

Taller Introducción a la Ingeniería Eléctrica Microcontrolador Arduino y Comunicaciones

Instituto de Ingeniería Eléctrica

Taller 2: Introducción a la programación y a Arduino

19 de Marzo de 2024

- 1 Recapitulando
- 2 Control de flujo
 - Ejercicio 1 - Clase
- 3 Entradas analógicas
 - Ejercicio 2 - Clase
- 4 Salidas con PWM
 - Ejercicio 3 - Clase
- 5 Materiales
 - Ejercicio 4 - Clase
 - Ejercicio 5 - Clase
- 6 EJERCICIOS para la próxima clase
 - Ejercicio 1
 - Ejercicio 2
 - Ejercicio 3

Esquema de la presentación

- 1 Recapitulando
- 2 Control de flujo
 - Ejercicio 1 - Clase
- 3 Entradas analógicas
 - Ejercicio 2 - Clase
- 4 Salidas con PWM
 - Ejercicio 3 - Clase
- 5 Materiales
 - Ejercicio 4 - Clase
 - Ejercicio 5 - Clase
- 6 EJERCICIOS para la próxima clase
 - Ejercicio 1
 - Ejercicio 2
 - Ejercicio 3

Resapitulando

¿Qué vimos la semana pasada?

- Estuvimos trabajando con el Arduino, un **MICROCONTROLADOR** que maneja pines analógicos y digitales y permite hacer muchas cosas a partir de eso.
- Hasta ahora nos focalizamos en el manejo de pines digitales, usándolos como salidas para encender y apagar leds (funciones: *pinMode()*, *digitalWrite()*).
- Fuimos combinando el manejo de los pines para definir distintas respuestas del micro. La idea de *sensar* el entorno y *actuar* en consecuencia.
- Comenzamos con control de flujo: sentencias *if* (ramificación).

Esquema de la presentación

- 1 Recapitulando
- 2 Control de flujo
 - Ejercicio 1 - Clase
- 3 Entradas analógicas
 - Ejercicio 2 - Clase
- 4 Salidas con PWM
 - Ejercicio 3 - Clase
- 5 Materiales
 - Ejercicio 4 - Clase
 - Ejercicio 5 - Clase
- 6 EJERCICIOS para la próxima clase
 - Ejercicio 1
 - Ejercicio 2
 - Ejercicio 3

Control de flujo

Sentencia "for"

Flujo de **repetición** sobre una cantidad de iteraciones conocida.

```
for (inicializo variable de control; condicion; paso o
    incremento) {
    codigo a repetir
}
```

sintaxis

Ejemplo:

```
for (int i = 2; i <= 4; i++) {
    digitalWrite(i, HIGH);
    delay(1000);
    digitalWrite(i, LOW);
    delay(1000);
}
```

atajo para utilización de instrucción "for"

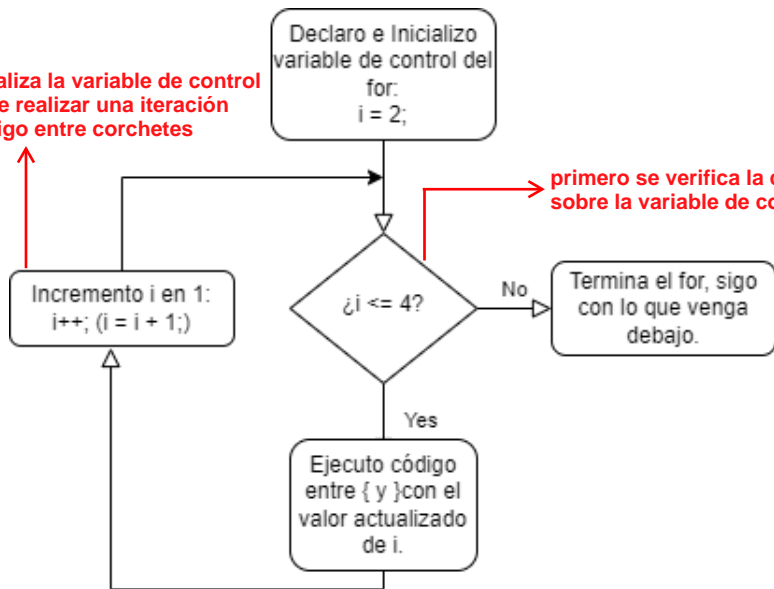
condición lógica (el resultado es "true" o "false")

Para la variable entera i , que posee el valor 2, se realizarán las acciones escritas dentro de los corchetes siempre que i sea \leq que 4 y si la condición anterior se cumple, lo primero que ocurre es que se actualiza su valor sumándole 1 (instrucción ++)

Control de flujo

Sentencia "for"

se actualiza la variable de control luego de realizar una iteración del código entre corchetes



Ejercicio clase

Ejercicio 1

Modificar el código del Ejercicio 3 de deberes del Taller 1 para lograr el mismo objetivo pero usando la sentencia "**for**".

