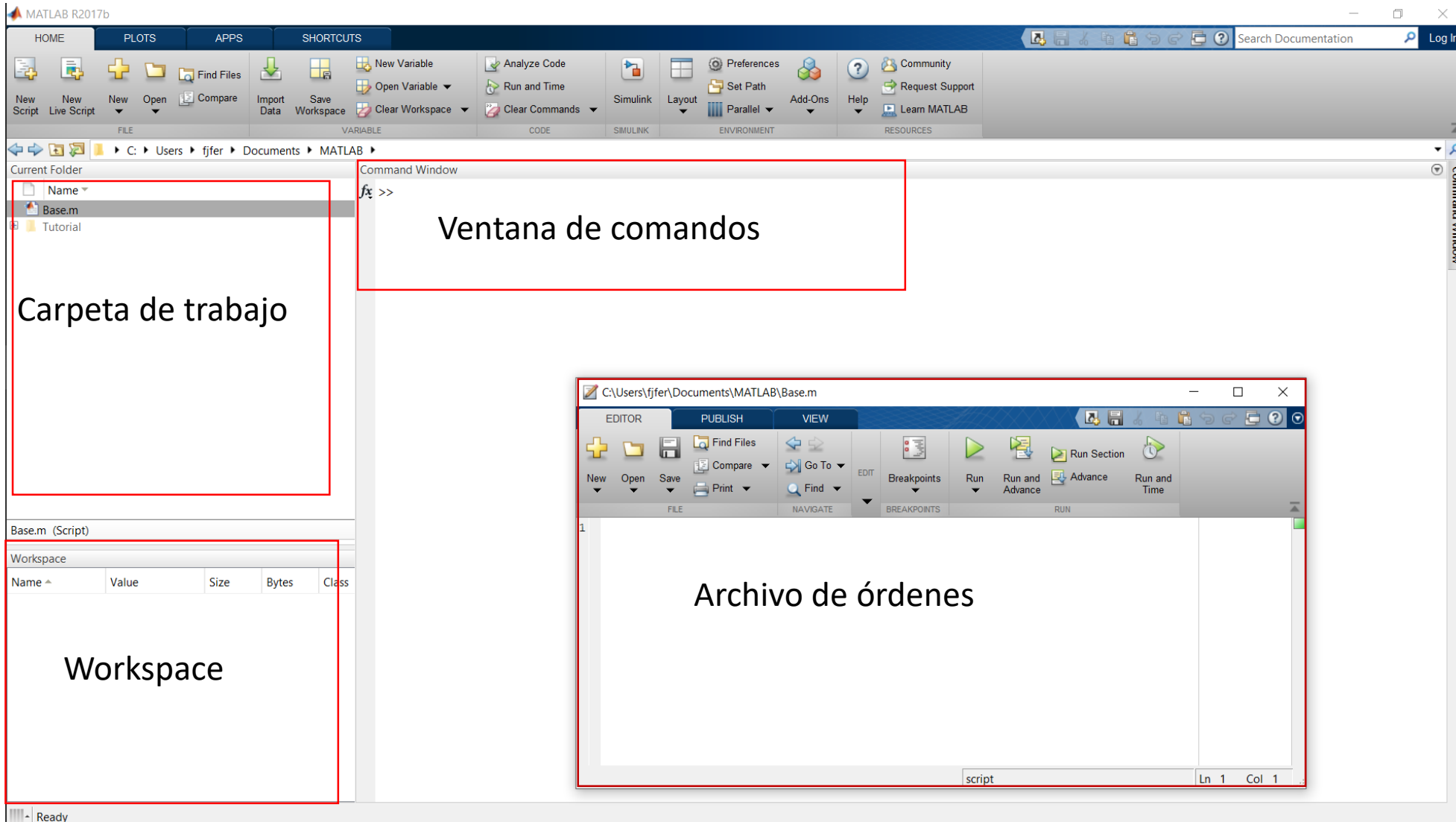


# LABORATORIO ING201

# Qué es MatLab?

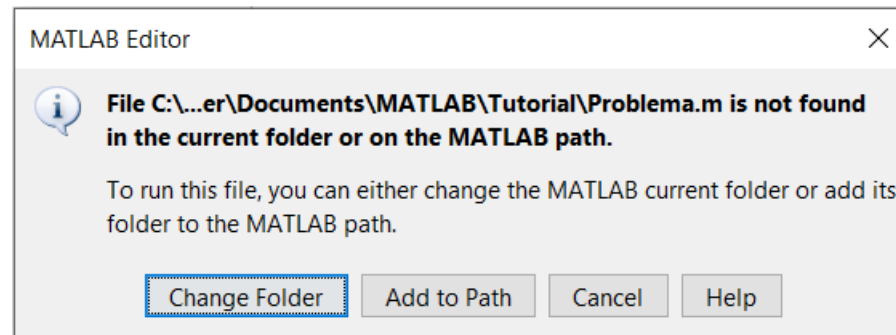
- Podemos mencionar como sus funciones principales el trabajo con matrices, la representación de datos, desarrollo e implementación de algoritmos, el desarrollo de GUI y la intercomunicación con programas que implementan distintos lenguajes. [mathworks](https://www.mathworks.com/)
- Principalmente nos enfocaremos en el desarrollo de algoritmos que trabajen con matrices y vectores para optimizar situaciones deterministas y estocásticas.

# Intro



# Carpeta de trabajo

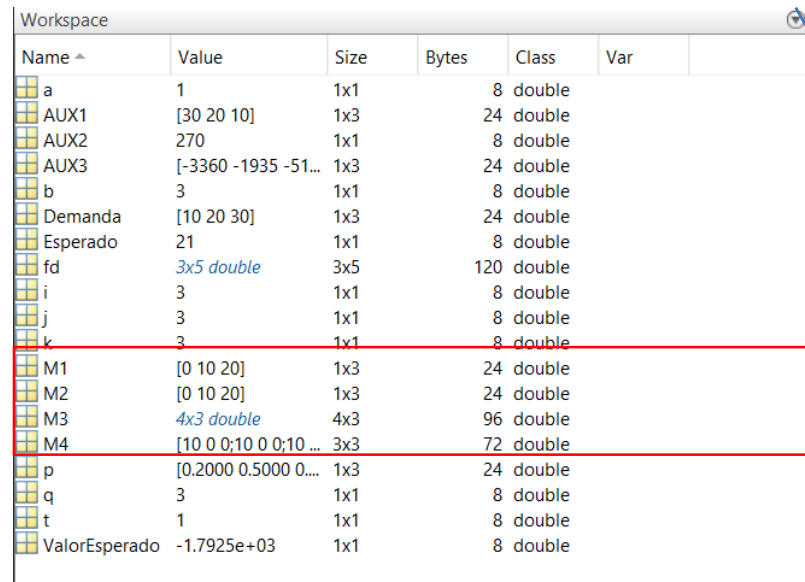
- En esta sección es donde se encuentran todos los archivos de órdenes (script) que pueden ser ejecutables en la ventana de comandos. Para poder correr archivos.m que no se encuentren en la carpeta, tendremos que incorporarlos con **“add to path”**.



- La terminación .m es importante para informarle al programa que se trata de un archivo con lenguaje M (sintaxis propia de MatLab). Lo mismo que pasa con las terminaciones .py (Python) o .cpp (C++)

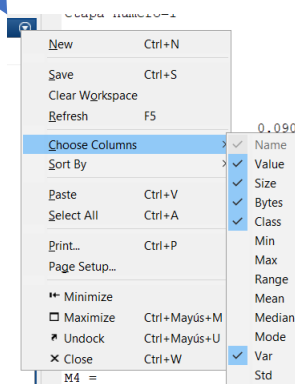
# Workspace

- Aquí se muestran todas las variables que se están trabajando en la ventana de comandos, con algunos de sus atributos. Por ejemplo si trabajamos un algoritmo con variables M1, M2, M3 y M4, el workspace nos mencionará los atributos de estas variables.



