

APLICACIÓN DE SOFTWARE
MATEMÁTICO PARA EL CÁLCULO EN
INGENIERÍA

Docente: María Dolores Jiménez

Organizado por: Asociación Jujeña de
Estudiantes de Ingeniería Química (AJEIQ)

Facultad de Ingeniería de la UNJu

2023

Mathcad®

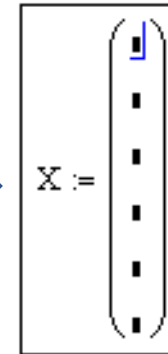
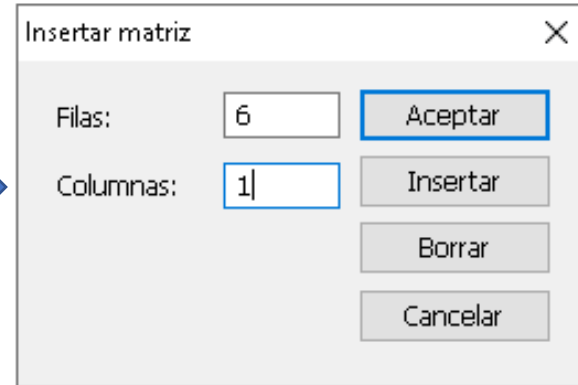


UNIDAD III



RANGO DE DATOS

Vector



$$X := \begin{pmatrix} -4 \\ -2 \\ 0 \\ 2 \\ 4 \\ 6 \end{pmatrix}_m$$

$$X_0 = -4$$
$$X_5 = 6$$

Ejemplo:
Suma de fracciones
másicas de un ensayo
de tamizado

$$fr := \begin{pmatrix} 0.15 \\ 0.45 \\ 0.30 \\ 0.10 \end{pmatrix}$$

$$f_T := fr_0 + fr_1 + fr_2 + fr_3 = 1$$

RANGO DE DATOS

Variable de rango

Matriz x

	\times_n	X^{-1}	$ X $
$\vec{f}(m)$	M^{\odot}	M^T	m..n
$\hat{a} \cdot \hat{v}$	$\hat{a} \times \hat{v}$	ΣU	

$X := -4m, -2m..6m$

X =

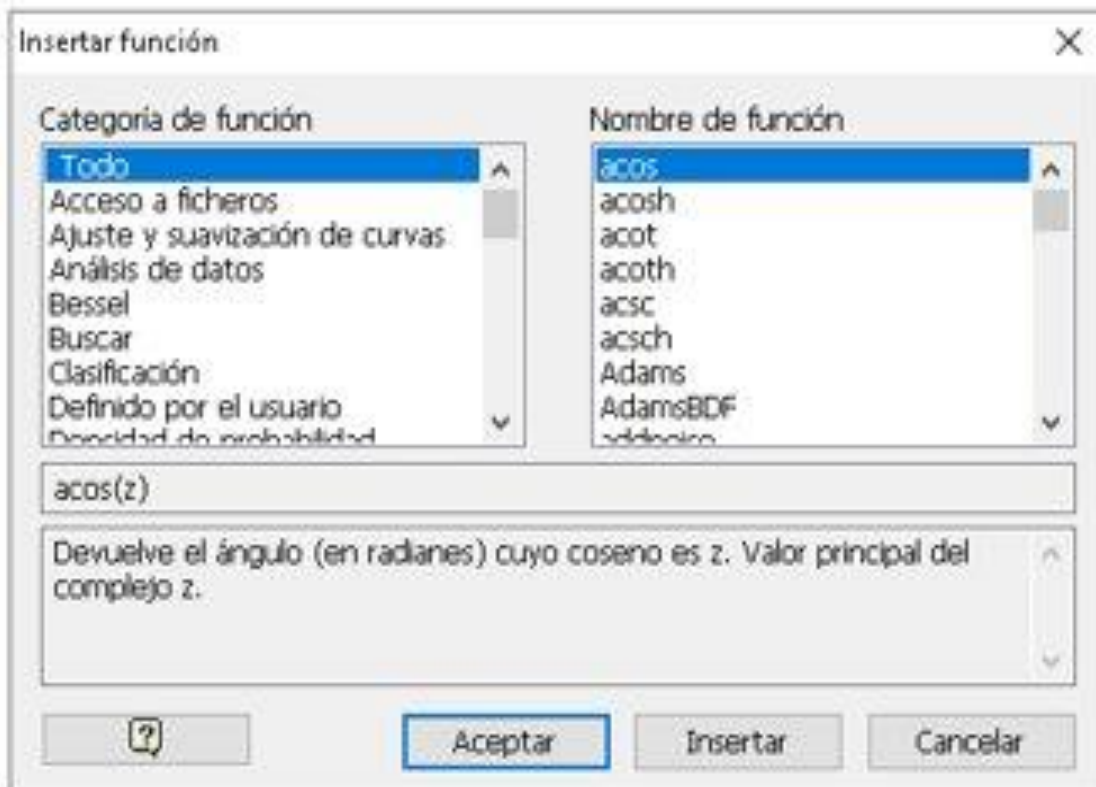
-4	m
-2	
0	
2	
4	
6	

El segundo número define el **valor incremental**.