Nota: Se pueden cambiar los pines de los leds o recordar sobre definición de vectores para facilitar.

Control de flujo

Sentencia "while"

```
while (condicion){
```

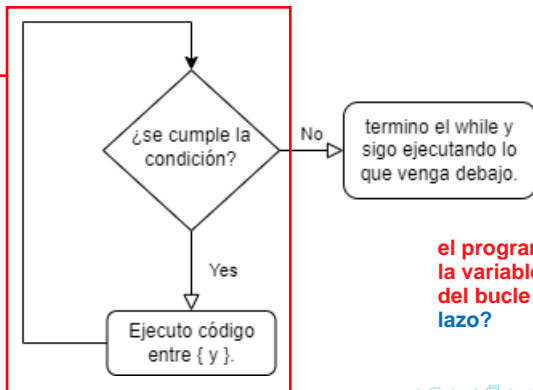
```
// instrucciones a repetir.
```

```
// siempre asegurarse que la condici0n cambiarA de valor  
 , para no quedar "atrapado" dentro del while.
```

```
}
```

→ **condición lógica**

Bucle o Lazo



el programador debe actualizar la variable de control dentro del bucle ¿qué es un bucle o lazo?

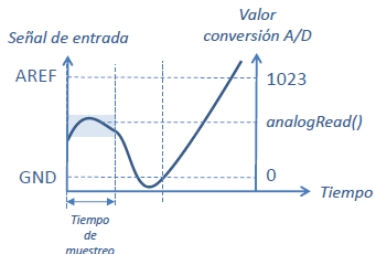
Esquema de la presentación

- 1 Recapitulando
- 2 Control de flujo
 - Ejercicio 1 - Clase
- 3 Entradas analógicas
 - Ejercicio 2 - Clase
- 4 Salidas con PWM
 - Ejercicio 3 - Clase
- 5 Materiales
 - Ejercicio 4 - Clase
 - Ejercicio 5 - Clase
- 6 EJERCICIOS para la próxima clase
 - Ejercicio 1
 - Ejercicio 2
 - Ejercicio 3

Entradas Analógicas del Arduino

Manejo de pines de entrada analógicos (serán las señales provenientes de algunos sensores).

- Los pines analógicos (A0 a A5) sólo sirven para **leer** señales, no para escribir.
- El Arduino tiene un convertidor Analógico/Digital (ADC), de 10 bits: recibe señales continuas entre 0 y 5V y retorna enteros entre 0 y 1023.
- Si la señal analógica está en 0V al momento de la lectura, el valor almacenado será el 0; si está a 5V será 1023. Para todos los valores de voltaje intermedios se ajustará al nivel entero correspondiente.
- Para leer el valor en un pin analógico, se debe usar la función **analogRead(pin)**:



Entradas Analógicas del Arduino

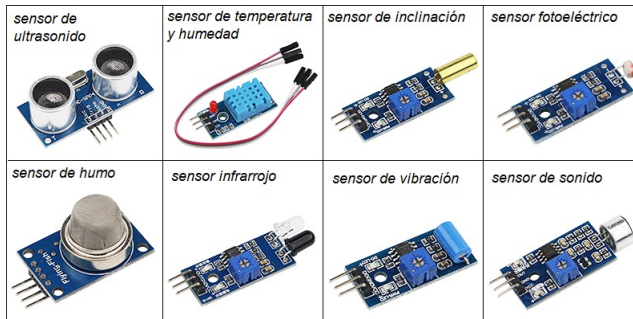
```
int dato; // variable que va a almacenar lo que se lee del
pin
int pin = 0; // pin A0 de entrada analogica
void setup(){
  // no es necesario inicializar nada
  ¿Por qué no usamos pinMode(pin,
  INPUT)?
}
void loop(){
  dato = analogRead(pin); // dato es un entero entre 0 y 1023
}
```

abuso de lenguaje



Figura: Analógico vs Digital

SENTIDOS (Sensores)



Sensores = Sentidos

Tomamos una muestra de energía de algún tipo, del ambiente y la convertimos en energía eléctrica que luego transferimos al microcontrolador a través de una señal eléctrica. De esta manera se puede monitorear casi cualquier magnitud física.

