Sumas acumuladas y diferencias sucesivas en el lenguaje de MATLAB (cumsum y diff)

Objetivos. Practicar el uso de las funciones `cumsum` y `diff` del lenguaje de MATLAB.

Requisitos. Creación de vectores en el lenguaje de MATLAB.

1. Análogos libres de MATLAB. Los ejemplo abajo está escritos en el dialecto GNU Octave del lenguaje de MATLAB. Otras buenas opciones son Scilab y FreeMat.

2. Sumas acumuladas. Dado un arreglo, podríamos calcular sus sumas acumuladas con el siguiente código:

```
a = [3; -5; 7; 1]
s = a;
for j = 2 : length(s),
  s(j) = s(j) + s(j - 1);
endfor
```

La función `cumsum` permite hacer lo mismo de manera más eficiente:

```
a = [3; -5; 7; 1]
s = cumsum(a)
```

3. Diferencias sucesivas. Dado un arreglo, podríamos calcular sus diferencias sucesivas con el siguiente código:

```
a = [3; -5; 7; 1]
d = zeros(length(a) - 1, 1);
for j = 1 : length(a) - 1,
  d(j) = a(j + 1) - a(j);
endfor
```

El ciclo se puede sustituir por una operación vectorial:

```
a = [3; -5; 7; 1]
d = a(2 : end) - a(1 : end - 1)
```

La función `diff` hace lo mismo, pero de manera más eficiente:

```
a = [3; -5; 7; 1]
d = diff(a)
```

Las funciones cumsum y diff son casi inversas una para la otra

4. Aplicar diff después de cumsum.

```
a = [3 -5 7 1]
b = cumsum(a)
c = diff(b)
```

Se ve que se pierde la primer componente del arreglo original. Para recuperar el arreglo original completo, hay que copiar la primera componente:

```
a = [3 -5 7 1]
b = cumsum(a)
c = [b(1) diff(b)]
```

5. Aplicar cumsum después de diff.

```
f = [3 -5 7 1]
g = diff(f)
h = cumsum(g)
```

6. Ejercicio. Modificar el código 5 de tal manera que se recupere el arreglo original.