Mathcad toma la diferencia entre el primer y el segundo número y lo usa como valor incremental.

FUNCIONES

Funciones predefinidas



Definir nuevas funciones

$$f(x) := \sqrt[4]{x-1}$$

$$f(2) = 1$$

$$f(\sqrt{25}) = 1.414$$

GRÁFICOS X-Y →

Sistema de coordenadas cartesianas para representar la **distribución bidimensional**. En el gráfico se obtiene un conjunto de puntos conocido como **diagrama de dispersión**.

La importancia de las distribuciones bidimensionales radica en investigar cómo influye una variable sobre la otra.

Gráfico usando vectores

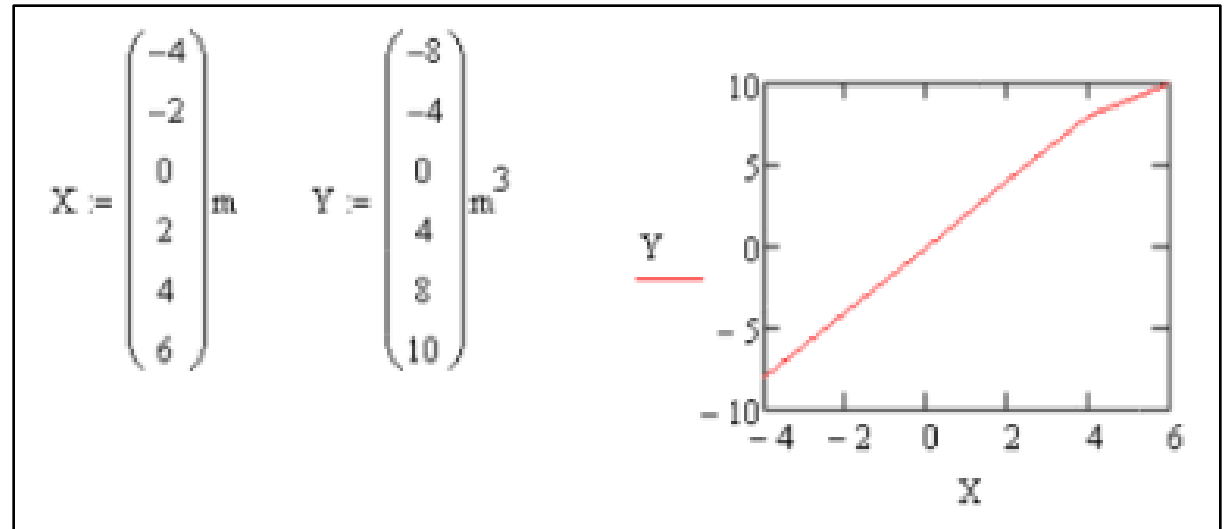
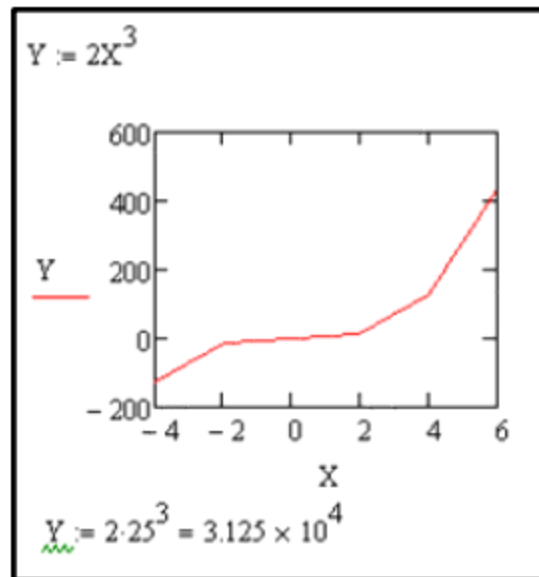
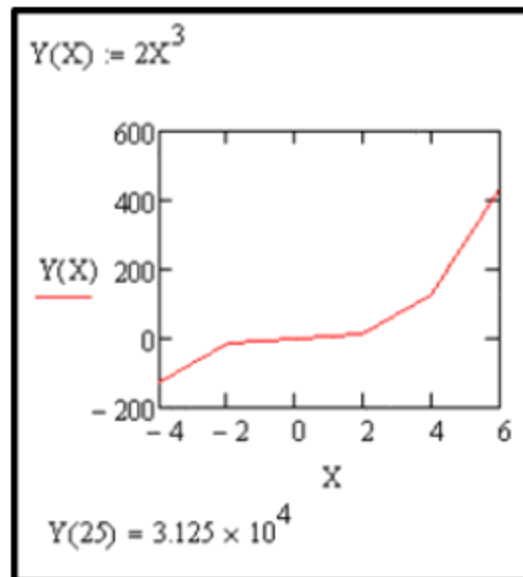