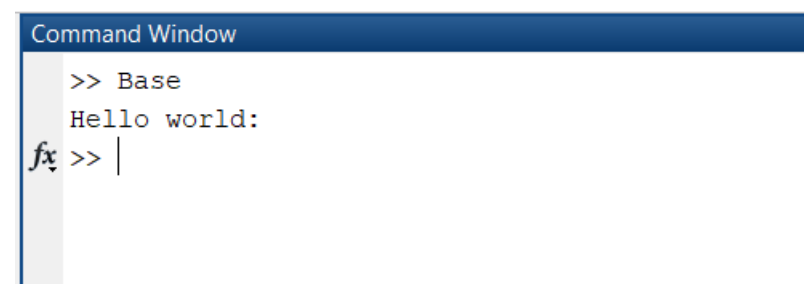
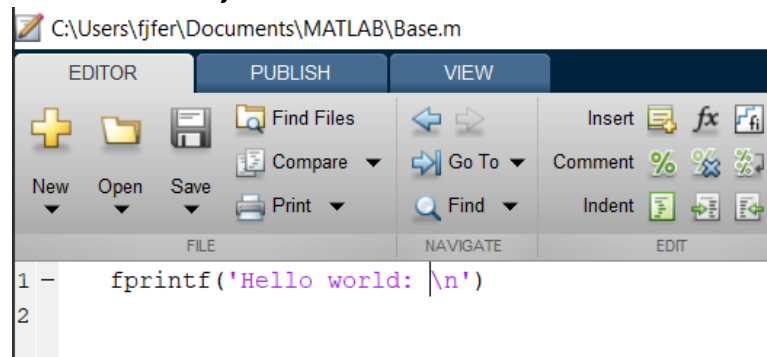
Name	Value	Size	Bytes	Class	Var
a	1	1x1	8	double	
AUX1	[30 20 10]	1x3	24	double	
AUX2	270	1x1	8	double	
AUX3	[-3360 -1935 -51...	1x3	24	double	
b	3	1x1	8	double	
Demanda	[10 20 30]	1x3	24	double	
Esperado	21	1x1	8	double	
fd	3x5 double	3x5	120	double	
i	3	1x1	8	double	
j	3	1x1	8	double	
k	3	1x1	8	double	
M1	[0 10 20]	1x3	24	double	
M2	[0 10 20]	1x3	24	double	
M3	4x3 double	4x3	96	double	
M4	[10 0 0;10 0 0;10 ...	3x3	72	double	
p	[0.2000 0.5000 0...	1x3	24	double	
q	3	1x1	8	double	
t	1	1x1	8	double	
ValorEsperado	-1.7925e+03	1x1	8	double	

- Por defecto nos arrojará los atributos de nombre, valor, tamaño (por dimensiones), cantidad de espacio ocupada y clase. Sin embargo esto puede ser modificado en la siguiente sección:



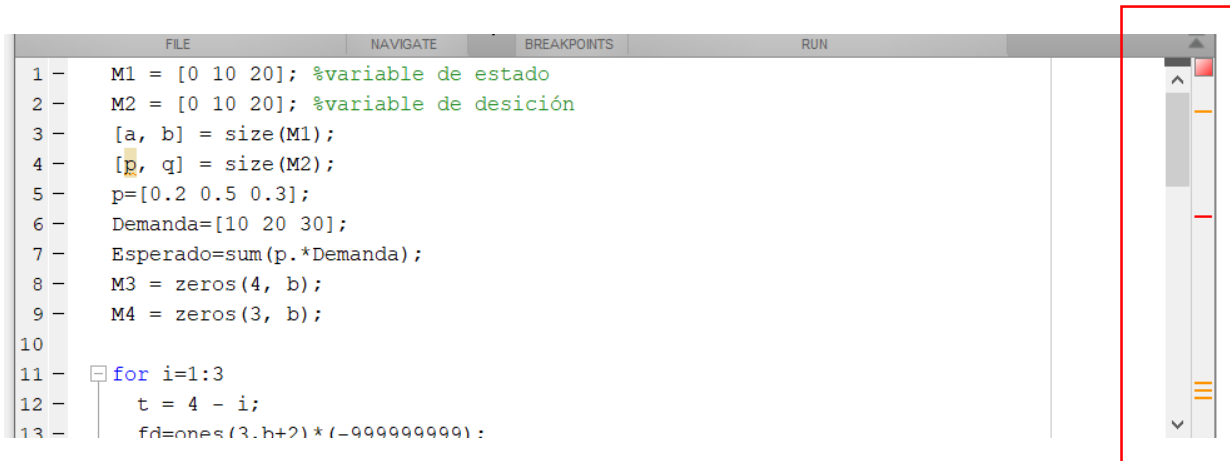
# Ventana de comandos

- La ventana de comandos permite hacer operaciones simples y/o escribir el algoritmo completo ahí. Sin embargo esta no está optimizada para escribir código. Comúnmente se utilizan los script(archivos de órdenes) para crear el código, y la ventana de comandos para visualizar valores de variables específicas y ejecutar archivos de programas.
- Para ejecutar un archivo, solo se tiene que escribir de manera idéntica el nombre, sin el “.m”.

