Modelação de Sistemas Físicos - Aula Prática nº8

Realização e resolução de problemas sobre:

- Forças conservativas

- Conservação de energia

Problema 1

Uma bola de massa $M = 0.2\,\mathrm{kg}$ é rolada por uma pista cuja forma é dada pela função

$$y(x) = \begin{cases} (11 - x)\,\mathrm{m}, & 0 \leq x < 10\,\mathrm{m} \\ 1\,\mathrm{m}, & x \geq 10\,\mathrm{m} \end{cases}$$

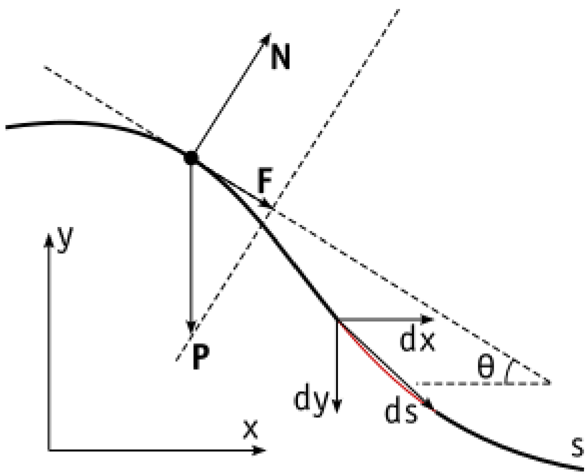
A bola acelera devido à força da gravidade, com aceleração que depende do declive da pista.

1. Faça uma simulação do movimento da bola para $t = [0,\ 3]\,\mathrm{s}$, usando o método de Euler-Cromer com as seguintes condições inicias:

$$\begin{aligned} x(t=0) &= 0\,\mathrm{m} \\ v(t=0) &= 0\,\mathrm{m/s} \end{aligned}$$

Solução:

Iniciamos com a descrição vetorial do problema com a ajuda da seguite figura, que é válida para qualquer relação $y(x)$, incluindo um plano com inclinação constante,



da qual rapidamente chegamos à conclusão que existem duas forças a atuar no corpo: (i) o peso $\mathbf{P}(x)$ e (ii) a força *normal* $\mathbf{N}(x)$ que *obriga* o corpo a percorrer a trajetória $y(x)$.

$$\mathbf{F} = \mathbf{P} + \mathbf{N},$$

onde

$$\mathbf{P} = -Mg\,\hat{\mathbf{j}}$$

e

$$\mathbf{N} = Mg\,\cos\theta\,(\sin\theta\,\hat{\mathbf{i}} + \cos\theta\,\hat{\mathbf{j}})$$

em que θ depende da posição, $\theta \equiv \theta(x)$. Após alguma manipulação algébrica chegamos a

$$\mathbf{F} = Mg\,\sin\theta\,(\cos\theta\,\hat{\mathbf{i}} - \sin\theta\,\hat{\mathbf{j}})$$

ou seja,

$$\mathbf{a} = \frac{\mathbf{F}}{M} = a\,\hat{\mathbf{a}}$$

onde

$$a = g\,\sin\theta$$

e $\hat{\mathbf{a}}$ é um vetor (unitário) com a direção da aceleração,

$$\hat{\mathbf{a}} = \cos\theta\,\hat{\mathbf{i}} - \sin\theta\,\hat{\mathbf{j}}.$$

Esta é uma solução geral, em que θ pode ser obtido a partir da derivada de y em x. De acordo com a figura em cima podemos concluir que

$$\theta(x) = -\arctan\left(\frac{\mathrm{d}y}{\mathrm{d}x}\right)$$

que também se aplica na descrição de um objeto que percorre um plano inclinado sob o efeito da gravidade (em que θ é neste caso uma constante).

Se desejarmos resolver o problema a uma dimensão, podemos *trabalhar* com os módulos da aceleração (a), da velocidade (v) e o espaço percorrido (s),

$$a = g\,\sin\theta$$

$$a = \frac{\mathrm{d}v}{\mathrm{d}t}$$

$$v = \frac{\mathrm{d}s}{\mathrm{d}t}.$$

Finalmente podemos obter as coordenadas Cartesianas por integração do espaço percorrido, a partir de

$$\mathrm{d}x = \mathrm{d}s\,\cos\theta$$

$$\mathrm{d}y = -\mathrm{d}s\,\sin\theta$$

Resolução do presente problema:

Neste caso o plano tem inclinação constante e o problema tem solução anaítica simples. Vamos obte-la de forma a podermos comparar com a solução numérica. Começamos com as coordenadas Cartesianas,

$$y(x) = \begin{cases} (11 - x)\,\mathrm{m}, & 0 \le x < 10\,\mathrm{m} \\ 1\,\mathrm{m}, & x \ge 10\,\mathrm{m} \end{cases}$$

A equação do movimento de um objeto que desce um plano inclinado sem atrito é uniformemente acelerado, e portanto,

$$s(t) = \frac{1}{2}\, a\, t^2$$

onde $a = g\, \sin\theta$ e assumindo que $s(t=0) = 0$ e $v(t=0) = 0$. Dado que o declive de $y(x)$ é igual a -1, o ângulo é simplesmente $\theta = 45^\circ$, a partir do qual chegamos a $a = g/\sqrt{2}$, e portanto

$$s(t) = g\, t^2/2\sqrt{2},$$

e

$$v(t) = \frac{\mathrm{d}s}{\mathrm{d}t} = g\, t/\sqrt{2}$$

onde assumimos que $v(t=0) = 0\,\mathrm{m/s}$.

Dado que $\Delta x = \Delta s\, \cos\theta$, a bola chega a $x = 10\,\mathrm{m}$ quando percorre

$$s(x=10) = 10/\cos\theta = 14.14\,\mathrm{m}.$$

Isso ocorre ao fim de,

$$t = 2\sqrt{\frac{10}{g}} = 2.02\,\mathrm{s},$$

e apartir daqui o movimento é uniforme (velocidade constante), com

$$v(t > 2.02\,\mathrm{s}) = \frac{g}{\sqrt{2}} \times 2\sqrt{\frac{10}{g}} = 14\,\mathrm{m/s}.$$

Ao fim de $t = 3\,\mathrm{s}$, a bola percorreu mais $\Delta s = v(t > 2.02\,\mathrm{s})\Delta t$ com movimento uniforme durante $\Delta t = 3 - 2\sqrt{10/g} = 0.98\,\mathrm{s}$, ou seja

$$\Delta s = 13.72\,\mathrm{m}$$

e finalmente, o espaço total percorrido é dado por

$$s = s(x=10) + \Delta s = 14.14\,\mathrm{m} + 13.72\,\mathrm{m} = 27.86\,\mathrm{m}$$