Gráfico conociendo la ecuación



Definiendo la variable dependiente como ecuación



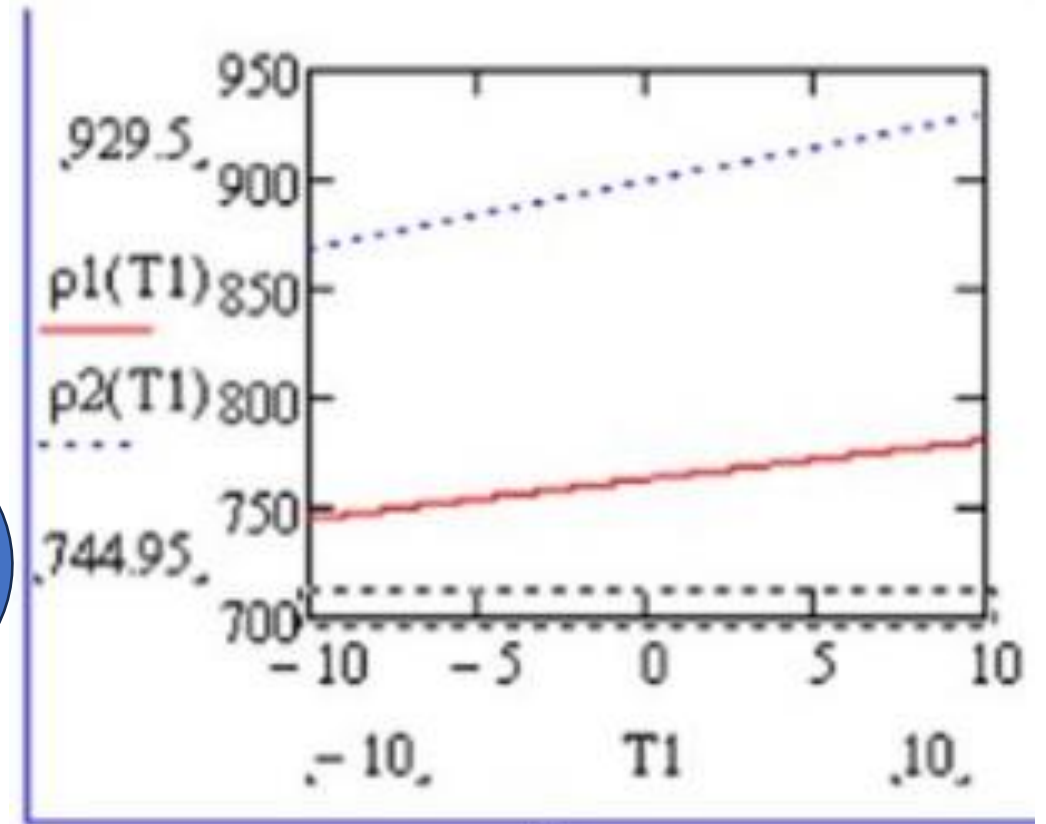
Definiendo la variable dependiente como función de la variable independiente

MÁS DE UNA VARIABLE EN EL EJE

Para habilitar un marcador de posición en el eje: Posicionarse en el margen derecho o izquierdo del eje habilitado y presionar,

