

MATLAB I

1.- Variables reales

- **Ejercicio 1** Introduce las siguientes instrucciones en la ventana de comandos de Matlab:

```
>> x=3; [entrar]
>> x=3*x-1 [entrar]
```

Interpreta el resultado. Observa la diferencia entre acabar una instrucción con punto y coma o no hacerlo.

- **Ejercicio 2** Introduce

```
>> clc [entrar]
```

¿Qué ocurre? Introduce

```
>> x [entrar]
```

Introduce

```
>> whos x [entrar]
```

Introduce

```
>> clear x [entrar]
>> x
```

¿Qué ocurre? Limpia de nuevo la ventana de comandos.

- **Ejercicio 3** Introduce en la ventana de comandos

```
>> help who [entrar]
```

¿Qué comandos se relacionan con `who`?

- **Ejercicio 4** Introduce

```
>> x=3;y=5; [entrar]
>> aux=x; [entrar]
>> x=y; [entrar]
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **Ejercicio 5 [POTENCIAS]** Introduce

```
>> x=3;y=4;           [entrar]
>> x^2
>> x^y
>> sqrt(x)
>> x^1/2
>> x^(1/2)
>> x^.5
>> sqrt(x^2+y^2)
```

• **Ejercicio 6** (clear all, clc) [PRECISIÓN] Escribe en la ventana de comandos de Matlab lo siguiente:

```
>> x=1e-15;y=((1+x)-1)/x           [entrar]

>> a=1e308;b=1.1e308;c=-1.001e308; [entrar]
>> a+(b+c)                         [entrar]
>> (a+b)+c                         [entrar]

>> realmax                          [entrar]
>> realmin                          [entrar]
```

Busca estos comandos con ayuda de [help](#). Busca también NaN, Inf.

• **Ejercicio 7** (clear all, clc) [NÚMEROS COMPLEJOS] Introduce

```
>> z=2+3i           [entrar]
>> real(z)          [entrar]
>> imag(z)          [entrar]
>> conj(z)          [entrar]
>> abs(z)           [entrar]
>> angle(z)         [entrar]
```

• **Ejercicio 8** (clear all, clc) [FORMATOS NUMÉRICOS] Introduce

```
>> pi           [entrar]
>> format long,pi [entrar]
>> format short,pi [entrar]
>> format long e,pi [entrar]
>> format short e,pi [entrar]
>> format rat,pi [entrar]
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

2.- Vectores

• Ejercicio 9 (clear all, clc) Introduce

```
>> v=[1 2 3 4]      [entrar]
>> v=[1,2,3,4]     [entrar]
>> w=[1;2;3;4]     [entrar]
>> v'              [entrar]
>> v.'            [entrar]
>> c=[1+i 1-i]     [entrar]
>> c'             [entrar]
>> c.'           [entrar]
>> clear all      [entrar]
>> clc           [entrar]
```

• Ejercicio 10 (clear all, clc) Introduce

```
>> u=[1:6]          [entrar]
>> u=1:6           [entrar]
>> v=[1:3:6]       [entrar]
>> v=1:2:6         [entrar]
>> w=1:.5:4        [entrar]
>> a=3:-1:1        [entrar]
>> b=linspace(1,8,2) [entrar]
>> b=linspace(1,8,4) [entrar]

>> b(1)            [entrar]
>> b(2)            [entrar]
>> b(2:4)          [entrar]
>> b(2:end)        [entrar]
>> b(2:3)          [entrar]

>> length(b)       [entrar]
>> size(b)         [entrar]
>> max(b)          [entrar]
>> min(b)          [entrar]
>> [m,c]=max(b)    [entrar]
>> sum(b)          [entrar]
>> y=cumsum(b)     [entrar]
>> y(end)          [entrar]
>> prod(b)         [entrar]
>> sort(b)         [entrar]
```

• Ejercicio 11 (clear all, clc) [OPERACIONES CON VECTORES] Introduce

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• Ejercicio 11 CONTINUACIÓN

```
>> dot(a,b) [entrar]
>> cross(a,b) [entrar]
>> norm(a) [entrar]

>> c=[a,b] [entrar]
>> d=[a;b] [entrar]
>> cross(c(1:3),c(4:end)) [entrar]
```

Busca los comandos `dot`, `cross` y `norm` con ayuda [help](#).

