UNIVERSIDAD DE OVIEDO

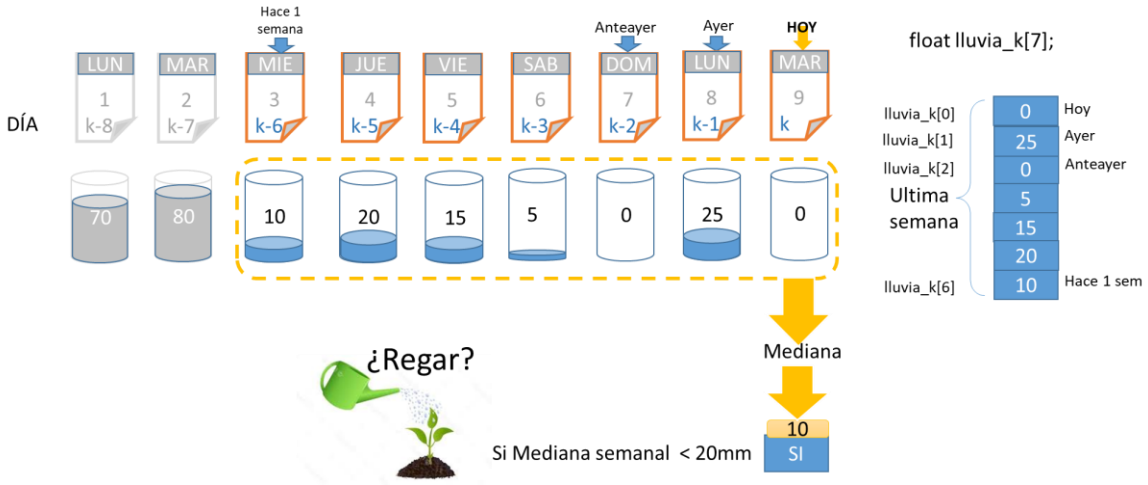
Departamento de Ingeniería Eléctrica,
Electrónica, de Computadores y Sistemas

Guía de Prácticas

ASIGNATURA:	Informática Industrial y Comunicaciones		
CENTRO:	Escuela Politécnica de Ingeniería de Gijón		
ESTUDIOS:	Grado en Ingeniería Electrónica y Automática		
CURSO:	3º	CUATRIMESTRE:	1
CARÁCTER:	Obligatoria	CRÉDITOS ECTS:	6
PROFESORADO:	Ignacio Alvarez, José Mª Enguita, Borja Millán		

PRACTICA 05: Manejo de tablas (arrays) y series temporales

1. Ejercicio a realizar: programa que obtenga una medida cada día de un sensor de lluvia acumulada (se simulará mediante entrada de teclado), y decida cada día si hay que realizar el riego en función del valor de la mediana de las 7 medidas más recientes (última semana).



El programa tendrá el estilo siguiente:

```
float lluvia_k[7];
float med_k;

InitTabla(lluvia_k,7);
while (1)
{
   DesplazarTabla(lluvia_k,7);
   lluvia_k[0]=LecturaSensor();  // realizar con scanf( )
   med_k=MedianaTabla(lluvia_k,7);
   UtilizarResultado(med_k);   // simplemente printf( )
   // Faltaría Sleep(Tm), pero no es necesario (se para en scanf)
}
```

2. Realizar las funciones InitTabla( ), DesplazaTabla( ) y PrintTabla( ), y comprobar que en cada pasada del bucle (instante de muestreo) se mantiene la tabla con los datos correctos.
3. Obtención de mediana: ordenar una tabla de menor a mayor, tomar el valor central. Para ordenar la tabla: buscar el índice donde se produce el valor mínimo, intercambiar el valor de ese índice con el 1er elemento de la tabla; repetir lo mismo para la tabla que empieza en el 2º elemento y tiene n-1 datos, etc.