$$\rho_1(T_1) := -3.5 \cdot 10^{-3} \cdot T_1^2 + 1.8 \cdot T_1 + 763.3$$

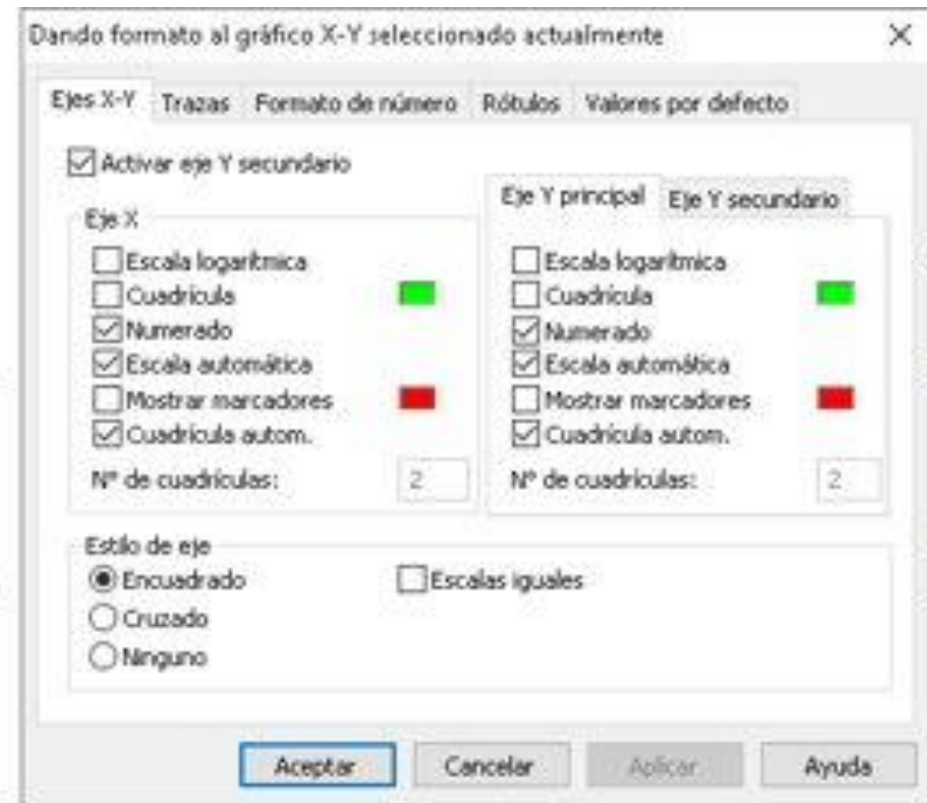
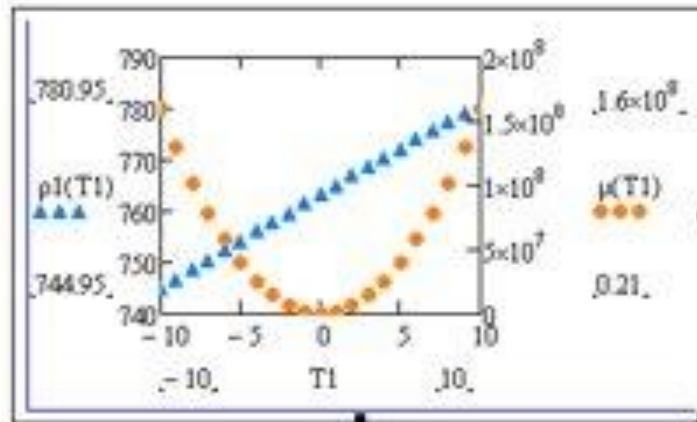
$$\rho_2(T_1) := -5 \cdot 10^{-3} \cdot T_1^2 + 3 \cdot T_1 + 900$$



EJE SECUNDARIO EN EL GRÁFICO

$$\rho I(T1) := -3.5 \cdot 10^{-3} \cdot T1^2 + 1.8 \cdot T1 + 763.3$$

$$\mu(T1) := 1.6 \cdot 10^6 \cdot T1^2 - 1.1 \cdot 10^{-3} \cdot T1 + 0.21$$



Determinación de PENDIENTE y ORDENADA AL ORIGEN

Dependencia funcional entre las variables z e y que mejor ajusta a la distribución bidimensional --> **REGRESIÓN LINEAL**
 $y(z) = a.z + b$

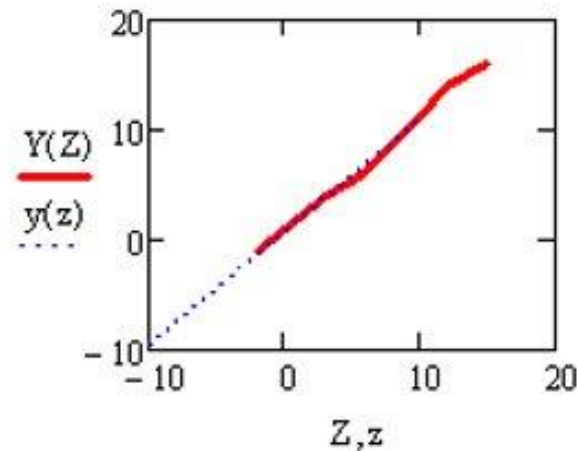
La regresión nos permite determinar el grado de dependencia de las series de valores z e y , y predecir el valor estimado que se obtendría para un valor que no esté en la distribución.

$$Z := \begin{pmatrix} -2 \\ 3 \\ 6 \\ 10 \\ 12 \\ 15 \end{pmatrix} \quad Y(Z) := \begin{pmatrix} -1 \\ 4 \\ 6 \\ 11 \\ 14 \\ 16 \end{pmatrix}$$

$$a := \text{slope}(Z, Y(Z)) = 1.031$$

$$b := \text{intercept}(Z, Y(Z)) = 0.775$$

$y(z) := a.z + b$ Ecuación de la recta que mejor ajusta a los datos.