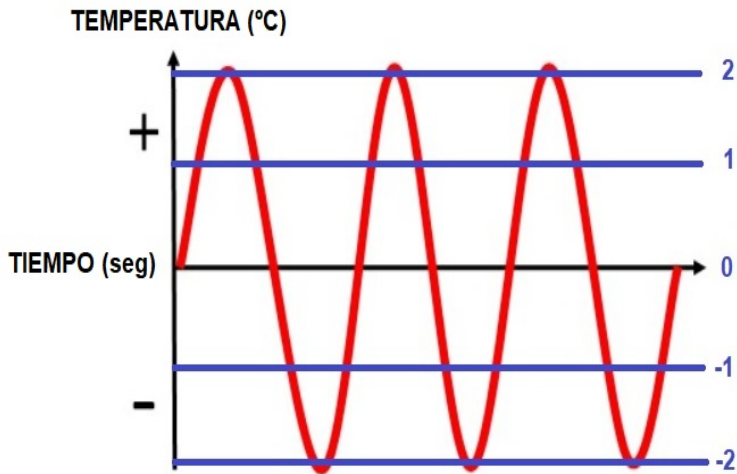


Figura: Discretización de señales

Conversión: Analógico a Digital

señal fácilmente manipulable por el Arduino

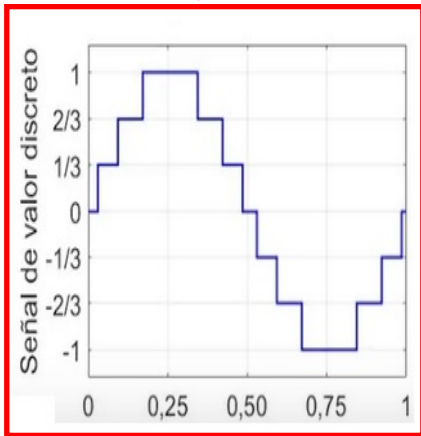
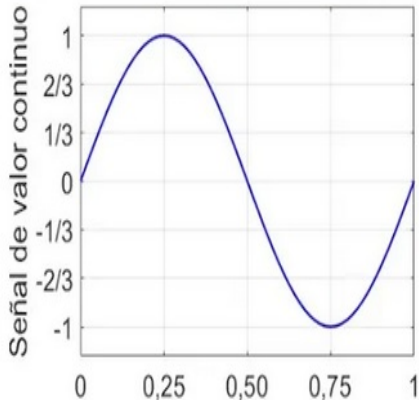


Figura: Discretización de señales

Analógico vs Digital

	Sistema digital	Sistema analógico
Definición	Sistema de manipulación de datos mediante dígitos (números).	Sistema de manipulación de datos físicos representados en valores continuos .
Valores de la señal	Valores discretos (finitos).	Valores continuos (infinitos).
Ventajas	<ul style="list-style-type: none">• Menor tamaño.• Eficiencia.• Precisión.• Diseño.• Estabilidad	<ul style="list-style-type: none">• Instantaneidad.• Economía.• Fidelidad.
Desventajas	<ul style="list-style-type: none">• Conversión.• Ancho de banda.• Alteración.	<ul style="list-style-type: none">• Menor tolerancia al ruido.• Degradación de la señal.• Dificultades técnicas.
Ejemplos	<ul style="list-style-type: none">• Computadoras.• Teléfonos móviles.• Sistemas de grabación de audio y video.• Instrumentos de precisión digitales.	<ul style="list-style-type: none">• Sistemas de audio y video antiguos.• Fotografía analógica.• Instrumentos de precisión tradicionales.

