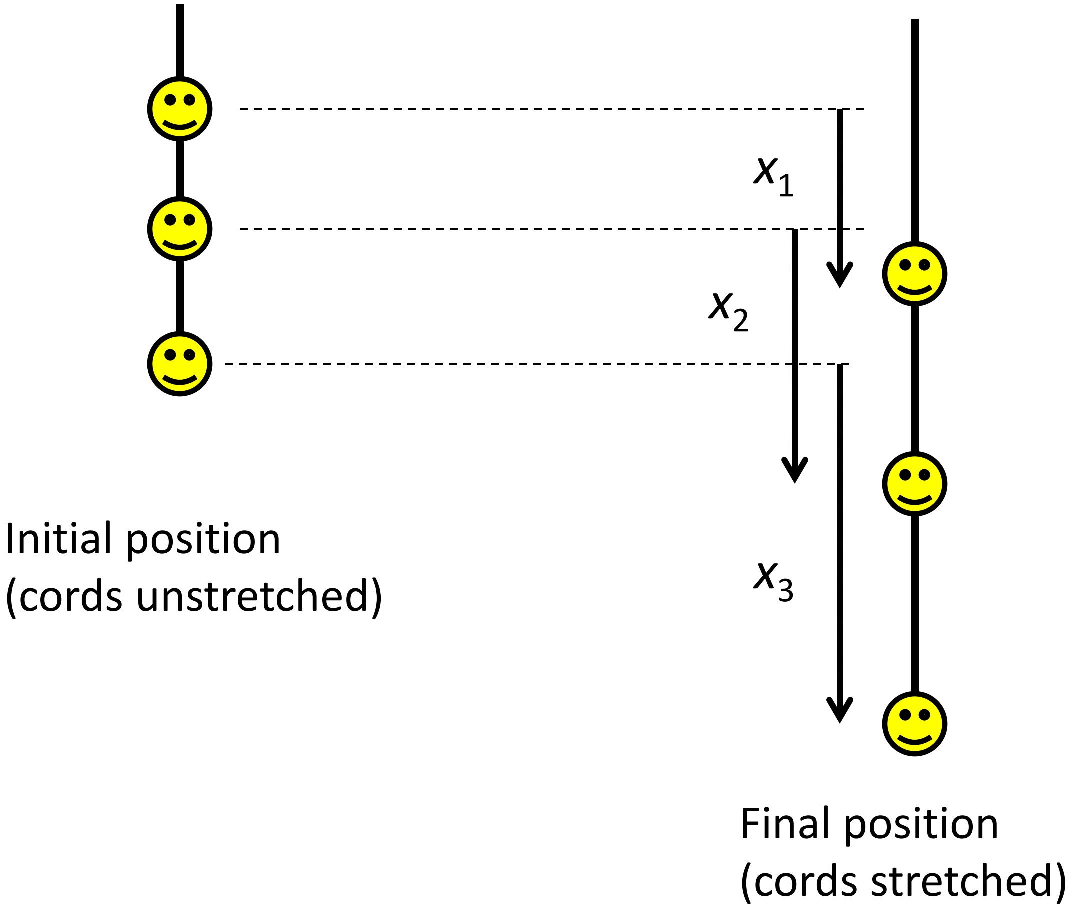
Problem (text p193): Three bungee jumpers are connected as shown in the diagram. How far will each jumper fall?



The three bungee cord segments have spring constants k_1, k_2, and k_3.
The three jumpers have masses m_1, m_2, and m_3.

The forces acting on each jumper must add up to zero.

Convention: downwards forces positive.

$$\text{for jumper 1}: m_1 g + k_2(x_2 - x_1) - k_1 x_1 = 0$$

$$\text{for jumper 2}: m_2 g + k_3(x_3 - x_2) - k_2(x_2 - x_1) = 0$$

$$\text{for jumper 3}: m_3 g - k_3(x_3 - x_2) = 0$$

Collecting terms and rearranging gives:

$$(k_1 + k_2)x_1 - k_2 x_2 = m_1 g$$

$$-k_2 x_1 + (k_2 + k_3)x_2 - k_3 x_3 = m_2 g$$

$$-k_3 x_2 + k_3 x_3 = m_3 g$$

These equations can be expressed in matrix format:

$$(k_1 + k_2)x_1 - k_2 x_2 = m_1 g$$

$$-k_2 x_1 + (k_2 + k_3)x_2 - k_3 x_3 = m_2 g$$

$$-k_3 x_2 + k_3 x_3 = m_3 g$$

$$\begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} m_1 g \\ m_2 g \\ m_3 g \end{bmatrix}$$

$$Ax = b$$

One way of solving an equation of the form *Ax=b* is to premultiply both sides by A^{-1} :

$$Ax = b$$

$$A^{-1}Ax = A^{-1}b$$

$$Ix = A^{-1}b$$

$$x = A^{-1}b$$

Ways of obtaining the inverse of a square matrix in Matlab:

```
>> A = [1 2; -3 8]
A =
     1     2
    -3     8
>> inv(A)
ans =
    0.5714   -0.1429
    0.2143    0.0714
>> A^-1
ans =
    0.5714   -0.1429
    0.2143    0.0714
```

Matlab Solution (using inverse matrix):

```
>> k1 = 50; k2 = 100; k3 = 50;     % in N/m
>> m1 = 60; m2 = 70; m3 = 80;      % in kg
>> g = 9.81;

>> A = [ k1+k2  -k2     0
         -k2    k2+k3  -k3
          0     -k3     k3 ];
>> b = [ m1*g; m2*g; m3*g ];
>> x = inv(A) * b
x =
   41.2020
   55.9170
   71.6130
```

Very Important Note: This is not a very good way of solving systems of equations. "Left division" is much better.

Left Division:

Normal “right division” can be thought of as post-multiplying by the inverse:

x/y is equivalent to $x * y^{-1}$

“Left division “is logically equivalent to pre-multiplying by the inverse:

$y\backslash x$ is equivalent to $y^{-1} * x$

In both cases the quantity under the slash is the divisor.

For scalars the two operations are equivalent:

```
>> 6/3
ans =
     2
>> 3\6
ans =
     2
```

For matrices this is not the case.

Matlab Solution (using left division):

```
>> k1 = 50; k2 = 100; k3 = 50;      % in N/m
>> m1 = 60; m2 = 70; m3 = 80;       % in kg
>> g = 9.81;

>> A = [ k1+k2  -k2     0
         -k2    k2+k3  -k3
          0     -k3     k3 ];
>> b = [ m1*g; m2*g; m3*g ];
>> x = A\b
x =
   41.2020
   55.9170
   71.6130
```

Left division is preferable because Matlab does not actually compute the inverse matrix and perform the multiplication. Instead it uses superior techniques to directly obtain the final answer.

As stated in the text, the problem actually involves finding the final position of the three jumpers. Each bungee cord is initially 20m long. This means that the jumpers are initially 20, 40, and 60 metres from “height zero”.

```
>> initialX = [ 20; 40; 60 ];
>> finalX = initialX + x
finalX =
   61.2020
   95.9170
  131.6130
```

The matrix transpose operator (‘) might be employed:

```
>> initialX = [ 20 40 60 ];
>> finalX = initialX + x'
finalX =
   61.2020   95.9170  131.6130
```

Notes on Bungee Jumper Example:

Simultaneous equations are overkill for this problem. The solution can be found much more easily.

In the steady state the first cord supports all three jumpers. Therefore:

$$x_1 k_1 = g(m_1 + m_2 + m_3)$$
$$x_1 = g(m_1 + m_2 + m_3)/k_1 = 41.202$$

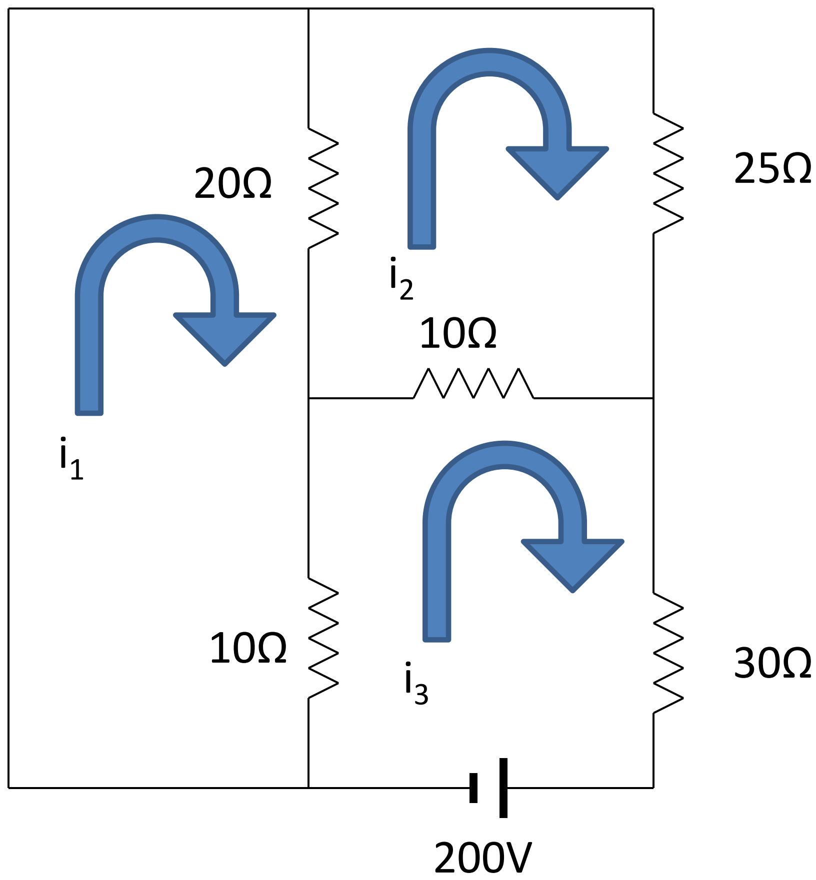
Similarly the second cord supports the second and third jumpers:

$$(x_2 - x_1)k_2 = g(m_2 + m_3)$$
$$x_2 = g(m_2 + m_3)/k_2 + x_1 = 55.917$$

And the third cord supports only the third jumper:

$$(x_3 - x_2)k_3 = gm_3$$
$$x_3 = gm_3 / k_3 + x_2 = 71.613$$

Problem: What are i1, i2, and i3 in the circuit shown below?



The voltage drops around each of the loops must add up to zero. For the left loop:

$$10(i_1 - i_3) + 20(i_1 - i_2) = 0$$

Simplifying and applying the same principle to the other two loops gives:

$$30i_1 - 20i_2 - 10i_3 = 0$$

$$-20i_1 + 55i_2 - 10i_3 = 0$$

$$-10i_1 - 10i_2 + 50i_3 + 200 = 0$$

$$30i_1 - 20i_2 - 10i_3 = 0$$

$$-20i_1 + 55i_2 - 10i_3 = 0$$

$$-10i_1 - 10i_2 + 50i_3 + 200 = 0$$

$$30i_1 - 20i_2 - 10i_3 = 0$$

$$-20i_1 + 55i_2 - 10i_3 = 0$$

$$-10i_1 - 10i_2 + 50i_3 = -200$$

In matrix form:

$$\begin{bmatrix} 30 & -20 & -10 \\ -20 & 55 & -10 \\ -10 & -10 & 50 \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ i_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -200 \end{bmatrix}$$