3.- Matrices

• Ejercicio 12 (clear `all`, `clc`) Introduce

```
>> fila=4;columna=5;
>> eye(fila)
>> zeros(columna)
>> zeros(fila,columna)
>> ones(fila)
>> ones(fila,columna)

>> rand
>> rand(n)
>> rand(n,m)
```

• Ejercicio 13 (clear `all`, `clc`) Introduce

```
>> A=[11,12,13;21,22,23;31,32,33] [entrar]
>> A' [entrar]
>> diag(A) [entrar]
>> diag([11,22,33]) [entrar]
>> triu(A) [entrar]
>> tril(A) [entrar]
>> triu(A,1) [entrar]
>> triu(A,-1) [entrar]
>> rot90(A) [entrar]
>> rot90(A,2) [entrar]
>> A(3,:)=[] [entrar]
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **Ejercicio 14 CONTINUACIÓN**

```
>> M4=A(:, [1 3 4])
>> M5=A(2:4, :)
>> M6=A(3, :)
>> M7=A(:, 3)
>> M8=A(:)
>> M9=[M3;M4]
>> M10=[M3 M4]
```

• **Ejercicio 15** Construir una matriz A de tamaño 5x5 con números aleatorios comprendidos entre -10 y 10

- 1) Extraer su 4ª columna.
- 2) Extraer su 3ª fila.
- 3) Extraer la submatriz comprendida entre las filas 3 y 5 y las columnas 2 y 4.
- 4) Extraer los tres elementos de la esquina inferior izquierda de A.
- 5) Extraer las columnas en las que está el número máximo y mínimo.
- 6) Extraer las filas en las que está el número máximo y mínimo.
- 7) Extrae los números máximo y mínimo de A, y su ubicación en la matriz.

• **Ejercicio 16 [OPERACIONES CON MATRICES]** (`clear all, clc`) Introduce

```
>> A=[1 2 3;0 1 1;0 -1 4]
>> B=[1 0 1;0 1 4;0 0 9]
>> A*B
>> A.*B
>> rank(A)
>> det(A)
>> inv(A)
>> eye(3)/A
>> A\eye(3)
>> inv(B)*A
>> B\A
>> A./B
>> 1./A
>> 1./(A*B)
>> A^2
>> A.^2
>> eig(A)
>> max(A)
>> max(max(A))
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **Ejercicio 17 CONTINUACIÓN**

Aplica los apartados 1) y 2) a:

- a) $x = \pi/4$
- b) $x = [\pi/3 \ \pi/4 \ \pi/2]$
- c) $x = [\pi/3 \ \pi/4; \pi/2 \ \pi]$

4.- Sumando y multiplicando cosas

• **Ejercicio 18 [BUCLES for-end]** Investiga que realizan los siguientes programas:

Programa 1

```
1 s=0;
2 for n=1:10
3     s=s+n;
4 end
5 s
```

Programa 2

```
1 p=1;
2 for n=1:10
3     p=p*n;
4 end
5 p
```

Programa 3

```
1 x=6;
2 p=1;
2 for n=1:10
3     p=p*x;
4 end
5 p
```

• **Ejercicio 19 [function]** Genera las funciones 'sum_n.m', 'fact_n.m' y 'pot_n.m' que efectúen los cálculos anteriores. El argumento de entrada debe ser un entero n.

• **Ejercicio 20 [function]** Generalizar, si es preciso, las funciones 'sum_n.m', y 'pot_n.m' para que admitan matrices cuadradas de orden arbitrario.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **Ejercicio 22** [function] [Función 'Inline'] Introduce en la ventana de comandos

```
>> f=inline('exp(-x)*cos(x)')
>> f(0)
>> feval(f,0)

>> x=[0:0.1:1];feval(f,x)
>> f=vectorize(f)
>> y=feval(f,x)
>> plot(x,y)

>> g= inline('exp(-x).*cos(x)')
>> plot(x,g(x))

>> g=inline('exp(-x^2-y^2)*cos(x^2+y^2)','x','y')
>> g(0,0)
>> feval(g,0,0)
```

• **Ejercicio 23** [function] [Función anónima] Introduce en la ventana de comandos

```
>> f=@(x) exp(-x)*cos(x)
>> f(0)
>> feval(f,0)
>> x=[0:0.1:1];feval(f,x)
>> f=vectorize(f)
>> feval(f,x)

>> f=@(x) exp(-x).*cos(x)
>> y=f(x);
>> plot(x,y)

>> g=@(x,y) sin(x)*cos(y)
>> g(pi/4,0)
>> feval(g, pi/4,0)
```