Figura: Discretización de señales

Pines digitales de I/O

Son pines que solo pueden tomar 2 valores de voltaje “ALTO” o “BAJO”, “HIGH” o “LOW”

- son 14 (0-13). Se deben usar primordialmente del 2-12
- los pines 3, 5, 6, 9, 10 y 11 pueden tomar valores intermedios, entre “ALTO” y “BAJO”. **Solo cuando funcionan como salida**
- los pines 0, 1, 13 tienen características especiales por eso se debe evitar usarlos

Pines análogos de entrada

- son 6 (A0-A5)
- solo funcionan como entradas

Pines I/O

- cada pin digital puede funcionar como entrada o como salida
- esta característica es programable (la persona debe escribir en el código en que condición funcionará cada pin digital utilizado)
- un pin debe funcionar como salida cuando el microcontrolador debe transmitir señales eléctricas a un dispositivo conectado a la placa de desarrollo
- un pin debe funcionar como entrada cuando el microcontrolador debe recibir señales eléctricas de un dispositivo conectado a la placa de desarrollo

Ejercicio - Entradas analógicas

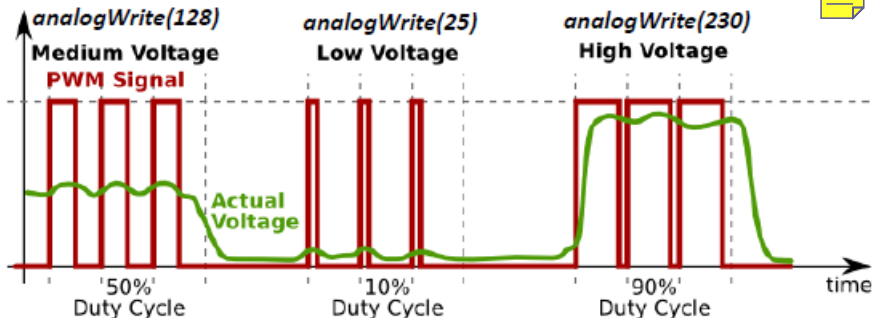
Determinar qué valor entero corresponde a una tensión de entrada de 3,5 volts en un pin de entrada analógica.

Esquema de la presentación

- 1 Recapitulando
- 2 Control de flujo
 - Ejercicio 1 - Clase
- 3 Entradas analógicas
 - Ejercicio 2 - Clase
- 4 Salidas con PWM**
 - Ejercicio 3 - Clase
- 5 Materiales
 - Ejercicio 4 - Clase
 - Ejercicio 5 - Clase
- 6 EJERCICIOS para la próxima clase
 - Ejercicio 1
 - Ejercicio 2
 - Ejercicio 3

Salidas PWM del Arduino

- Las salidas digitales rotuladas como PWM pueden imponer no solo 0V o 5V sino también 256 valores entre 0V y 5V. → **sintaxis**
- Para esto se utiliza la función `analogWrite(pinNumber, valor)` donde *pinNumber* es el número del pin de salida del Arduino y *valor* es un número entre 0 y 255.
- Si el valor es 0, impondrá 0V, si es 255 impondrá 5V y en otros casos, los valores intermedios correspondientes.



Ejercicio - Salidas PWM

Determinar el valor a poner en un pin de salida PWM para tener una tensión de 3,5 volts.

Ejemplo

Código Proceso de lectura de un voltaje y su posterior procesamiento para establecer un voltaje equivalente en un pin PWM de interés

En el siguiente código se lee el valor de una entrada analógica y se saca ese valor a través de una salida PWM.

```
int pwmPin = 9; // pin de salida que soporta PWM
int analogPin = A0; //pin de entrada analogica .
int val = 0; // variable para almacenar el valor leído y a
    escribir
float volt = 0; // variable para almacenar el voltaje leído
    en la entrada

void setup(){
pinMode(pwmPin, OUTPUT); // inicializa el pin como salida
}

void loop(){
val = analogRead(analogPin); // lee la entrada analogica
volt =(5.0 * val) / 1023; // calcula el voltaje leído
val = 255 * (volt / 5);
//calcula el valor a sacar que corresponde al voltaje leído
analogWrite(pwmPin, val); // saca el valor correspondiente
}
```


Esquema de la presentación

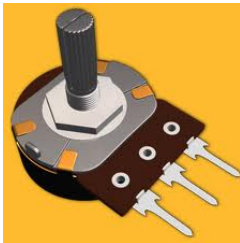
- 1 Recapitulando
- 2 Control de flujo
 - Ejercicio 1 - Clase
- 3 Entradas analógicas
 - Ejercicio 2 - Clase
- 4 Salidas con PWM
 - Ejercicio 3 - Clase
- 5 **Materiales**
 - Ejercicio 4 - Clase
 - Ejercicio 5 - Clase
- 6 EJERCICIOS para la próxima clase
 - Ejercicio 1
 - Ejercicio 2
 - Ejercicio 3

Potenciómetro

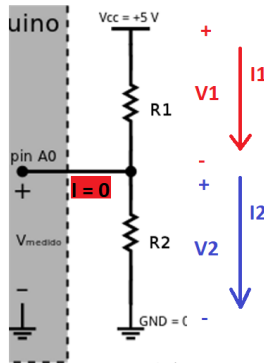
Divisor resistivo

Para simular entradas analógicas de diversos niveles, usaremos un Potenciómetro.