$$Ax = b$$

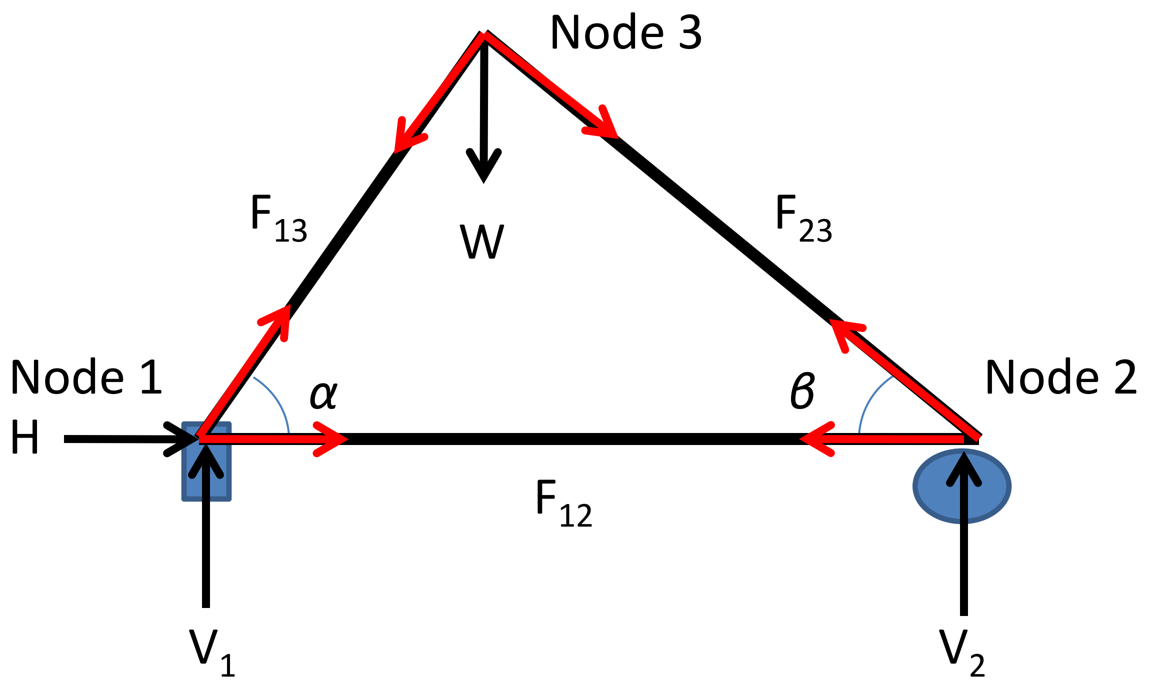
Matlab Solutions:

```
A = [ 30 -20 -10
     -20 +55 -10
     -10 -10 50 ];
b = [0; 0; -200];

%using matrix inverse
x = inv(A) * b
x =
   -3.0000
   -2.0000
   -5.0000

% using left division (much better)
x = A\b
x =
   -3.0000
   -2.0000
   -5.0000
```

Problem: What are the forces (F_{13}, F_{12}, and F_{23}) in the members of the truss shown below and what are the reactions (H, V_1, V_2) at the supports?



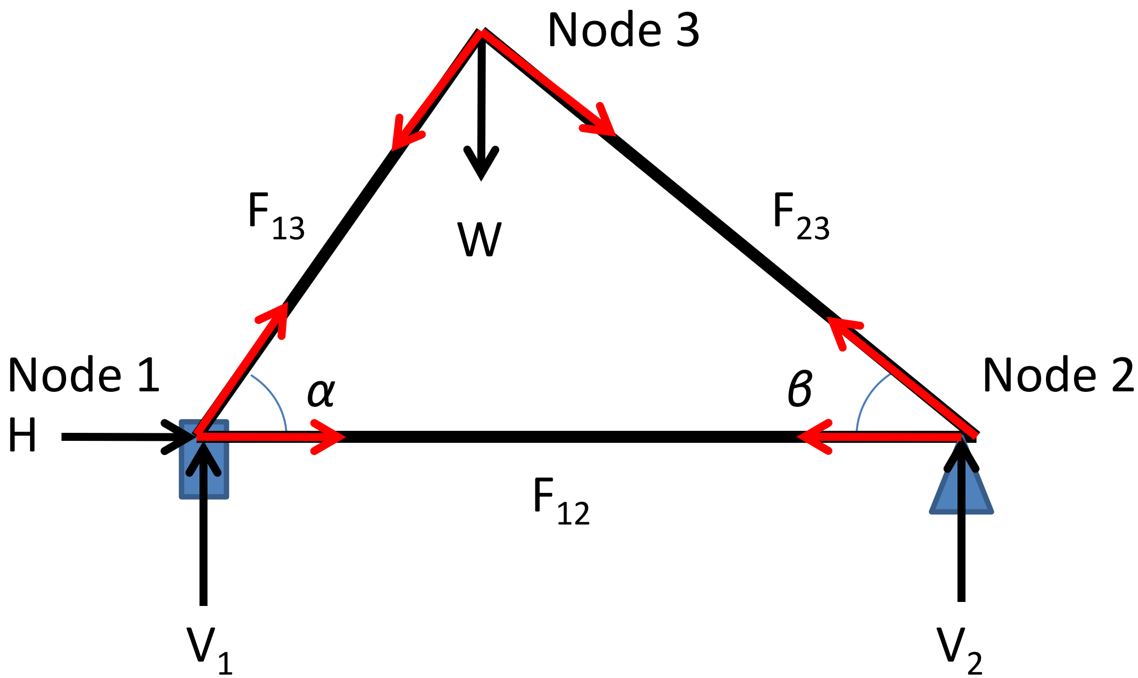
The horizontal and vertical forces acting at each of the nodes must add up to zero.

