En Octave (busque Octave GUI, si no le apareció el ícono al instalarlo):

Tipee lo que figura en cada línea, a continuación del >> y pulse ENTER al final de la línea. Es sensible a mayúsculas y minúsculas.

```
>> a=1:10
>> b=ones(1,length(a))
>> c=a*b'
```

Interprete lo que se obtuvo en la variable c.

Pruebe qué hace la comilla vertical:

>> b'

En el sector llamado "espacio de trabajo" o "workspace" puede ver qué variables se definieron, su clase y su valor.

Pruebe estas otras opciones:

>> d = 1:2:13

>> d= 20: -1 : 3

>> d = 2: 0.2: 4

¿Qué es lo que generan los ":"?

La estructura es: inicio:paso:final (puede alcanzarse o no, el valor final, pero no se supera)

En un vector, el primer elemento es d(1)

>> d(1)

>> d(2)

Obtendremos el valor de c de otra forma:

```
>> acum=0
>> l=length(a)
>> for i=1:l
acum=acum+a(i);
end
>> acum
```

Para saber qué hace un comando:

>> help length

o también:

>> help("length")

Observe que hasta no terminar de definir el bucle for, no vuelve a aparecer el >>.

¿Para qué es el ";" en el interior del bucle for? (Sug.: pruebe borrándolo).

¿Se obtuvo el mismo resultado en **acum** que en **c**?

Vamos a medir la eficiencia, en tiempo, de ambos procedimientos.

Los comandos para medir el tiempo son tic y toc (para iniciar y parar de medir)

Para no editar en la línea de comandos, vamos a anotar todos los comandos en un archivo de texto, llamado SCRIPT.

Al "ejecutarlo", luego de guardado, se interpretará línea a línea.

La idea es ir incrementando por 10 la longitud de a, y comparar ambos métodos.

¿Hasta cuando incrementar? Eso es lo que queremos ver.

Avísenme por el chat si llegaron hasta acá.