A solução numérica para o problema (Método de Euler-Cromer) inicia-se com o cálculo da aceleração, $a(x) = g\, \sin\theta(x)$ ao longo da trajetória $y(x)$. Para isso necessitamos da derivada dy/dx,

$$\frac{\mathrm{d}y(x)}{\mathrm{d}x} = \begin{cases} -1, & 0 \le x < 10\,\mathrm{m} \\ 0, & x \ge 10\,\mathrm{m} \end{cases}$$

Quando $x < 10\,\mathrm{m}$ o ângulo é dado por $\theta(x) = -\arctan(-1) = \pi/4$. Desta forma $\sin\theta = \sqrt{2}/2$, e portanto

$$a = \begin{cases} g\sqrt{2}/2, & 0 \le x < 10\,\mathrm{m} \\ 0, & x \ge 10\,\mathrm{m} \end{cases}$$

A solução pelo método de Euler-Cromer segue então a receita usual:

$$\frac{\mathrm{d}v}{\mathrm{d}t} = a$$

$$\frac{\mathrm{d}s}{\mathrm{d}t} = v$$

O que na forma discreta se traduz em:

$$v_{i+1} = v_i + a_i\,\delta\tau$$

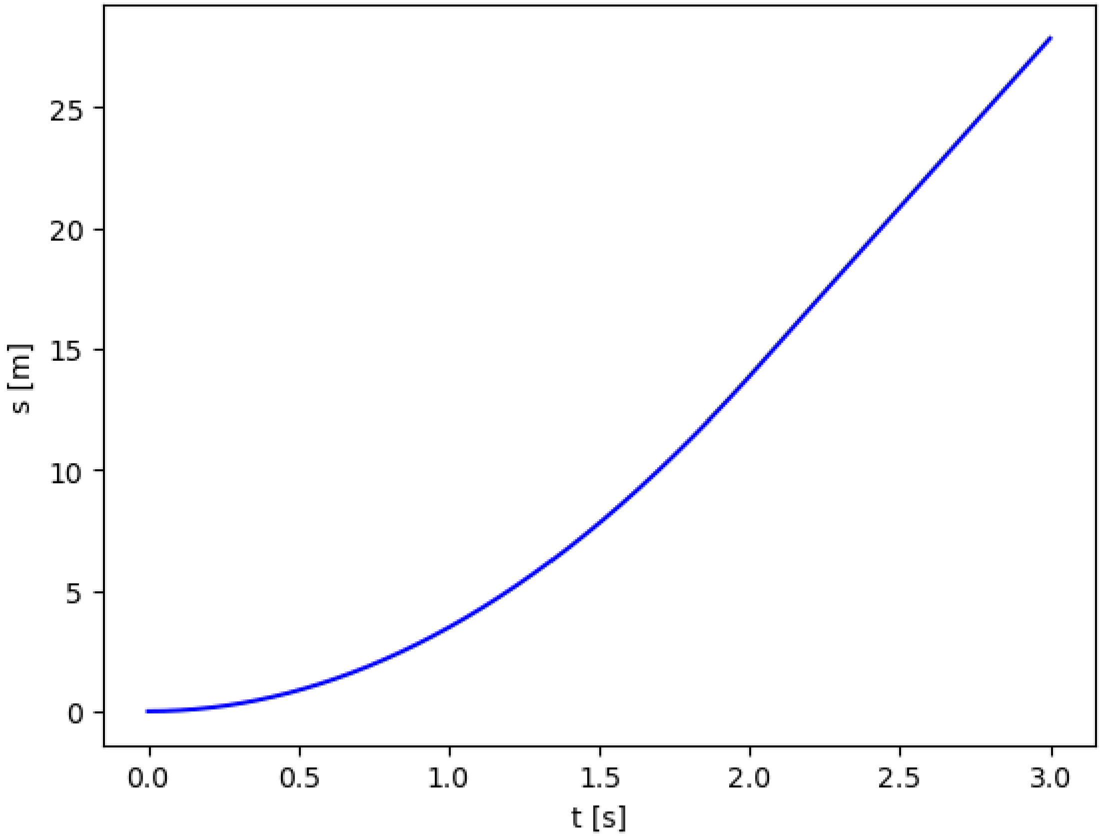
$$s_{i+1} = s_i + v_{i+1}\,\delta\tau$$

Note que utilizamos de v_{i+1} no cáculo da posição (no método de Euler usa v_i).

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt

t0 = 0.0                    # condição inicial, tempo [s]
tf = 3.0                    # limite do domínio, tempo final [s]
dt = 0.001                  # passo [s]
v0 = 0.0                    # condição inicial, módulo da velocidade inicial [m/s]
x0 = 0.0                    # condição inicial, coordenada x da posição inicial [m]
y0 = 11.0                   # condição inicial, coordenada y da posição inicial [m]
M = 0.2                     # Massa do corpo [kg]
g = 9.80665                 # aceleração gravitacional (valor standard) [m/s^2]

# Derivada de y em ordem a x
# y(x) = 11 - x para x < 10 de outra forma y(x) = 0
# dy/dx = -1 de outra forma dy/dx = 0
def dydx_func(x: float) -> float:
    return -1 if x < 10.0 else 0.0

# inicializar domínio [ano]
t = np.arange(t0, tf, dt)

# inicializar solução, aceleração a 1D [m/s^2]
a = np.zeros(np.size(t))

# inicializar solução, velocidade [m/s]
v = np.zeros(np.size(t))
v[0] = v0

# inicializar solução, posição [m]
s = np.zeros(np.size(t))
s[0] = x0
x = np.zeros(np.size(t))
y = np.zeros(np.size(t))
x[0] = x0
y[0] = y0

theta = np.zeros(np.size(t))

for i in range(np.size(t) - 1):

    # ângulo θ
    theta[i] = -np.arctan(dydx_func(x[i]))

    # aceleração
    a[i] = g * np.sin(theta[i])

    # Método de Euler-Cromer
    v[i + 1] = v[i] + a[i] * dt
    s[i + 1] = s[i] + v[i + 1] * dt

    # posição carteziana
    x[i + 1] = x[i] + (s[i + 1] - s[i]) * np.cos(theta[i])
    y[i + 1] = y[i] - (s[i + 1] - s[i]) * np.sin(theta[i])
```

O espaço percorrido pela bola para $t = [0, 3]\,\mathrm{s}$

In [2]:
```
plt.plot(t, s, 'b-')
plt.xlabel("t [s]")
plt.ylabel("s [m]")
plt.show()
print("Espaço percorrido, s(t = 3 s) = {0:.2f} m".format(s[-1]))
print("Alcançe, x(t = 3 s) = {0:.2f} m".format(x[-1]))
```




