Cómo definir variables en Matlab:

- input: A = 15 → Matlab guardará este valor hacia el futuro.
- input: A = 10 → Matlab Sobreescribirá esta variable, y guardará el nuevo valor hacia el futuro.

Operaciones Básicas:

→ Partiremos definiendo las variables A, B y C, y luego veremos como operarlas:

- input: A = 4
- input: B = 2
- input: C = -1

↳ Ahora, podemos pedirle a Matlab que nos opere nuestras variables:

- input: A + 1
  ↳ output: 5
- input: A + B
  ↳ output: 6
- input: B - A
  ↳ output: -2
- input: A/B → (A dividido B)
  ↳ output: 2
- input: A + C
  ↳ output: 3
- input: A ^ C → (A elevado a C)
  ↳ output: 0,25
- input: A * C → (A multiplicado por C)
  ↳ output: -4

Cómo definir un vector y una matriz:

→ Es posible para nosotros definir una variable como un vector o como una matriz. Esto lo haremos de la siguiente manera:

- input: A = [1 2 3; 4 5 6; 7 8 9]
  ↳ Output: $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$ → Notar que los elementos dentro de la misma fila pero distinta columna se separan con espacios, y las distintas filas se separan con un punto y coma ";".
- input: B = [1 2 3]
  ↳ Output: B = [1 2 3] → Si solo ponemos espacios, la variable queda como un vector.

↳ También podemos definir una matriz de ceros, y una matriz identidad con funciones incluidas en matlab:

- input: Z = zeros(3, 3) → notar que el primer índice indicará el número de filas, y el segundo el de columnas.
  ↳ Output: $Z = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
- input: I = eye(3, 3) → Nuevamente, el primer índice indicará el número de filas, y el segundo el de columnas.
  ↳ output: $I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Operaciones matriciales básicas:

→ Partiremos definiendo las variables A, B y C, y luego veremos como operarlas:

- input: A = [1 2 1; 2 4 3; 2 1 1]
- input: B = [2 4 1]
- input: C = [2; 3; 2]
- input: D = eye(3).*3 + 1 → Notar que multiplicar una matriz por un escalar va a multiplicar todos los elementos de la matriz por ese escalar, y que si le sumamos un escalar, ocurrirá lo mismo.

- input: A + D
  - Output: $\begin{bmatrix} 5 & 3 & 2 \\ 3 & 8 & 4 \\ 3 & 2 & 5 \end{bmatrix}$

- input: A - D
  - Output: $\begin{bmatrix} -3 & 1 & 0 \\ 1 & 0 & 2 \\ 1 & 0 & -3 \end{bmatrix}$

- input: transpose (B) o bien B.' → Notar que ambas formas hacen lo mismo, es decir trasponer la matriz. (B! = B punto apóstrofe)
  - output: $\begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix}$

- input: A * C → Notar que esto es una multiplicación.
  - Output: $\begin{bmatrix} 10 \\ 22 \\ 9 \end{bmatrix}$

↳ En el caso que tengamos un sistema de ecuaciones del siguiente estilo:

$$\underbrace{\begin{bmatrix} 1 & 6 & 9 \\ 4 & 3 & 1 \\ 9 & -2 & 6 \end{bmatrix}}_{[A]} \cdot \underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{[C]} = \underbrace{\begin{bmatrix} 8 \\ 1 \\ 13 \end{bmatrix}}_{[B]}$$

→ Es lo mismo que $[A]\cdot[C] = [B]$, por lo que sabemos que $[C] = [A]^{-1}\cdot[B]$. Para resolver esto en Matlab, debemos:

- input: A\B o bien inv(A) * B → Si bien a grandes razgos ambas operaciones hacen lo mismo, se recomienda usar el operador "\".
  - Output: $\begin{bmatrix} 0{,}4869 \\ -0{,}2638 \\ 1{,}3440 \end{bmatrix}$

↳ Ahora, imaginemos que queremos generar una matriz E, que se componga de las primeras dos filas de la matriz A, y luego de las primeras dos filas de la matriz D. Es decir:

$$[A] = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 3 \\ 2 & 1 & 1 \end{bmatrix} \qquad [D] = \begin{bmatrix} 4 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 4 \end{bmatrix}$$

entonces, nosotros buscamos $[E] = \begin{bmatrix} 1 & 2 & 1 & 4 & 1 & 1 \\ 2 & 4 & 3 & 1 & 4 & 1 \end{bmatrix}$

Para hacer esto en Matlab, haremos lo siguiente:

- input: E = [A(1:2, :) D(1:2, :)] → Aquí hay varias cosas que analizar. Primero, tenemos el corchete [ ], el que indica la generación de una matriz. Luego tenemos los Paréntesis () después de la matriz A y D, el que nos permite especificar qué elementos de dentro de la matriz seleccionaremos. Finalmente, tenemos los dos puntos :, los que nos indicarán el rango que deseamos. (1:2, 3:6) → Esto significaría que tomamos el rango de la fila 1 a la 2, y de la columna 3 a la 6. Si los dos puntos se encuentran "solos", indican que se toma todo ese rango. (1:2, :) → Esto significa que tomaremos de la fila 1 a la 2, y que se incluirán todas las columnas.
  - Output: $\begin{bmatrix} 1 & 2 & 1 & 4 & 1 & 1 \\ 2 & 4 & 3 & 1 & 4 & 1 \end{bmatrix}$

↳ También es importante notar como es que matlab está concatenando las matrices. En el ejemplo anterior, nuestro input fue el siguiente:

E = [A(1:2, :) D(1:2, :)]

↳ Notar que se usó un espacio entre A y D. Si recordamos de antes, los espacios se usan para separar elementos dentro de una misma fila, por lo que matlab genera lo siguiente:

E = [Elementos matriz A Elementos Matriz D] → Es casi como si fuese una matriz de matrices, por lo que si SOMOS CONSISTENTES CON LAS DIMENSIONES, Matlab nos arrojará una nueva matriz, generada de la concatenación de las matrices A y D.

↳ Así, podríamos también hacer lo siguiente:

- input: E = [A(1:2, :); D(1:2, :)]
  - ↳ Output:

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 3 \\ 4 & 1 & 1 \\ 1 & 4 & 1 \end{bmatrix}$$

↳ Para encontrar los valores propios de la matriz [A], simplemente debemos:

- input: eig(A) → lo que nos entrega un vector con los valores propios de la matriz.

Cómo definir una función:

→ Para definir una función, no podemos hacerlo desde la ventana de comandos, sino que la tenemos que definir en un archivo de Script. Para ello, debemos hacer click en "Nuevo Script", y nombramos al archivo de la misma forma que nombraremos a la función. Así, si nuestra función será "f(x)", el archivo deberá llamarse "f.m", donde el ".m" es la extensión. Así, si queremos definir la función $f(x) = x \cdot 3 + 6$, debemos:

- Input:

```
function y = f(x)
    y = x*3 + 6;
end
```

↳ Luego, para llamar a la función, simplemente debemos:

- Input: f(3)
  - ↳ Output: 15 → Notar que si el archivo se llama "función.m", debemos llamarla poniendo "función(3)".