Las palabras slope e intercept se agregan mediante tipeo

DETERMINACIÓN DE RAÍCES

- Función **root** (método iterativo de Newton)

Valores de la variable independiente que hacen que la variable dependiente sea nula

$$q(n) := (5 - n)e^n - 10$$

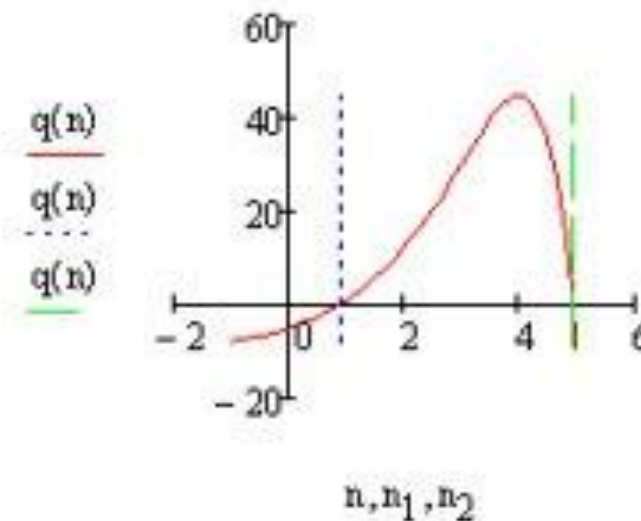
$$n := -1, -0.9..5$$

$$n_{\text{aprox}} := 1$$

$$n_1 := \text{root}(q(n_{\text{aprox}}), n_{\text{aprox}}) = 0.889$$

$$n_{\text{aprox}} := 5$$

$$n_2 := \text{root}(q(n_{\text{aprox}}), n_{\text{aprox}}) = 4.928$$



La palabra **root** se debe tippear

La función **root** encuentra la raíz más próxima al valor asignado para el cálculo.

Si el valor inicial está muy alejado del real, la función **root** no siempre permite obtener soluciones válidas

DETERMINACIÓN DE RAÍCES

- Función **root + TOL**

Si necesitamos una solución más precisa



Se puede usar la función TOL, que por defecto tiene un valor de 0,001.

$$f(x) := (5 - x) \cdot e^x - 40$$

$$x := -1, -0.95.. 5$$

$$x_{\text{apox1}} := 5$$

$$\text{TOL} = 1 \times 10^{-3}$$

$$x1 := \text{root}(f(x_{\text{apox1}}), x_{\text{apox1}}) = 4.597$$

$$f(x1) = -1.741 \times 10^{-12}$$

$$x_{\text{apox2}} := 5$$

$$\text{TOL} := 10^{-15}$$

$$x2 := \text{root}(f(x_{\text{apox2}}), x_{\text{apox2}}) = 4.597$$

$$f(x2) = -7.105 \times 10^{-15}$$

La función TOL determina el límite de convergencia entre interacciones sucesivas y, por lo tanto, el cese del cálculo numérico.

DETERMINACIÓN DE RAÍCES

- Función **polyroots** (cuando se quiere obtener todas las raíces de un polinomio)

Valores de la variable independiente que hacen que la variable dependiente sea nula

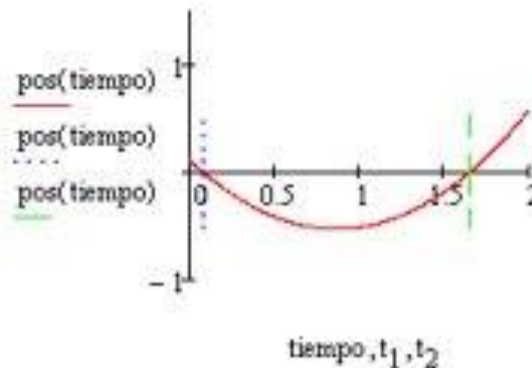
$$\text{pos}(\text{tiempo}) := 0.857 \cdot \text{tiempo}^2 - 1.486 \text{ tiempo} + 0.118$$

$$v := \text{pos}(\text{tiempo}) \text{ coefs} \rightarrow \begin{pmatrix} 0.118 \\ -1.486 \\ 0.857 \end{pmatrix} \quad v := \begin{pmatrix} 0.118 \\ -1.486 \\ 0.857 \end{pmatrix}$$

$$r := \text{polyroots}(v) = \begin{pmatrix} 0.083 \\ 1.651 \end{pmatrix} \quad r^T \rightarrow (0.08342123171539946 \quad 1.6505344275611469)$$

La palabra **polyroots** se debe tipear

```
t_aprox := 0.083
t1 := root(pos(t_aprox), t_aprox)
t1 = 0.083
t_aprox := 1.651
t2 := root(pos(t_aprox), t_aprox)
t2 = 1.651
```