```
Espaço percorrido, s(t = 3 s) = 27.87 m
Alcançe, x(t = 3 s) = 23.72 m
```

2. Calcule a velocidade final.

In [3]:
```
v1 = v
print("A velocidade final v = {0:.2f} m/s²".format(v1[-1]))
```

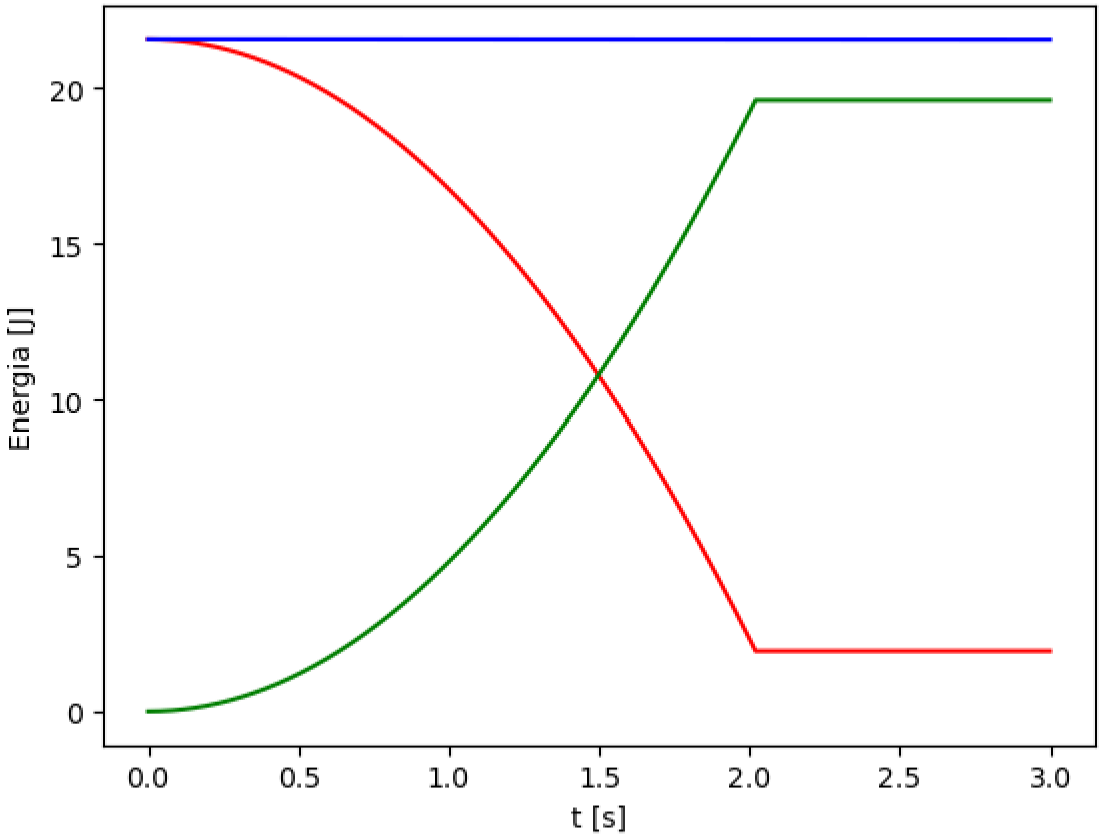
```
A velocidade final v = 14.01 m/s²
```

3. Faça um gráfico da energia potencial, E_p, da energia cinética, E_c, e a energia total, E_t em função do tempo.

$$\begin{aligned} E_\mathrm{p} &= M g y \\ E_\mathrm{c} &= \frac{1}{2} M v^2 \\ E_\mathrm{t} &= E_\mathrm{p} + E_\mathrm{c} \end{aligned}$$

In [4]:
```
x1 = x # guardar para mais tarde
y1 = y