Su funcionamiento se basa en lo que se denomina el **Divisor Resistivo**: la misma corriente circula por ambas resistencias, por lo que la tensión total de alimentación "se reparte" entre ambas resistencias.



(a) Potenciómetro



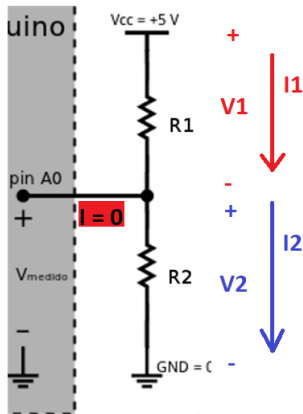
(b) Divisor Resistivo

Ejercicio clase

Ejercicio 4

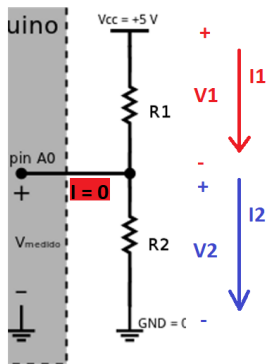
Ejercicio - Divisor Resistivo

¿Si la tensión de alimentación es 5V, cuál es el voltaje en el punto intermedio entre las resistencias, es decir, el valor de entrada A0 (V2 en la figura)?



Potenciómetro

Divisor resistivo

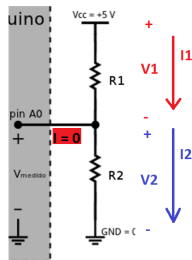


Supuestos:

- Denotemos por V la tensión de alimentación (5V), V_1 la caída en R_1 y V_2 la caída en R_2 (tensión de entrada en A0).
- La corriente por ambas resistencias es la misma (idealmente, el Arduino no consume corriente, sólo sensa la tensión).

Potenciómetro

Divisor resistivo



- Denotemos por V la tensión de alimentación (5V), V_1 la caída en R_1 y V_2 la caída en R_2 (tensión de entrada en A0).
- La corriente por ambas resistencias es la misma (idealmente, el arduino no consume corriente, sólo sensa la tensión).

$$V_1 = R_1 \cdot I \quad V_2 = R_2 \cdot I \quad V = V_1 + V_2 = (R_1 + R_2) \cdot I \quad \Rightarrow I = \frac{V}{R_1 + R_2}$$

$$V_1 = V \cdot \frac{R_1}{R_1 + R_2}$$

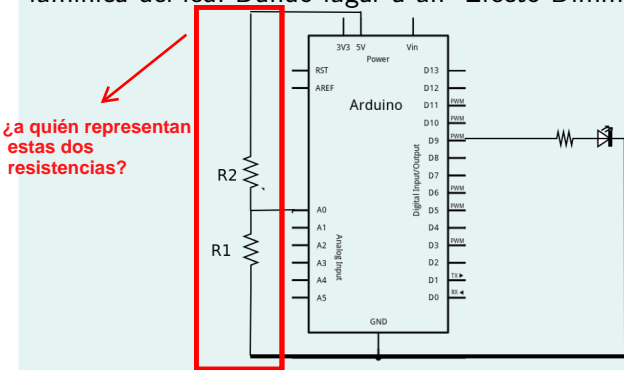
$$V_2 = V \cdot \frac{R_2}{R_1 + R_2}$$

Ejercicio clase

Ejercicio 5

Ejercicio - Bucles y manejo de entradas analógicas / salidas PWM

Armar un proyecto con el siguiente hardware y preparar un código para que al ir variando la posición del potenciómetro se vaya variando la intensidad lumínica del led. Dando lugar a un "Efecto Dimmer".



Made with Fritzing.org

Nota: repasar código de ejemplo presente en esta clase.