Simbólico	
→	■ →
Modificantes	flotante
rectangular	asumir
resolver	simplificar
reemplazar	factor
expandir	coefs
recopilar	serie
fracpar	fourier
laplace	ztrans
invfourier	invlaplace
invztrans	$\pi^T \rightarrow$
$\pi^{-1} \rightarrow$	$ \pi \rightarrow$
explicitar	combinar
fraccon	rescribir

La función **polyroots** encuentra todas las raíces del polinomio

SISTEMA DE ECUACIONES (GL = 0)



$$x := 1$$

$$y := 1$$

$$z := 1$$

Given

$$6 \cdot x - 2y + z = 10$$

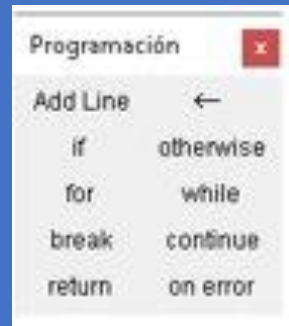
$$10 \cdot x + 3y - 9z = 4$$

$$7x - y + 5z = -3$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} := \text{Find}(x, y, z) = \begin{pmatrix} 0.221 \\ -5.325 \\ -1.974 \end{pmatrix}$$

INSTRUCCIÓN CONDICIONAL

PROGRAMACIÓN



Para ejecutar una expresión únicamente si ocurre alguna condición específica

Para insertar la instrucción de condicionalidad:

- Seleccionar la paleta de programación que contienen los operadores
- Seleccionar el botón "if" presionándolo (no escribir la sentencia)
- Abrir la paleta de operadores booleanos para seleccionar el símbolo de mayor
- Seleccionar "Add line" en caso de necesitar más instrucciones
- Seleccionar el botón "otherwise" (predefinida en Mathcad, no escribirla como tal)
- En el último cuadro colocar el valor que regresará el programa

```
Chequeando(Temp) :=  $\left\{ \begin{array}{l} RV \leftarrow \text{"Busca_hielo"} \text{ if } Temp < 273.15K \\ RV \leftarrow \text{"No_hay_problema"} \text{ otherwise} \end{array} \right.$ 
```

```
Chequeando(280K) = "No_hay_problema"
```

```
Chequeando(250K) = "Busca_hielo"
```

Se requiere uno o más "if" antes que una instrucción "otherwise"
La instrucción "otherwise" se ejecuta solo cuando todas las condiciones son falsas

```
Viscosidad(Temperatura) :=  $\left\{ \begin{array}{l} V \leftarrow 1.02 \cdot 10^2 \cdot \text{Temperatura}^2 - 17 \cdot 10^{-1} \cdot \text{Temperatura} + 0.87 \text{ if } \text{Temperatura} \leq 100 \\ V \leftarrow 1.53 \cdot 10^2 \cdot \text{Temperatura}^2 - 12 \cdot 10^{-1} \cdot \text{Temperatura} + 1.03 \text{ otherwise} \end{array} \right.$ 
```

```
Viscosidad(50) =  $2.549 \times 10^5$ 
```

```
Viscosidad(150) =  $3.442 \times 10^6$ 
```

PROGRAMACIÓN

Si un ciclo está corriendo indefinidamente, presionar la tecla [ESC] para terminar el programa.

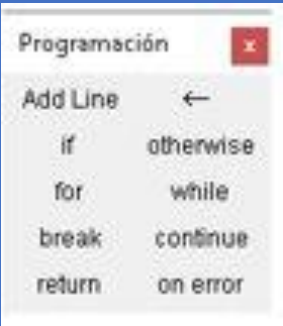
CICLOS DE PROGRAMACIÓN

Un ciclo es una instrucción del programa que causa que una o más instrucciones (el cuerpo del ciclo) se ejecuten repetidamente hasta que una condición particular ocurre.

Hay dos clases de ciclos:

- Ciclos "for" son útiles cuando sabes exactamente cuántas veces necesitas ejecutarlo
- Ciclos "while" son útiles cuando quieres parar una ejecución hasta que ocurra una

PROGRAMACIÓN



Se utiliza un ciclo "for" cuando se conoce exactamente cuántas veces se necesita ejecutar el cuerpo del ciclo.