# Energia potencial
E_p1 = M * g * y1

# Energia cinética
E_c1 = 0.5 * M * v1 ** 2

# Energia total
E_t1 = E_p1 + E_c1
```

```
plt.plot(t, E_p1, 'r-', t, E_c1, 'g-', t, E_t1, 'b-')
plt.xlabel("t [s]")
plt.ylabel("Energia [J]")
plt.show()
```



4. Repita o exercício anterior para cada das seguintes trajetórias:

Trajetória parabólica:

$$y(x) = \begin{cases} [0.1(x-10)^2 + 1]\ \mathrm{m}, & 0 \leq x < 10\ \mathrm{m} \\ 1\ \mathrm{m}, & x \geq 10\ \mathrm{m} \end{cases}$$

Trajetória cúbica

$$y(x) = \begin{cases} [-0.04x^3 + 0.9x^2 - 6x + 11]\ \mathrm{m}, & 0 \leq x < 10\ \mathrm{m} \\ 1\ \mathrm{m}, & x \geq 10\ \mathrm{m} \end{cases}$$

Solução para a trajetória parabólica:

Usamos novamente a solução geral para a aceleração,

$$a = g\,\sin\theta$$

em que

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \begin{cases} (x-10)/5, & 0 \leq x < 10\ \mathrm{m} \\ 0\ \mathrm{m}, & x \geq 10\ \mathrm{m} \end{cases}$$

e obtemos o ângulo θ a partir de

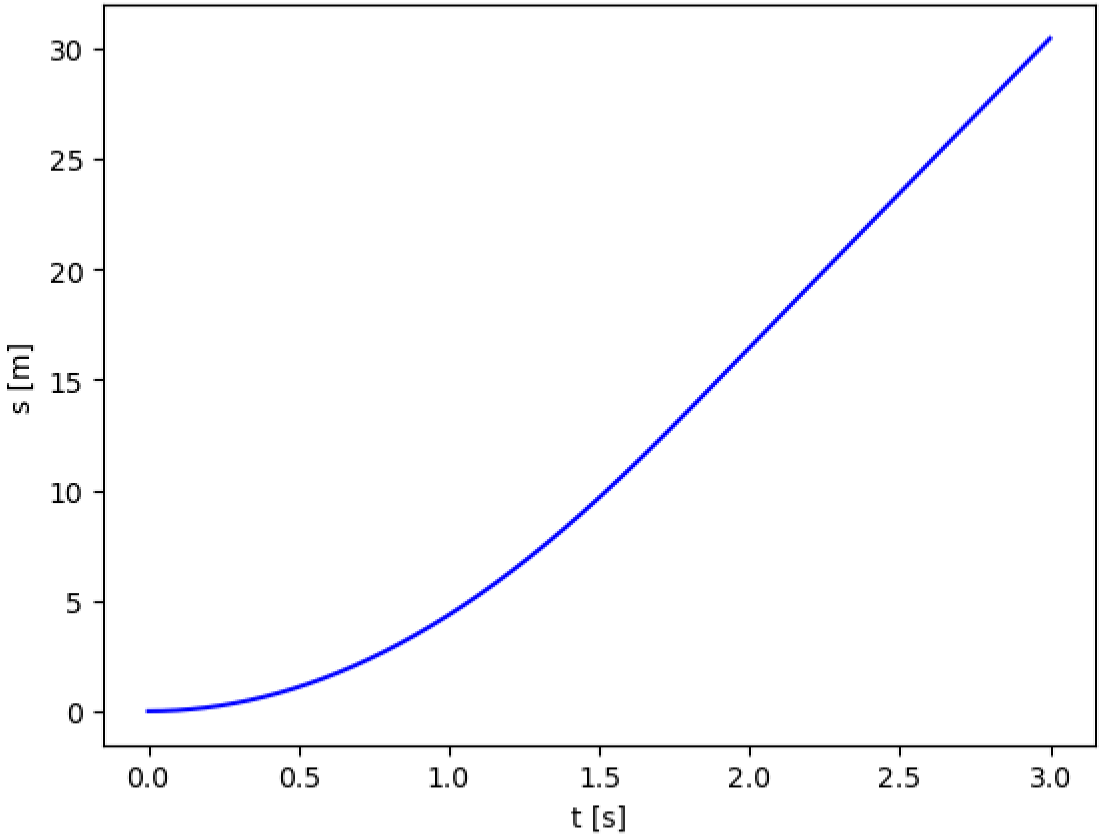
$$\theta(x) = -\arctan\left(\frac{dy}{dx}\right)$$

In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
```

```
t0 = 0.0                    # condição inicial, tempo [s]
tf = 3.0                    # limite do domínio, tempo final [s]
dt = 0.001                  # passo [s]
v0 = 0.0                    # condição inicial, módulo da velocidade inicial [m/s]
x0 = 0.0                    # condição inicial, coordenada x da posição inicial [m]
y0 = 11.0                   # condição inicial, coordenada y da posição inicial [m]
M = 0.2                     # Massa do corpo [kg]
g = 9.80665                 # aceleração gravitacional (valor standard) [m/s^2]

# Derivada de y em ordem a x
# y(x) = (x - 10)^2 / 10 + 1 para x < 10 de outra forma y(x) = 0
# dy/dx = (x - 10) / 5 de outra forma dy/dx = 0
def dydx_func(x: float) -> float:
    return (x - 10.0) / 5.0 if x < 10.0 else 0.0

# inicializar domínio [ano]
t = np.arange(t0, tf, dt)

# inicializar solução, aceleração a 1D [m/s^2]
a = np.zeros(np.size(t))

# inicializar solução, velocidade [m/s]
v = np.zeros(np.size(t))
v[0] = v0

# inicializar solução, posição [m]
s = np.zeros(np.size(t))
s[0] = x0
x = np.zeros(np.size(t))
y = np.zeros(np.size(t))
x[0] = x0
y[0] = y0

theta = np.zeros(np.size(t))

for i in range(np.size(t) - 1):

    # ângulo θ
    theta[i] = -np.arctan(dydx_func(x[i]))

    # aceleração
    a[i] = g * np.sin(theta[i])

    # Método de Euler-Cromer
    v[i + 1] = v[i] + a[i] * dt
    s[i + 1] = s[i] + v[i + 1] * dt

    # posição carteziana
    x[i + 1] = x[i] + (s[i + 1] - s[i]) * np.cos(theta[i])
    y[i + 1] = y[i] - (s[i + 1] - s[i]) * np.sin(theta[i])
```

O espaço percorrido pela bola para $t = [0, 3]$ s

In [6]:
```
plt.plot(t, s, 'b-')
plt.xlabel("t [s]")
plt.ylabel("s [m]")
plt.show()
print("Espaço percorrido, s(t = 3 s) = {0:.2f} m".format(s[-1]))
print("Alcançe, x(t = 3 s) = {0:.2f} m".format(x[-1]))
```




```
Espaço percorrido, s(t = 3 s) = 30.45 m
Alcançe, x(t = 3 s) = 25.65 m
```

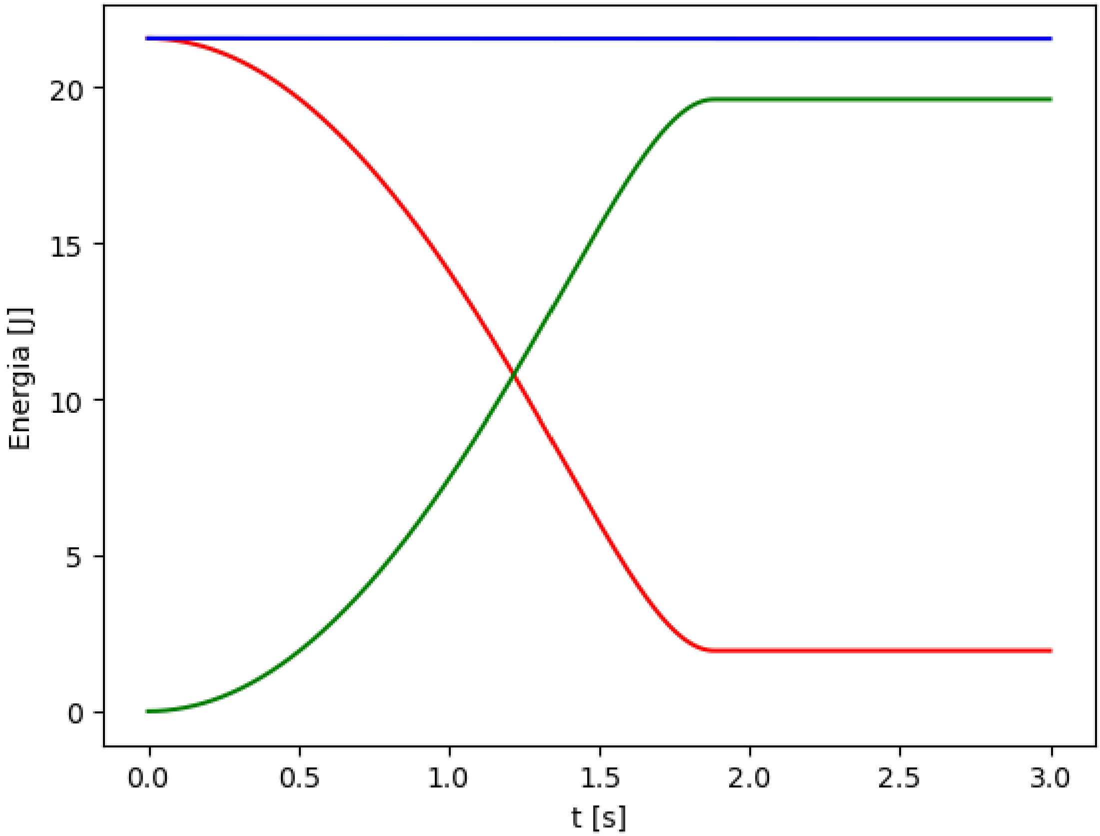
In [7]:

```
v2 = v
print("A velocidade final v = {0:.2f} m/s²".format(v2[-1]))
```

```
A velocidade final v = 14.01 m/s²
```

