0, 200000, 40000))
#Marcadores para melhor compreensão do gráfico
plt.show()
plt.plot(t_array, Ep_array)
plt.title('Energia potencial')
plt.xlabel('Tempo')
plt.ylabel('Energia')
plt.show()
plt.plot(t_array, Ec_array)
plt.title('Energia cinética')
plt.xlabel('Tempo')
plt.ylabel('Energia')
plt.show()
```

```
plt.plot(t_array, Q_array)
plt.title('Energia térmica')
plt.xlabel('Tempo')
plt.ylabel('Energia')
plt.show()
```