Para insertar un ciclo "for":

- Seleccionar el lugar donde desea colocar el ciclo for
- Seleccionar en la paleta de programación que contiene los operadores de programación
- Seleccionar el botón "for" (predefinida en Mathcad, no escribirla como tal)
- En el espacio a la izquierda del "existe" colocar la variable de iteración
- En el espacio de la derecha colocar el rango de valores que deben tomar la variable de iteración (la mayoría de las veces se usa un rango variable, pero también puede usarse un vector, una lista de escalares y un vector separado por comas)

```

sum(n) := | s ← 0
          | for x ∈ 1..n
          |   s ← s + 1
          | s

sum(3) = 3
sum(1) = 1
sum(100) = 100
sum(10) = 10
  
```

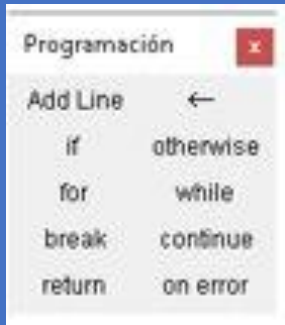
```

sum(n) := | s ← n
          | for x ∈ 1..n
          |   s ← 2·s + 10
          | s

sum(3) = 94      2·3 + 10 = 16
                2·16 + 10 = 42
                2·42 + 10 = 94
  
```


CICLOS "WHILE"

PROGRAMACIÓN



Se requiere que el cuerpo del ciclo se ejecute hasta que se cumpla la condición, pero si no conoce exactamente cuántas veces se necesita, usar el ciclo "While".

Se debe asegurar que se tiene una condición que hace falsa la condición. De otra forma el ciclo se ejecutará indefinidamente y necesitaras interrumpirlo "abortarlo" con [Esc].

Para insertar un ciclo "while":

- Seleccionar el lugar donde desea colocar el ciclo "while"
- Seleccionar la paleta de programación que contiene los operadores de programación
- Seleccionar el botón "while" (predefinida en Mathcad, no escribirla como tal)
- En el cuadro a la derecha del while teclear la expresión booleana. Selecciona el botón "Add Line" de la paleta de programación en caso de que se necesite para escribir más instrucciones
- En el lugar abajo del while colocar las instrucciones que se requiera ejecutar repetidamente. Usar el botón de "Add Line" cuando se necesite colocar más instrucciones.

```
Num(N) := while Conteo ≤ N
  Conteo ← Conteo + 1
  n ← mod(N, Conteo)
  m ← m + 1 if n = 0
  m
Numero ← "No primo" if m ≠ 2
Numero ← "EscribirNumeroPositivo" if N < 0
Numero ← "Primo" otherwise
```

Num(-10) = "EscribirNumeroPositivo"

Num(8) = "No primo"

Num(17) = "Primo"

```
demowhile(semilla) := x ← semilla
  while x < 100
    x ← x2.3
  x
```

demowhile(2) = 4.599×10^3

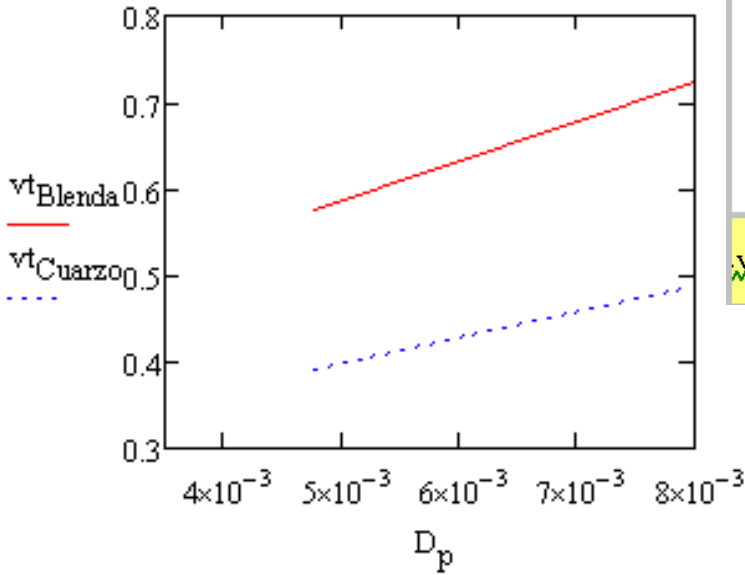
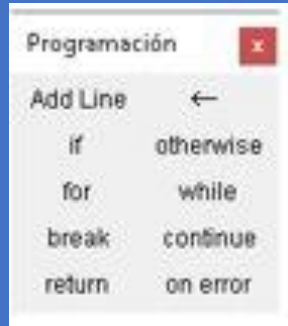
demowhile(10) = 199.526

demowhile(200) = 200

demowhile(-3) = ■

These values can not be compared.