In [8]:

```
x2 = x # guardar para mais tarde
y2 = y

# Energia potencial
E_p2 = M * g * y2

# Energia cinética
E_c2 = 0.5 * M * v2 ** 2

# Energia total
E_t2 = E_p2 + E_c2

plt.plot(t, E_p2, 'r-', t, E_c2, 'g-', t, E_t2, 'b-')
plt.xlabel("t [s]")
plt.ylabel("Energia [J]")
plt.show()
```



Solução para a trajetória parabólica:

Usamos novamente a solução geral para a aceleração,

$$a = g\,\sin\theta$$

em que

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \begin{cases} -0.12x^2 + 1.8x - 6, & 0 \leq x < 10\,\mathrm{m} \\ 0\,\mathrm{m}, & x \geq 10\,\mathrm{m} \end{cases}$$

e obtemos o ângulo θ a partir de

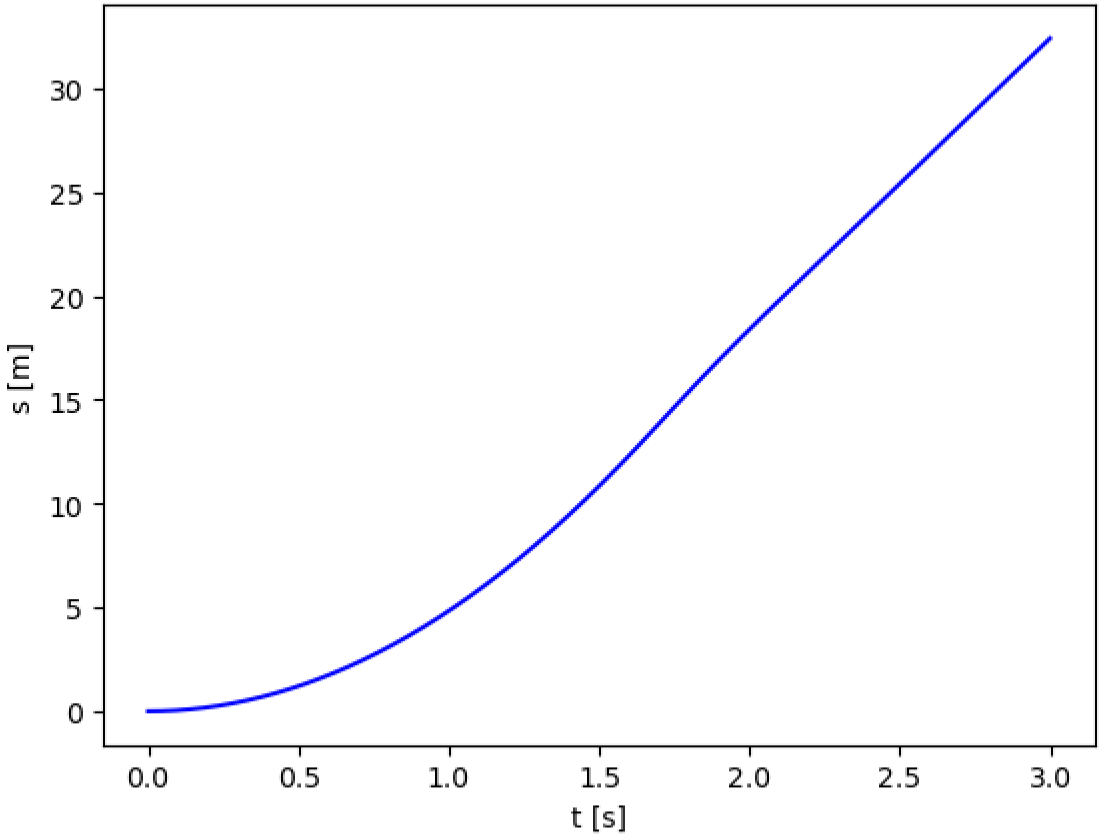
$$\theta(x) = -\arctan\left(\frac{dy}{dx}\right)$$

In [9]:
```
import numpy as np
import matplotlib.pyplot as plt

t0 = 0.0                    # condição inicial, tempo [s]
tf = 3.0                    # limite do domínio, tempo final [s]
dt = 0.001                  # passo [s]
v0 = 0.0                    # condição inicial, módulo da velocidade inicial [m/s]
x0 = 0.0                    # condição inicial, coordenada x da posição inicial [m]
y0 = 11.0                   # condição inicial, coordenada y da posição inicial [m]
M = 0.2                     # Massa do corpo [kg]
g = 9.80665                 # aceleração gravitacional (valor standard) [m/s^2]

# Derivada de y em ordem a x
# y(x) = -0.04 x^3 + 0.9 x^2 - 6 x + 11 para x < 10 de outra forma y(x) = 0
# dy/dx = -0.12 x^2 + 1.8 x - 6 de outra forma dy/dx = 0
def dydx_func(x: float) -> float:
    return -0.12 * x ** 2 + 1.8 * x - 6.0 if x < 10.0 else 0.0

# inicializar domínio [ano]
t = np.arange(t0, tf, dt)
```

```python
# inicializar solução, aceleração a 1D [m/s^2]
a = np.zeros(np.size(t))

# inicializar solução, velocidade [m/s]
v = np.zeros(np.size(t))
v[0] = v0

# inicializar solução, posição [m]
s = np.zeros(np.size(t))
s[0] = x0
x = np.zeros(np.size(t))
y = np.zeros(np.size(t))
x[0] = x0
y[0] = y0

theta = np.zeros(np.size(t))

for i in range(np.size(t) - 1):

    # ângulo θ
    theta[i] = -np.arctan(dydx_func(x[i]))

    # aceleração
    a[i] = g * np.sin(theta[i])

    # Método de Euler-Cromer
    v[i + 1] = v[i] + a[i] * dt
    s[i + 1] = s[i] + v[i + 1] * dt

    # posição carteziana
    x[i + 1] = x[i] + (s[i + 1] - s[i]) * np.cos(theta[i])
    y[i + 1] = y[i] - (s[i + 1] - s[i]) * np.sin(theta[i])
```

O espaço percorrido pela bola para $t = [0, 3]$ s

In [10]:
```python
plt.plot(t, s, 'b-')
plt.xlabel("t [s]")
plt.ylabel("s [m]")
plt.show()
print("Espaço percorrido, s(t = 3 s) = {0:.2f} m".format(s[-1]))
print("Alcançe, x(t = 3 s) = {0:.2f} m".format(x[-1]))
```

