OBJETIVOS DE APRENDIZAJE

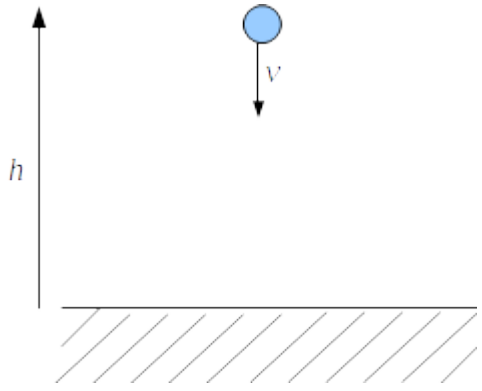
- **Integración numérica para simulación**
- **Revisión general:**
  - **Conditiones**
  - **Loops**
  - **Funciones**

EJERCICIO 9.1 INTEGRACIÓN NUMÉRICA

El objetivo es integrar la función $f(x)=\cos^2(x+pi)$ desde $x=a$ a $x=b$. La persona usuaria proporcionará los valores de entrada de *a*, *b* y n (el número de intervalos de integración), con lo que podremos calcular $\Delta x=(b-a)/n$ y el script deberá imprimir en pantalla el resultado de integrar la función en el intervalo de valores dado por la persona usuaria. Por ejemplo:

```
>> functionIntegration
Give me the minimum value of x: 3.14
Give me the maximum value of x: 6.28
Give me the number of integration slices: 100
The integrative of cos(x+pi)^2 in range [3.14,6.28] is 1.58
```

EJERCICIO 9.2 CAÍDA LIBRE



El objetivo es simular la altura (h) de un objeto en caída libre. El objeto está suspendido en el aire con *v=0*, y su velocidad (v) hacia la Tierra se incrementa con la aceleración correspondiente a la gravedad ($G=9.8\,m/s^2$). Al altura del objeto queda especificada por las siguientes ecuaciones:

$$\dot{v}=G$$
$$\dot{h}=-v$$

las cuales pueden ser discretizadas de la siguiente manera:

$$v_{t+1}=v_t+G\,\Delta t$$
$$h_{t+1}=h_t-v\,\Delta t$$

Además, asumiremos que el objeto no rebotará desde el suelo y que su posición permanecerá fija una vez haya llegado al suelo (*h=0*).

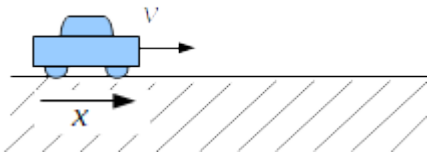
1. Escribe la siguiente función:

```
function [t,h] = simulate_free_fall (h_0, total_time , delta_t)
% This function simulates the free fall of an object that
% is initially 'h_0' meters above the ground
% It returns two vectors:
%   -t: the time elapsed
%   -h: the height of the object
```

2. Implementa un script que muestre gráficamente (plot) la posición de un objeto durante 10 segundos, con punto de partida h = 20 y asumiendo *delta_t=0.01*.

3. Implementa un script que muestre en un gráfico (plot) los mismos datos pero con los siguientes valores para *delta_t: 0.5, 0.1, 0.01* y *0.001*.

EJERCICIO 9.3 VEHÍCULO CON VELOCIDAD CONSTANTE



Un coche se mueve a lo largo del eje *x* con velocidad constante *v= 5m/s*. Al empezar la simulación (*t=0 s*), su posición es *x=10 m*. La ecuación diferencial que describe el sistema es:

$$\dot{x} = v$$

la cual puede ser discretizada de la siguiente manera:

$$x_{t+1} = x_t + v\,\Delta t$$

1. Define una función (*simulate_constant_speed*) que devuelva dos vectores con la evolución del tiempo y posición durante el intervalo de tiempo deseado *[0,total_time],* utilizando *delta_t* como paso (*time-step*) de la integración.

```
function [t,x] = simulate_constant_speed (x_0, total_time , delta_t)
% This function simulates the movement of a vehicle with constant
speed
% in position x=x_0 meters
% It returns two vectors:
%   -t: the time elapsed
%   -x: the position of the vehicle
```

2. Implementa un script que muestre en un gráfico su posición en el intervalo *t=[0,60]* utilizando un paso de tiempo de *0.1 s*.

EJERCICIO 9.4 FRENADO DEL VEHÍCULO

Teniendo en cuenta las mismas condiciones que en el ejercicio anterior ($x_0 = 10\,m$ y $v = 5\,m/s$), supongamos que el conductor ve un unicornio en $t = 2$ y, asustado, empieza a pisar el freno, decelerando el coche con una aceleración negativa $a = -0.2 m/s^2$. Considerando que la velocidad no puede ser negativa por el efecto de los frenazos, calcular la posición y la velocidad del coche en *t=[0,60]*. Utiliza un paso de integración *delta_t=0.01*. El sistema se describe mediante las ecuaciones diferenciales:

$$\dot{v} = a$$
$$\dot{x} = v$$

las cuales pueden ser discretizadas del siguiente modo:

$$a_t = \begin{cases} 0 if\ t > 2 \\ -0.2 otherwise \end{cases}$$
$$v_{t+1} = v_t + a_t \Delta t$$
$$x_{t+1} = x_t + v_t \Delta t$$

1. Define a función que devuelva las variables integradas durante la simulación: tiempo (*t*), velocidad (*v*) y posición (*x*). Las entradas de esta función serán las mismas que en los ejercicios anteriores.

```
function [t,x] = simulate_car_break (x_0, total_time , delta_t)
% This function simulates the movement of a vehicle that applies
% the brakes.
% It returns three vectors:
%   -t: the time elapsed
%   -x: the position of the vehicle
%   -v: the speed of the vehicle
```

2. Codifica un script que represente gráficamente tanto la variable *x* como la *v* con respecto a *t* en dos subplots diferentes (x vs. t y v vs. t). Además, comprueba visualmente que las posiciones del coche permanecen fijas desde el momento en que alcanza la velocidad de *0 m/s*.