PROGRAMACIÓN



Ejemplo:
 Ensayo de elutriación (separación fluidodinámica entre 2 sólidos de igual tamaño y diferente densidad)

Incógnitas: velocidad terminal del sólido, coeficiente de arrastre y Reynolds

Ciclo de programación para resolver los parámetros mencionados para el sólido 1 (blenda), para el tamaño de partícula mas pequeño del sistema

BLENDAS - Dinf

$vt_{Blenda1} := 100 \frac{cm}{s}$ $D_{inf} = 4.76 \cdot mm$

$ciclo(vt_{Blenda1}) :=$

$$Re_{Blenda1} \leftarrow \frac{D_{inf} \cdot vt_{Blenda1} \cdot \rho_w}{\mu_w}$$

$$Cd_{Blenda1} \leftarrow \frac{24}{Re_{Blenda1}} \cdot \left[1 + \left[8.171 \cdot e^{(-4.0655 \cdot \varphi_b)} \right] \cdot Re_{Blenda1}^{(0.0964 + 0.5565 \cdot \varphi_b)} \right] + \frac{\left[73.69 \cdot e^{(-5.0748 \cdot \varphi_b)} \cdot Re_{Blenda1} \right]}{\left[Re_{Blenda1} + 5.378 \cdot e^{(6.2122 \cdot \varphi_b)} \right]}$$

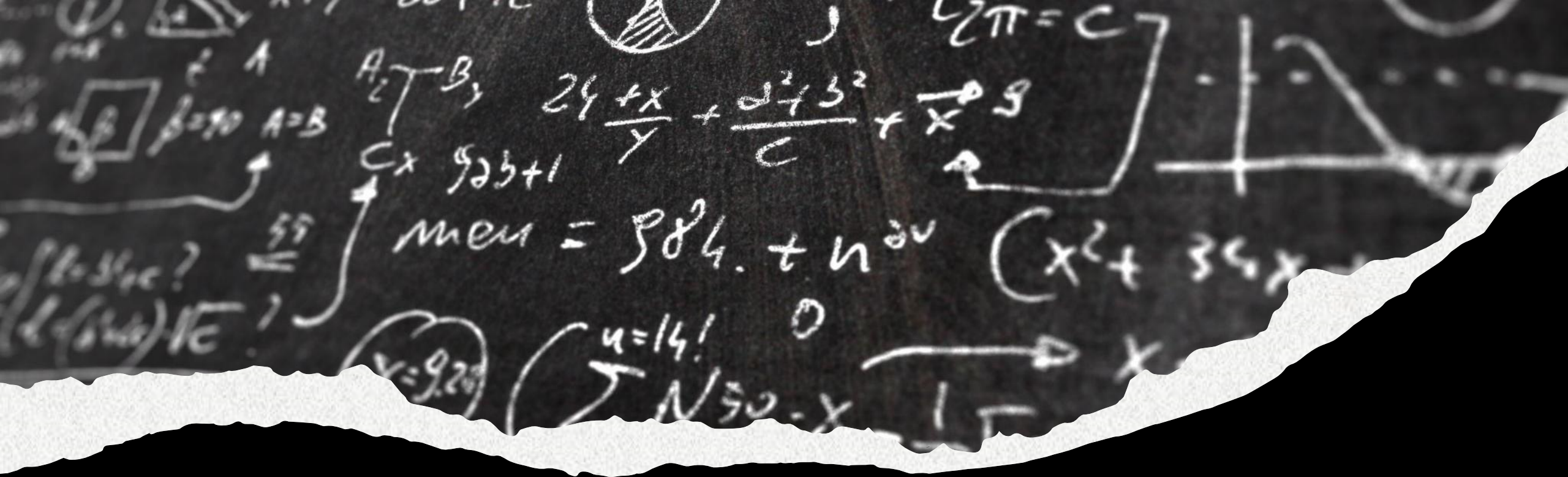
$$vt_{Blenda1}' \leftarrow \sqrt{D_{inf} \cdot g \cdot \frac{(\rho_{blenda} - \rho_w)}{\rho_w \cdot Cd_{Blenda1}} \cdot \frac{4}{3}}$$

$vt_{Blenda1}' - vt_{Blenda1}$

$vt_{Blenda1} := root(ciclo(vt_{Blenda1}), vt_{Blenda1}) = 0.576 \frac{m}{s}$

$$Re_{Blenda1} := \frac{D_{inf} \cdot vt_{Blenda1} \cdot \rho_w}{\mu_w} = 3.421 \times 10^3$$

$$Cd_{Blenda1} := \frac{24}{Re_{Blenda1}} \cdot \left[1 + 8.171 \cdot e^{(-4.0566 \cdot \varphi_b)} \cdot Re_{Blenda1}^{(0.0964 + 0.5565 \cdot \varphi_b)} + \frac{73.69 \cdot e^{-5.0748 \cdot \varphi_b} \cdot Re_{Blenda1}}{Re_{Blenda1} + 5.378 \cdot e^{6.2122 \cdot \varphi_b}} \right] = 0.207$$



PRÁCTICA UNIDAD III

Disponible en AV: **Trabajo Práctico UNIDAD III.pdf**

Modalidad: GRUPAL (4-6 integrantes)

Fecha de entrega: hasta el 16/08 inclusive

Enviar al correo: dolores.jimenez@fi.unju.edu.ar