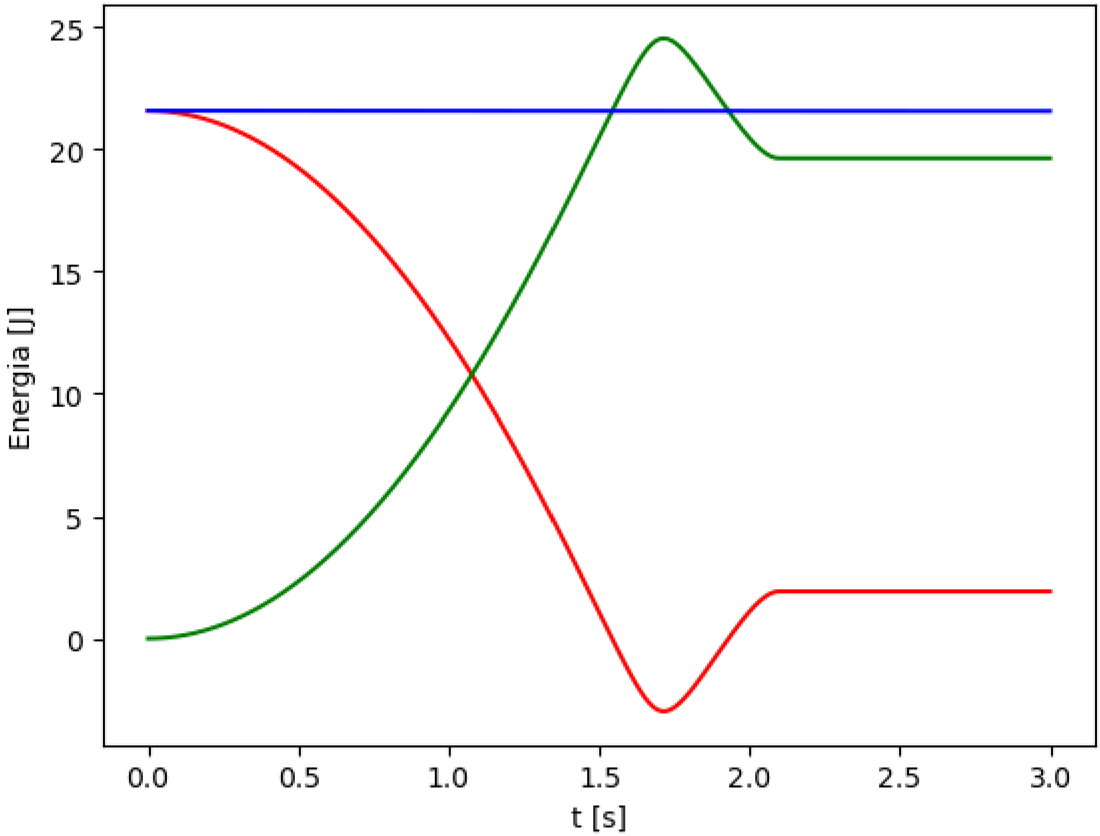



```
Espaço percorrido, s(t = 3 s) = 32.40 m
Alcançe, x(t = 3 s) = 22.64 m
```

In [11]:
```
v3 = v
print("A velocidade final v = {0:.2f} m/s²".format(v3[-1]))
```

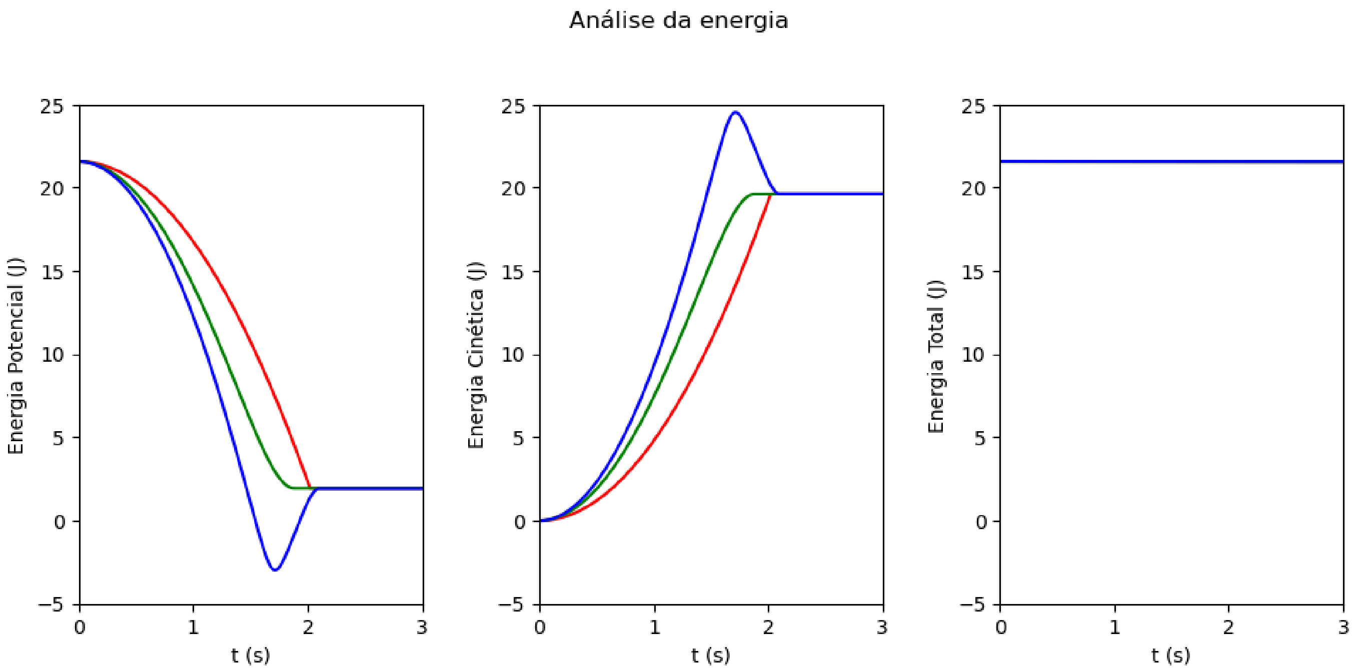
```
A velocidade final v = 14.01 m/s²
```

In [12]:
```
x3 = x # guardar para mais tarde
y3 = y

# Energia potencial
E_p3 = M * g * y3

# Energia cinética
E_c3 = 0.5 * M * v3 ** 2

# Energia total
E_t3 = E_p3 + E_c3

plt.plot(t, E_p3, 'r-', t, E_c3, 'g-', t, E_t3, 'b-')
plt.xlabel("t [s]")
plt.ylabel("Energia [J]")
plt.show()
```



