1) Una empresa de electro-medicina ha conformado un equipo de trabajo a fin de desarrollar un conjunto de nuevos productos. En este caso, se busca desarrollar un equipo capaz de medir diferentes parámetros asociados a la actividad eléctrica del corazón.

Como parte del equipo de desarrollo, se le ha encargado la realización del firmware del primer prototipo el cual debe cumplir con los siguientes requisitos:

- Modo de Inicio/detención del funcionamiento a través de un pulsador
- Medición del ritmo cardíaco
- Actualización y visualización del mismo en un conjunto de 3 displays de 7 segmentos (cátodo común).
- Envío del parámetro calculado mediante comunicación serie cada 3 segundos.

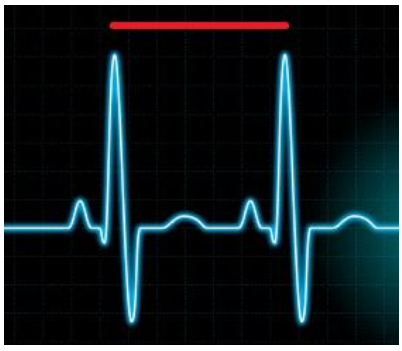
Dentro del grupo de trabajo ya se han desarrollado algunos módulos que resuelven la adquisición de la señal y han encapsulado los mismos a través de un buffer que almacena la información y un grupo de primitivas.

```
struct datos
{
        uint32_t   valor; // es el valor de la muestra
        uint32_t   tiempo; // es el tiempo de la muestra expresado en milisegundos
}

//Ejemplo de declaración para el buffer de datos
struct datos senal[N] ; // siendo N el numero de muestras

//prototipo de la primitiva para la adquisición
uint8_t ObtenerSenal(struct datos *) ;
```

Al invocar la primitiva, la misma devuelve 0 en caso de haberse llenado el buffer con todos los datos y 1 en caso de no estar aún disponible el buffer para su uso.

La distancia entre dos picos (dos señales) resultará proporcional al ritmo cardíaco. De forma de invocar a la función ObtenerRitmo se debe encontrar el máximo de dos señales consecutivas y pasar como argumento los tiempos de muestra correspondiente.

```
//prototipo de la primitiva para el cálculo del ritmo cardíaco
void ObtenerRitmo(uint32_t, uint32_t) ;
```



Cuando se disponga de información válida se deberá actualizar la información que se muestra en el display. En este mismo caso, cada 3 segundos se deberá enviar por comunicación serie la siguiente trama.

Inicio (1 byte)	Información (1 byte)	Finalización (1 byte)
$	Valor calculado	#

Al iniciar el sistema mediante un pulsador, el display muestra el valor 0 y luego a partir de las mediciones se irá actualizando la información. Al volver a presionar el pulsador debe dejar en el display la última medición y detener el envío hacia la PC. En caso de una nueva presión el sistema vuelve a comenzar. El pulsador ya fue configurado como interrupción externa (EINT3) por flanco descendente.

2) Como agregado de nuevas funciones, se desea medir en forma períodica (500 ms) la temperatura del paciente. Como en el caso anterior, el mismo grupo ha desarrollado una primitiva la cual devuelve el valor de temperatura.

   uint8_t ObtenerTemperatura (void) ;

A su vez, se desea realizar una modificación en la función de trama realizada en el punto 1 para incorporar la nueva información.

Inicio (1 byte)	**Información (1 byte)**	**Información (1 byte)**	**Finalización (1 byte)**
$	Valor ritmo	Valor temperatura	#

Se pide:

- Máquina de estado del sistema
- Realización de la función *main* en donde quede claramente incluida la lógica general de funcionamiento del sistema.
- La rutina de atención para el inicio/detención del sistema (EINT3).
- La/s rutinas para la visualización de los datos
- La rutina de atención para la comunicación serie
- La/las función/es necesaria/s para el armado de la/s trama/s y envío de datos
- Las rutinas para gestionar la temperatura en forma periódica.
- La nueva función de armado de trama (incluye la temperatura)

Condición de aprobación: ejercicio 1 completo sin errores conceptuales

Importante:

- Si Ud. considera que necesita una base de tiempo, puede usar el Systick sabiendo que **YA ha sido configurado** en 1ms.
- Ya se han realizado todas las inicializaciones correspondientes a GPIO.
- Puede utilizar las primitivas getPin() y setPin()