Ejemplo: float t[5] = {1.0, 3.5, 2.4, 1.7, -3.8}; → mediana = 1.7

Ordenación de tabla (necesita tabla auxiliar para no modificar la original).

ind	tam		i_min	tabla_ordenada				
				t+0	t+1	t+2	t+3	t+4
0	5	Tabla desde t+ind y con tam elems →	**4**	1.0	3.5	2.4	1.7	-3.8
		Intercambia i_min y 1º para dicha tabla	→	**-3.8**	3.5	2.4	1.7	**1.0**
1	4	Tabla desde t+ind y con tam elems →	**3**	-3.8	3.5	2.4	1.7	1.0
		Intercambia i_min y 1º para dicha tabla	→	-3.8	**1.0**	2.4	1.7	**3.5**
2	3	Tabla desde t+ind y con tam elems →	**1**	-3.8	1.0	2.4	1.7	3.5
		Intercambia i_min y 1º para dicha tabla	→	-3.8	1.0	**1.7**	**2.4**	3.5
3	2	Tabla desde t+ind y con tam elems →	**0**	-3.8	1.0	1.7	2.4	3.5
		Intercambia i_min y 1º para dicha tabla	→	-3.8	1.0	1.7	**2.4**	3.5
4	1	Tabla desde t+ind y con tam elems →	**0**	-3.8	1.0	1.7	2.4	3.5
		Intercambia i_min y 1º para dicha tabla	→	-3.8	1.0	1.7	2.4	**3.5**

Tras finalizar la ordenación, la mediana es el valor que se queda en t[2] = 1.7

4. Añadir el uso de un contador para evitar cálculos incorrectos antes de disponer de suficientes datos:

```
float lluvia_k[7];
float med_k;
int counter=0;

InitTabla(lluvia_k,7);
while (1)
{
   DesplazarTabla(lluvia_k,7);
   lluvia_k[0]=LecturaSensor();  // realizar con scanf( )
   counter++;
   if (counter>=7)
   {
      counter=7;
      med_k=MedianaTabla(lluvia_k,7);
      UtilizarResultado(med_k);          //   simplemente   printf(   )
   }
   // Faltaría Sleep(Tm), pero no es necesario (se para en scanf)
}
```

RECORDATORIO:

Manejo de datos de series temporales: (diapositivas 49 y 56 de
http://isa.uniovi.es/~ialvarez/Curso/descargas/Programacion%20de%20Control%20(Bases).pdf

Ejemplo de riego:
http://isa.uniovi.es/~ialvarez/Curso/descargas/series_temporales_regando.pdf

AMPLIACIONES PROPUESTAS:

1. Solicitar por teclado al principio del programa el número de datos "a recordar", para que no sean 7 de forma estática, y utilizar asignación dinámica de memoria para la tabla.
2. Añadir asignación dinámica de memoria para la tabla copia que se necesita crear en la función Mediana( ): malloc( ) y free( )
3. Añadir el algoritmo de decisión siguiente:

   Si la mediana de lluvia de la última semana es menor que 10 y no se ha regado ninguno los dos días anteriores → regar

   En caso contrario (mediana mayor o igual que 10, o bien se ha regado alguno de los dos días anteriores) → no regar.

Se requiere una segunda tabla para los resultados de riego, que se trata de forma similar a la de lluvia:

```
float lluvia_k[7];  // Se necesita 1 semana según el algoritmo
float riego_k[3];   // Se necesitan 3 días (actual y 2 anteriores)
...

InitTabla(lluvia_k,7);
InitTabla(riego_k,3);
while (1)
{
   DesplazarTabla(lluvia_k,7);
   DesplazarTabla(riego_k,3);
   lluvia_k[0]=LecturaSensor();  // realizar con scanf( )
   counter++;
   if (counter>=7)
   {
      counter=7;
      med_k=MedianaTabla(lluvia_k,7);
      riego_k[0]=CalcularRiego(med_k,riego_k[1],riego_k[2]);
      printf estado de riego_k[0]
   }
   // Faltaría Sleep(Tm), pero no es necesario (se para en scanf)
}
```

4. Solicitar por teclado al principio del programa el tipo de cálculo a realizar, que puede ser uno de los caracteres: 'm' (mediana), '>' (máximo), '<' (mínimo). Aplicar el cálculo indicado en lugar de la mediana.
5. Mejorar el algoritmo de cálculo de la mediana mediante la estrategia de división descrita en http://sarielhp.org/research/CG/applets/linear_prog/median.html. Se requiere usar la función de C rand( ), que devuelve un entero aleatorio entre 0 y RAND_MAX (para obtener un entero aleatorio entre 0 y n, usar rand( ) % (n+1)).