5. Compare as energias cinéticas, energias potenciais e energias totais para as três trajetórias. O que conclui?

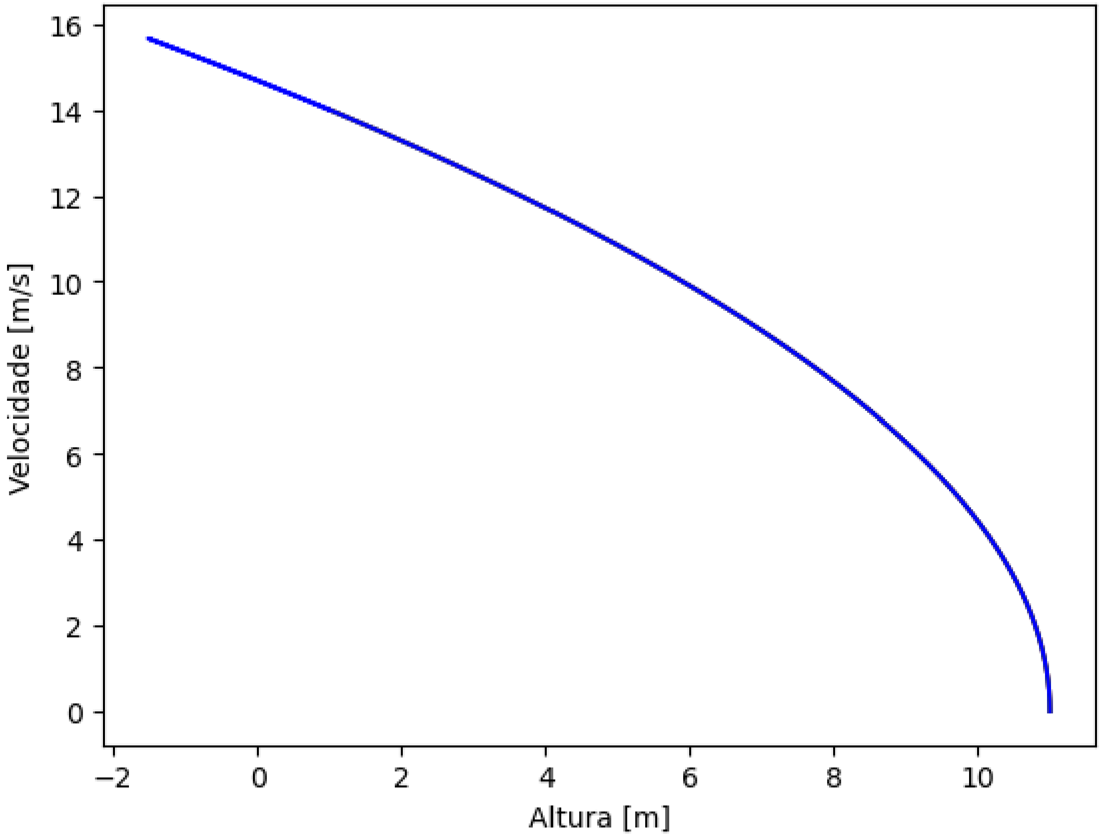
In [13]:
```
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(10, 5))
fig.suptitle('Análise da energia')

ax1.plot(t, E_p1, 'r-', t, E_p2, 'g-', t, E_p3, 'b-')
ax1.set_xlim([t0, tf])
ax1.set_ylim([-5, 25])
ax1.set_xlabel('t (s)')
ax1.set_ylabel('Energia Potencial (J)')

ax2.plot(t, E_c1, 'r-', t, E_c2, 'g-', t, E_c3, 'b-')
ax2.set_xlim([t0, tf])
ax2.set_ylim([-5, 25])
ax2.set_xlabel('t (s)')
ax2.set_ylabel('Energia Cinética (J)')

ax3.plot(t, E_t1, 'r-', t, E_t2, 'g-', t, E_t3, 'b-')
ax3.set_xlim([t0, tf])
ax3.set_ylim([-5, 25])
ax3.set_xlabel('t (s)')
ax3.set_ylabel('Energia Total (J)')

fig.tight_layout(pad=2.0)

plt.show()
```



- Energia potencial
  - de certa forma "reflete" a altura da bola
  - $E_\mathrm{p}(t=0)$ é idêntica para as três bolas (todas partem da mesma altura)
  - A bola azul (potencial mais "porfundo") acelera mais rapidamente
  - A bola vermelha (potencial do plano inclinado) tem a aceleração menor (movimento parabólico)
  - Iniciam e terminam todos com energia potencial idêntica
- Energia cinnética
  - Reflete a velocidade da bola
  - $E_\mathrm{c}(t \gtrsim 2)$ é idêntica para as três bolas (porque todas acabam com a mesma E_p)
  - A bola azul ganha mais velocidade (maior aceleração)
  - A bola vermelha ganha menor aceleração (plano inclinado)
  - Iniciam e terminam todos com energia cinética idêntica
- Energia total
  - É idêntica e conserva-se nos três casos

6. Faça o gráfico da velocidade v em função da altura y para cada forma da pista. Explique o resultado.

In [14]:
```
plt.plot(y1, v1, 'r-', y2, v2, 'g-', y3, v3, 'b-')
plt.xlabel("Altura [m]")
plt.ylabel("Velocidade [m/s]")
plt.show()
```