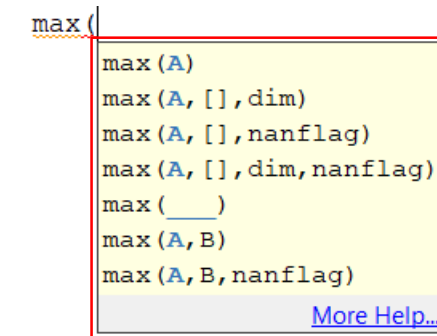


# Archivo de órdenes

- Aquí es donde desarrollaremos los códigos, la ventana está diseñada para marcarnos los errores de sintaxis a la derecha, junto con la visualización de contenido de ayuda al momento de escribir las funciones:



```
1 - M1 = [0 10 20]; %variable de estado
2 - M2 = [0 10 20]; %variable de decisión
3 - [a, b] = size(M1);
4 - [p, q] = size(M2);
5 - p=[0.2 0.5 0.3];
6 - Demanda=[10 20 30];
7 - Esperado=sum(p.*Demanda);
8 - M3 = zeros(4, b);
9 - M4 = zeros(3, b);
10
11 - for i=1:3
12 -     t = 4 - i;
13 -     fd=ones(3,b+2)*(-999999999);
```



```
max(  
max(A)  
max(A, [], dim)  
max(A, [], nanflag)  
max(A, [], dim, nanflag)  
max(__)  
max(A, B)  
max(A, B, nanflag)  
More Help...
```

# Sintaxis (enfocado al trabajo algebraico)

- Algunos de los aspectos principales que tenemos que conocer sobre el lenguaje se presentan a continuación:


```
%FORMAS DE COMENTAR:  
    % SOLO CON % PARA COMENTAR POR LINEA  
    %{  
    Así se comentan parrafos  
    %}  
  
%CREAR VARIABLES:  
N = 5; %";" se utiliza cuando no queremos que se muestre la operación en la ventana de comandos  
  
%CREAR VECTORES:  
M = [1 2 3 4 5 6] %creamos un vector con 6 elementos  
  
%CREAR MATRICES:  
M1 = [1 2 3  
      4 5 6] % en vez de "enter" podemos diferenciar las filas con ;  
M2 = [1 2 3;4 5 6]
```






# Sintaxis

- Para revisar los aspectos de escritura de las principales funciones de programación, los invito a visualizar la página de documentación de mathworks adjunto en el anexo.

Motor de búsqueda: En el ingresan las funciones que desean implementar para conocer su campo de trabajo, sintaxis entre otros

 [Productos](#) [Soluciones](#) [Educación](#) [Soporte](#) [Comunidad](#) [Eventos](#)

[Documentación](#) [Todo](#) [Ejemplos](#) [Funciones](#)  [Documentation](#) 

[Consiga MATLAB](#)  

[Trials](#) [Product Updates](#)

[CONTENTS](#) [Cerrar](#)

[« Inicio de Documentación](#)  
[« MATLAB](#)  
[« Programming Scripts and Functions](#)

**Control Flow**  
Scripts  
Functions  
Live Scripts and Functions  
Files and Folders  
Debugging  
Code Editor  
Code Analysis and Execution

Esta página aún no se ha traducido para esta versión. [Puede ver la versión más reciente de esta página en inglés.](#)

## Control Flow

Conditional statements, loops, branching

### Sintaxis del lenguaje de MATLAB

<code>if, elseif, else</code>	Ejecutar instrucciones si la condición es verdadera
<code>for</code>	for loop para repetir el número especificado de veces
<code>parfor</code>	Parallel for loop
<code>switch, case, otherwise</code>	Ejecutar uno de varios grupos de instrucciones
<code>try, catch</code>	Execute statements and catch resulting errors
<code>while</code>	Bucle while para repetir cuando la condición es verdadera

<code>break</code>	Terminar la ejecución de bucle for o while
<code>continue</code>	Pass control to next iteration of for or while loop
<code>end</code>	Terminate block of code, or indicate last array index
<code>pause</code>	Stop MATLAB execution temporarily
<code>return</code>	Return control to invoking function

### Temas

**Sentencias condicionales**  
Para determinar qué bloque de código se debe ejecutar en tiempo de ejecución, utilice las instrucciones condicionales `if` o `switch`.

**Comandos de control de bucle**  
Para ejecutar repetidamente un bloque de código, utilice bucles `for` y `while`.

**R2018b**

# Anexo

- [Documentación MatLab](#)