Esquema de la presentación

- 1 Recapitulando
- 2 Control de flujo
 - Ejercicio 1 - Clase
- 3 Entradas analógicas
 - Ejercicio 2 - Clase
- 4 Salidas con PWM
 - Ejercicio 3 - Clase
- 5 Materiales
 - Ejercicio 4 - Clase
 - Ejercicio 5 - Clase
- 6 EJERCICIOS para la próxima clase
 - Ejercicio 1
 - Ejercicio 2
 - Ejercicio 3

Ejercicio 1

Modificar el secuenciado de 3 leds del Taller 1 (Ejercicio 2 de deberes), de forma de poder controlar con un potenciómetro la frecuencia de la secuencia.

Al ir variando la perilla del potenciómetro los LEDs deben irse encendiendo en secuencia pero más o menos rápido (según hacia donde mueva la perilla)

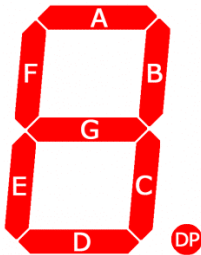
Ejercicio 2

- 1 Utilizando otro potenciómetro, agregar el efecto *dimmer* al secuenciador de 3 leds del Ejercicio 1 de deberes de este taller.
- 2 Utilizando una de las secuencias de bucle vistas, incrementar progresivamente y de manera automática el brillo de los leds mientras la entrada pin7 este en **HIGH**. En caso contrario disminuir el brillo en forma progresiva (misma cadencia que al aumentar) hasta el estado de inicio y mantenerlo fijo hasta que pin7 vuelva a estar en **HIGH**.

Ejercicio para hacer durante la próxima clase:

Ejercicio 3

Se trabajará con un display de 7 segmentos con punto, como el de las figuras.



<https://www.circuitbasics.com/>

Ejercicio para hacer durante la próxima clase:

Ejercicio 3

El objetivo general es, a partir del proyecto **"TR2021taller1Ej3TD"** brindado por los docentes a través de Tinkercad, completar el hardware y el software para preparar la realimentación visual de una cuenta regresiva de 9 segundos.

En particular, si un botón (a conectarse a un pin libre a elección) es pulsado, la cuenta regresiva deberá comenzar (desde el 9). Si el botón deja de pulsarse antes de terminar dicha cuenta regresiva (antes de llegar al 0), se deberá detener y dejar la cuenta lista para empezar nuevamente desde el 9. Quedando así preparada para que si se presiona el botón de nuevo, la cuenta regresiva empezará desde el principio nuevamente.

Ejercicio para hacer durante la próxima clase:

Ejercicio 3

Se pide:

- 1 Mapear el estado de cada uno de los segmentos a los pines para cada número (como ejemplo, ver primera línea del código brindado).
- 2 Interpretar las líneas de ejemplo dentro de la sentencia *for* presente en la función *void loop()*. En particular, investigar sobre el concepto de enmascarar, sobre la operación de corrimiento (\ll) y repasar el operador lógico AND (&). Se anima a hacer uso del monitor Serial para ver como van cambiando los bytes luego de cada operación.
- 3 Realizar un diagrama de flujo que represente la solución del objetivo general. Habrá una tarea en EVA para la entrega del mismo. Se recomienda la siguiente plataforma para la preparación de diagramas (<https://app.diagrams.net/> \rightarrow "FlowChart").
- 4 Comenzar a completar el código. Primero, completar lo básico para mostrar el 0. Luego, agregar otro número.
- 5 Agregar el botón al sistema.
- 6 Terminar de completar el código, basándose en el diagrama de flujo para alcanzar el objetivo general del ejercicio.

Resumen para la próxima clase:

- 1 Si no se terminaron los ejercicios para hacer en este taller, terminarlos.
- 2 Tener funcionando el Ejercicio 1 y el 2 de deberes para compartir en la siguiente clase. Leer la letra y comenzar con el Ejercicio 3 que terminaremos durante la próxima clase.
- 3 Haber leído y comprendido el Ejercicio 3 en el cual se trabajará durante la siguiente clase.
- 4 Por dudas utilizar el *Foro de consultas*. Recordar clase de consulta Jueves a la tarde.
- 5 Queda disponible un cuestionario sobre esta clase, que deberá ser completado en el sitio EVA. Lo deberá hacer cada estudiante individualmente!!
- 6 Se recomienda continuar con la lectura de la documentación sugerida en la sección de *Introducción* en el sitio de EVA. En particular, leer sobre funciones.