• **Ejercicio 24** [function] Diseña la función

```
function [suma,epsilon]=sumatorio0(f,n1,N)

% DESCRIPCIÓN:
% sumatorio0 suma desde n1 hasta N la función f(n)
% introducida por el usuario como "inline('f(n)')"
% También invocando una función exterior cargada con 'f'
% OUTPUT:
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Cartagena99

- **Ejercicio 25** [function] Analiza la función (donde f puede ser 'inline' u otro fichero f.m)

```
function s=sumatorio1(f,n1,N)
format long
f=vectorize(f);
n=n1:N;
x=f(n);
s=sum(x);
```

Incluye en la función anterior un argumento de salida que sea la diferencia en valor absoluto de la diferencia de los dos últimos términos del sumatorio. Añadir a `sumatorio1` un argumento de salida que sea la función cumulativa de f, es decir, aquella función g que cumple

$$g(n) = \sum_{n=n_1}^N f(N), \quad n_1 \leq n \leq N$$

- **Ejercicio 26** [FICHEROS] Introduce en la ventana de comandos:

```
>> A=rand(10,5);
>> fid=fopen('PruebaMatriz.txt','w')
>> fprintf(fid,'%f',A)
>> fclose(fid)

>> fid=fopen('PruebaMatriz.txt','w')
>> fprintf(fid,'%f\n',A)
>> fclose(fid)

>> fid=fopen('PruebaMatriz','r')
>> fid=fopen('PruebaMatriz.txt','r')

>> [A,cont]=fscanf(fid,'%f')
>> [A,cont]=fscanf(fid,'%f',[10,5])

>> rewind(fid)
>> [A,cont]=fscanf(fid,'%f',[10,5])

>> rewind(fid)
>> [A,cont]=fscanf(fid,'%f',[5,inf])

>> rewind(fid)
>> [A,cont]=fscanf(fid,'%f',[4,inf])

>> rewind(fid)
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

- **Ejercicio 27 [Funciones con argumentos de entrada y/o salida no definidos]** Busca información sobre las funciones `varargin`, `nargin`, `varargout` y `nargout`.

Consideremos el sistema de ecuaciones escrito en forma matricial

$$Ax = b$$

donde A es una matriz $n \times n$, x y b vectores columna $n \times 1$, siendo x la incógnita y A y b los datos. Diseña una función (basada en la descomposición LU de MATLAB que buscaras en `help`) de modo que:

- Si el argumento de entrada es una matriz cuadrada A y se pidan dos argumentos de salida, resulte $[L, U]$.
- Si el argumento de entrada es una matriz cuadrada A y se pidan tres argumentos de salida, resulte $[L, U, P]$.
- Si los argumentos de entrada son A y el vector b , los argumentos de salida sean $[L, U, x]$

Solución

```
function varargout=DescLU(a,varargin)
[L,U,P]=lu(a);
% Descomposición LU
% CASO 1.- INPUT=A --> OUTPUT=[L,U]
% CASO 2.- INPUT=A --> OUTPUT=[L,U,P]
% CASO 3.- INPUT=[A,b]--> OUTPUT=[L,U,x] (Ax=b)
if (nargin==1) & (nargout==2)
    varargout{1}=L;
    varargout{2}=U;
elseif (nargin==1) & (nargout==3)
    varargout{1}=L;
    varargout{2}=U;
    varargout{3}=P;
elseif (nargin==2) & (nargout==3)
    b=varargin{1};
    varargout{1}=L;
    varargout{2}=U;
    varargout{3}=U \ (L\b);
end
return
```

- **Ejercicio 28 [Funciones con argumentos de entrada y/o salida no definidos]** Busca información sobre las funciones `varargin`, `nargin`, `varargout` y `nargout`.

Diseña una función `argfun0` de modo que

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **Ejercicio 29** [Bucles `for-end` y bucles `while-end`] Idea básica:

- Si conocemos el número de iteraciones: Empleamos un bucle `for-end`.
- Si lo que buscamos es que se repita el bucle hasta que se satisfaga determinada condición (o condiciones): Empleamos un bucle `while-end`. En este caso, es conveniente introducir una condición límite de iteraciones mediante un `if-end` para evitar infinitas iteraciones.

Aplicación: [Tolerancias] Se sabe que la ecuación recurrente

$$\begin{cases} x_{n+1} = \frac{1}{2} \left(x_n + \frac{a}{x_n} \right), & n = 0, 1, 2, \dots \\ x_0 = a_0 \end{cases}$$

converge al valor \sqrt{a} si $a > 0$, $a_0 > 0$.