This provides six equations (three nodes times two equations per node).

Convention for truss members: positive force = tension.

Positive forces act in the directions shown by the arrows.

Sign convention for adding up forces: upwards = positive, to the right = positive.



For node 1: $$\text{horizontal forces} = H + F_{12} + \cos(\alpha)F_{13} = 0$$

$$\text{vertical forces} = V_1 + \sin(\alpha)F_{13} = 0$$

Note: F_{13} obviously must be negative. This indicates that the arrows are backward and that the truss member is actually in compression.

Applying the same idea to the other two nodes gives the complete set of equations:

$$H + F_{12} + F_{13}\cos(\alpha) = 0$$
$$V_1 + F_{13}\sin(\alpha) = 0$$

$$-F_{12} - F_{23}\cos(\beta) = 0$$
$$V_2 + F_{23}\sin(\beta) = 0$$

$$-F_{13}\cos(\alpha) + F_{23}\cos(\beta) = 0$$
$$-F_{13}\sin(\alpha) - F_{23}\sin(\beta) - W = 0$$

$$\begin{bmatrix} 0 & 1 & 0 & 1 & \cos(\alpha) & 0 \\ 1 & 0 & 0 & 0 & \sin(\alpha) & 0 \\ 0 & 0 & 0 & -1 & 0 & -\cos(\beta) \\ 0 & 0 & 1 & 0 & 0 & \sin(\beta) \\ 0 & 0 & 0 & 0 & -\cos(\alpha) & \cos(\beta) \\ 0 & 0 & 0 & 0 & -\sin(\alpha) & -\sin(\beta) \end{bmatrix} \begin{bmatrix} V_1 \\ H \\ V_2 \\ F_{12} \\ F_{13} \\ F_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ W \end{bmatrix}$$

Matlab Solution:

```
a = pi/6;  % alpha = 30 degrees (Matlab works in radians)
 b = pi/3; % beta = 60 degrees
W = 100; % W = 100  N

A = [  0  1  0  1  cos(a) 0
       1  0  0  0  sin(a) 0
       0  0  0 -1  0 -cos(b)
       0  0  1  0  0 sin(b)
       0  0  0  0  -cos(a) cos(b)
       0  0  0  0  -sin(a) -sin(b) ];
r = [ 0; 0; 0; 0; 0; W];
x = A\r;

x =
```

25.0000	V_1
0	H
75.0000	V_2
43.3013	F_{12}
-50.0000	F_{13} (member is in compression)
-86.6025	F_{23} (member is in compression)

Notes on Truss Example:

This example is somewhat artificial for two reasons.

1/. The two equations for Node 3 can be solved without considering any of the other equations. This gives F_{13} and F_{13}. Once these forces are known everything else is easily calculated. There is no real need to solve the six equations simultaneously.

2/. Solving for reaction H is rather pointless in that it must obviously be zero (as anything else would cause the whole truss to accelerate to the right). Basically H is included just to make the number of unknowns match the number of equations. Both H and the equation in which it appears can be eliminated (leaving five equations in five unknowns). The equation can legitimately be discarded because, once H is removed, the equation is a linear combination of two of the other equations.

Simultaneous equations and the Casio calculator.

Press MODE until options include EQN, then select EQN.

Select number of unknowns.

Enter the equations. The prompts assume:

$$
\begin{aligned}
a_1x + b_1y + c_1z &= d_1 \\
a_2x + b_2y + c_2z &= d_2 \\
a_3x + b_3y + c_3z &= d_3
\end{aligned}
$$

Once the last value has been entered the calculator displays the value of *x.*

Use the up arrow and down arrows to cycle through the values for *x*, *y*, and *z*.

To repeat the whole process hit “=“ until the “a1?” prompt appears.

To exit the mode hit MODE and “1”.