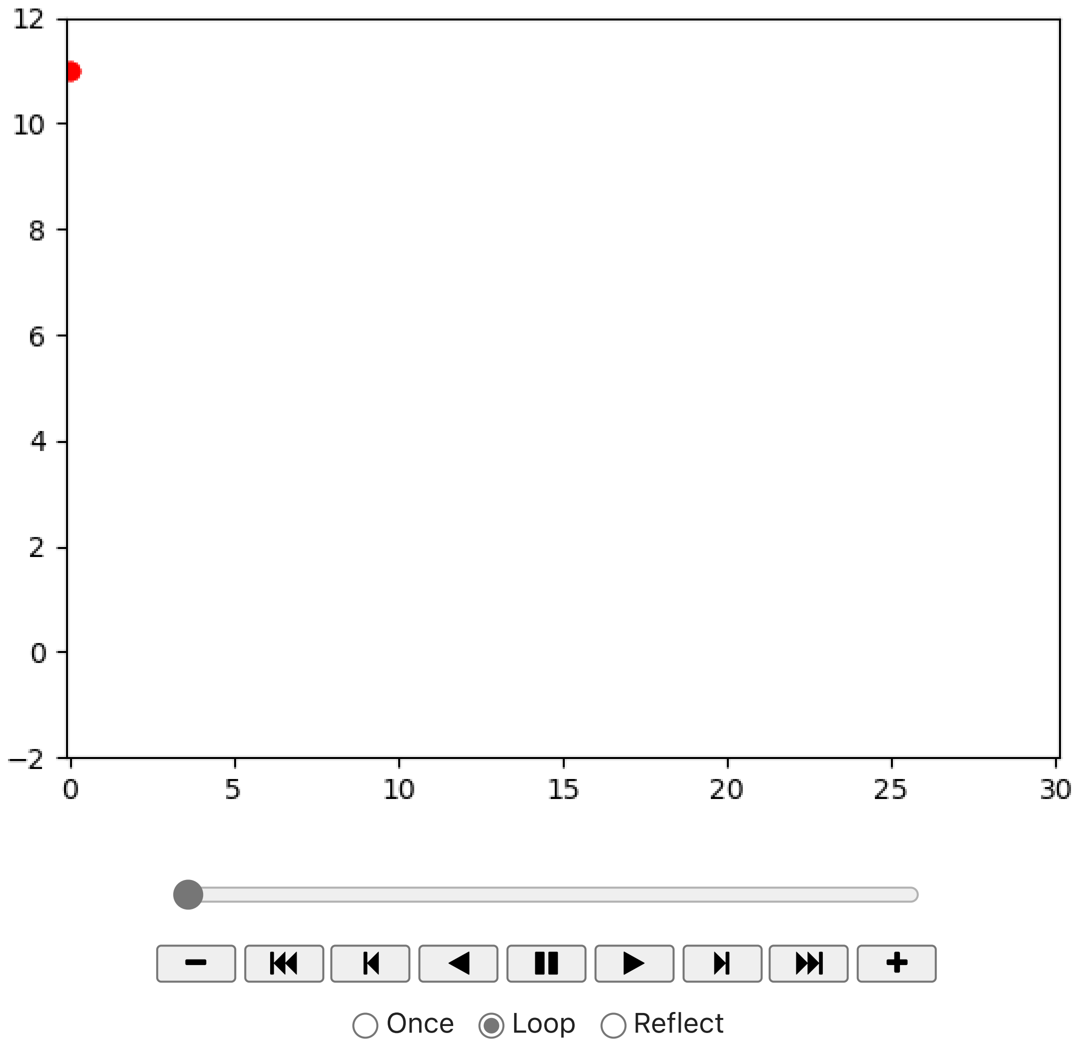
A velocidade em função da altura é idêntica para as três bolas porque para uma determinada altura $y = h$, a energia potencial é idêntica nas três bolas ($E_\mathrm{p} = Mgh$). Considerando a lei de conservação da energia, concluimos que as três energias cinéticas, e consequentemente as três velocidades, são idênticas.

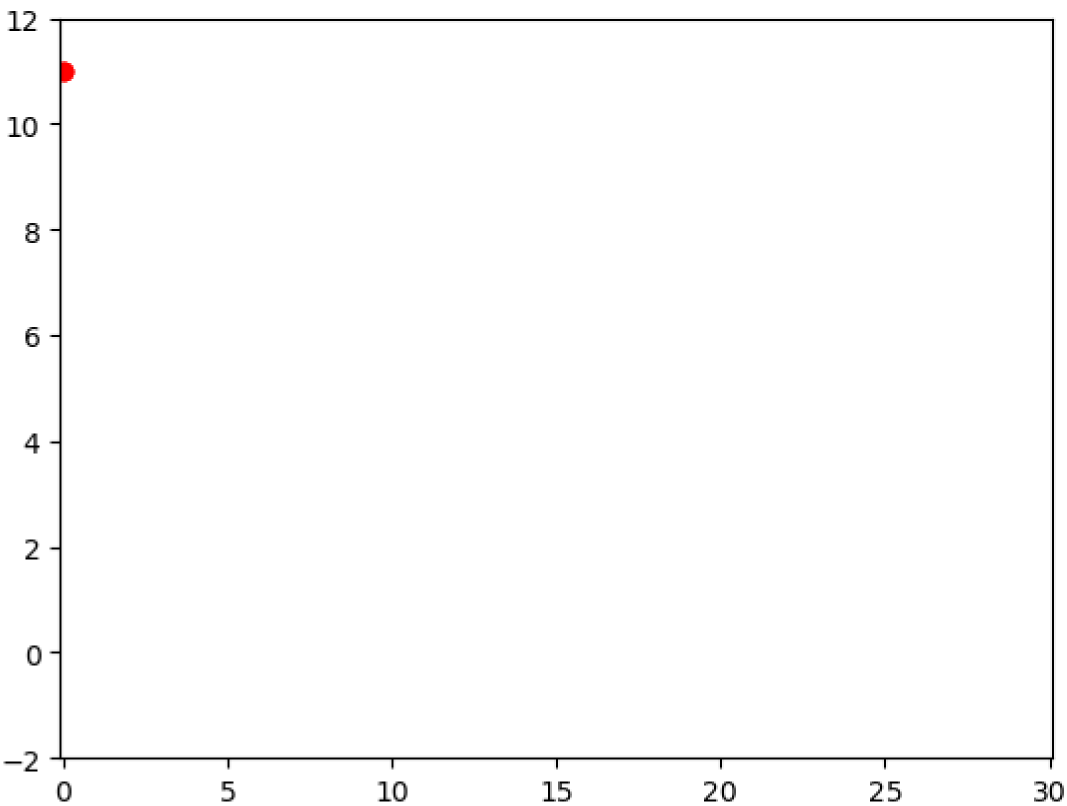
7. Faça uma animação do movimento da bola para cada forma da pista. Para qual delas é que a bola atinge $x = 10\ \mathrm{m}$ primeiro?

In [15]:

```
from matplotlib.animation import FuncAnimation
from IPython.display import HTML

fig = plt.figure()
ax = plt.axes(xlim=(-0.1, 30.1), ylim=(-2, 12))
bola = ax.plot([], [], 'ro', [], [], 'go', [], [], 'bo')[0]        # bola, posição in

def update(frame):
    # atualizar o plot da posição da bola
    bola.set_xdata([x1[frame], x2[frame], x3[frame]])
    bola.set_ydata([y1[frame], y2[frame], y3[frame]])
    return bola

nframes = 100
total_frames = np.size(t)
iframes = np.arange(0, total_frames, total_frames // nframes)
ani = FuncAnimation(fig=fig, func=update, frames=iframes, interval=100)

HTML(ani.to_jshtml())
```

Out[15]:





A bola que segue a trajetória parabólica chega a primeiro a $x = 10\ \mathrm{m}$.

Consegue inventar uma forma da pista que seja mais rápida?

O campo gravítico é conservativo, logo a energia potencial perdida na descida é completamente convertida para energia cinética. Desta forma, só é possível conceber uma pista com uma velocidade final mais elevada se a diferença entre a altura inicial, $y(t=0)$, e a altura final, $y(t=3\,\mathrm{s})$ for superior a $10\,\mathrm{m}$.