- 1) Escribe una función 'raiter00' cuyos argumentos de entrada sean el valor de $a > 0$ y el número N de iteraciones y dé como resultado el valor x_{N+1} . Emplear, por tanto, un bucle `for-end`.
- 2) Escribe una función 'raiter01' cuyos argumentos de entrada sean a y el número N de iteraciones y dé como resultado un vector $x = [x(1), \dots, x(N+1)]$ que contenga los valores x_0, x_1, \dots, x_N , el error definido por $\text{error} = \text{abs}(x(N+1) - x(N))$ y una gráfica de la sucesión $\{x_n\}$ frente a n . Emplear un bucle `for-end`.
- 3) ¿Cómo se podría añadir el cálculo del error en 'raiter00'?
- 4) Escribe una función 'raiter02' cuyos argumentos de entrada sean a , una tolerancia tol (de modo que si en algún paso n del proceso se cumple que $\text{abs}(x(N+1) - x(N)) \leq \text{tol}$, el programa finalice) y un número máximo N_{max} de iteraciones permitidas, y dé como resultado el valor x_{N+1} y el número de iteraciones n_{iter} efectuadas, en caso de no llegar a N_{max} .

• **Ejercicio 30** [examen] Introduce en la ventana de comandos `help norm`. Averigua qué significa la norma2 de una matriz.

Diseña las funciones:

- 1) `function [eA,error]=expA(A,N)`
- 2) `function [ln1A,error]=logA(A,N)`

que den el valor de las sumas respectivas (truncadas en N términos)

$$\exp(A) = \mathbb{I} + \sum_{n=1}^{\infty} \frac{A^n}{n!}, \quad \log(\mathbb{I} + A) = \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} A^n$$

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **SEGUNDA PARTE: ECUACIONES NO LINEALES Y OBTENCIÓN DE RAÍCES**

- **Ejercicio 31 [Método de la bisección (o bipartición)]** (Explicación en pizarra) Escribe la función y el algoritmo correspondiente:

```
function [c,niter]=biseccion(f,a,b,tol)
% BÚSQUEDA DE CEROS POR EL MÉTODO DE LA BISECCIÓN
% DATOS-----
% f=@(x) f(x) // también llamar a 'f' // también 'inline'
% [a,b], extremos del intervalo de búsqueda
% tol=TOLERANCIA, anchura del intervalo final máximo permitido
% SALIDAS-----
% c=solución aproximada
% niter=número de iteraciones realizadas
```

Aplicación: Aproximar la raíz de la ecuación $\sqrt{x} \sin x - x^3 + 2 = 0$ en el intervalo [1,2] con un error menor que 10^{-4} . Comprobar mediante un gráfico.

- **Ejercicio 32 [Método de Newton-Raphson 1D]** (Explicación en pizarra) Diseña la función y el algoritmo correspondiente:

```
function [c,niter]=newtonraphson1D(f,df,x0,tol,Nmax)
% BÚSQUEDA DE CEROS POR EL MÉTODO DE NEWTON-RAPHSON (1D)
% DATOS-----
% f=@(x) f(x) (u otra forma)
% df=@(x) 'df/dx' (u otra forma)
% x0, valor de arranque del método
% tol=TOLERANCIA, anchura del intervalo final máximo permitido abs(x(n+1)-x(n))
% Nmax=número máximo de iteraciones
% SALIDAS-----
% c=cero aproximado por el método
% niter=número de iteraciones realizadas
```

Aplicación: Obtener $\sqrt[5]{35}$ de modo que $|x_n - x_{n-1}| < 10^{-6}$.

- **Ejercicio 33 [Método de la secante 1D]** (Explicación en pizarra) Diseña la función:

```
function [c,niter]=secante1(f,x0,tol,Nmax)
% MÉTODO DE LA SECANTE (1D)
% LLAMADAS-----
% función f
% función 'newtonraphson1D'
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **Ejercicio 34 [Método de 'regula falsi']** (Explicación en pizarra) Diseña la función:

```
function [c,niter]=regulafalsi(f,a,b,tol,Nmax)
% MÉTODO DE LA REGULA FALSI (1D)
% LLAMADAS-----
% función f
% DATOS-----
% a y b, valores del intervalo Bolzano
% tol=TOLERANCIA, anchura del intervalo final máximo permitido abs(x(n+1)-x(n))
% Nmax=número máximo de iteraciones
% SALIDAS-----
% c=cero aproximado por el método
% niter=número de iteraciones realizadas
```

Aplicación: Aplicar el método anterior a la ecuación $e^{-x} - x = 0$ en el intervalo $[0,1]$ de modo que $|x_n - x_{n-1}| < 10^{-6}$. Comprobar con un gráfico.

• **Ejercicio 35 [Vectores y Matrices de funciones. Archivos .m]** Consideremos las funciones

$$f(x, y) = x^2 - 2x - y + 0.5, \quad g(x, y) = x^2 + 4y^2 - 4$$

- 1) Escribir una función F en Matlab cuyo argumento de entrada sea un vector columna $X = (X(1) = x, X(2) = y)$ y la salida sea el vector $Z = (Z(1) = f(x, y), Z(2) = g(x, y))$. Aplicarlo al enunciado para $(2, 0.25)$.
- 2) Escribir una función JF en Matlab cuyo argumento de entrada sea un vector $X = (X(1) = x, X(2) = y)$ y la salida sea la matriz jacobiana de (f, g) evaluada en X . Aplicarlo al enunciado para $(2, 0.25)$.

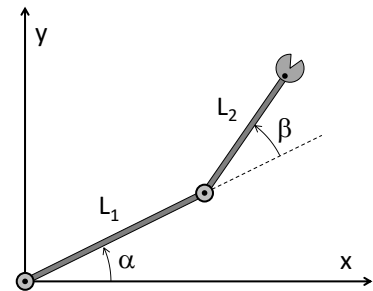
• **Ejercicio 36 [Método de Newton-Raphson nD]** Diseña la función

```
function [c,niter]=newtonraphsonsystem(F,JF,X0,tol)
% BÚSQUEDA DE CEROS POR EL MÉTODO DE NEWTON-RAPHSON (nD)
% DATOS-----
% F=Función vectorial del sistema no lineal(Fichero F.m)
% JF=Jacobiano de F(Fichero JF.m)
% x0, vector valor de arranque del método
% tol=TOLERANCIA, anchura del intervalo final máximo permitido abs(x(n+1)-x(n))
% SALIDAS-----
% c=cero vector aproximado por el método
% niter=número de iteraciones realizadas
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

• **Ejercicio 38** Se considera un brazo articulado de robot formado por un brazo 1 de longitud 20 cm que pivota en el origen de coordenadas (0,0) y otro brazo 2 de longitud 10 cm que pivota sobre el extremo del primero. El movimiento del brazo transcurre en el plano xy . Se pide encontrar los ángulos α (del brazo 1 respecto al eje x) y β (del brazo 2 respecto a la dirección del brazo 1) para que el extremo libre del brazo alcance las coordenadas (15,10) cm. Dichos ángulo son inicialmente $\alpha = \beta = \pi/6$ rad. Emplear para ello un método **Newton-Raphson**.



PISTA:

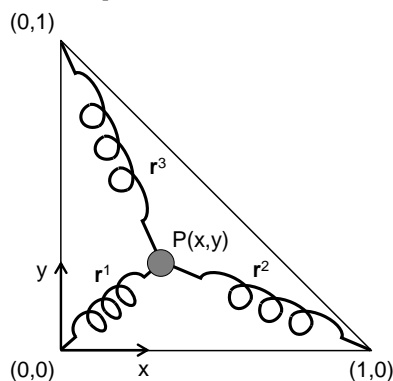
Las coordenadas (x, y) (en cm) del extremo del brazo vienen dadas por las ecuaciones:

$$x = 20 \cos(\alpha) + 10 \cos(\alpha + \beta), \quad y = 20 \sin(\alpha) + 10 \sin(\alpha + \beta)$$

• **Ejercicio 39** Three springs are arranged having one of their ends joined into a point (P) that is free to move and the other ends respectively fixed to the three corners of a rectangular triangle, see Figure. Each spring i has a relaxed length equal to 0. The relation between its force vector \mathbf{F}^i and its actual length $\|\mathbf{r}^i\|$ is quadratic and can be expressed as

$$\mathbf{F}^i = \begin{Bmatrix} F_x \\ F_y \end{Bmatrix} = -k \|\mathbf{r}^i\| \mathbf{r}^i = -k \sqrt{(x - x_0^i)^2 + (y - y_0^i)^2} \begin{Bmatrix} x - x_0^i \\ y - y_0^i \end{Bmatrix}$$

being k an stiffness [N/m^2] and $\mathbf{r}^i = (x - x_0^i, y - y_0^i)$ the length vector where (x_0^i, y_0^i) is the position of the fix end of each spring and (x, y) the position of the point P.



- Write a function that gives the value of the resulting force (vector) in the point P, due to the three springs.
INPUT: Coordinates of point P (vector). **OUTPUT:** Force vector
- Obtain the equilibrium position of P: (x, y) at which total force is equal to zero. The Newton-Raphson method (for system of equations) should be used.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70