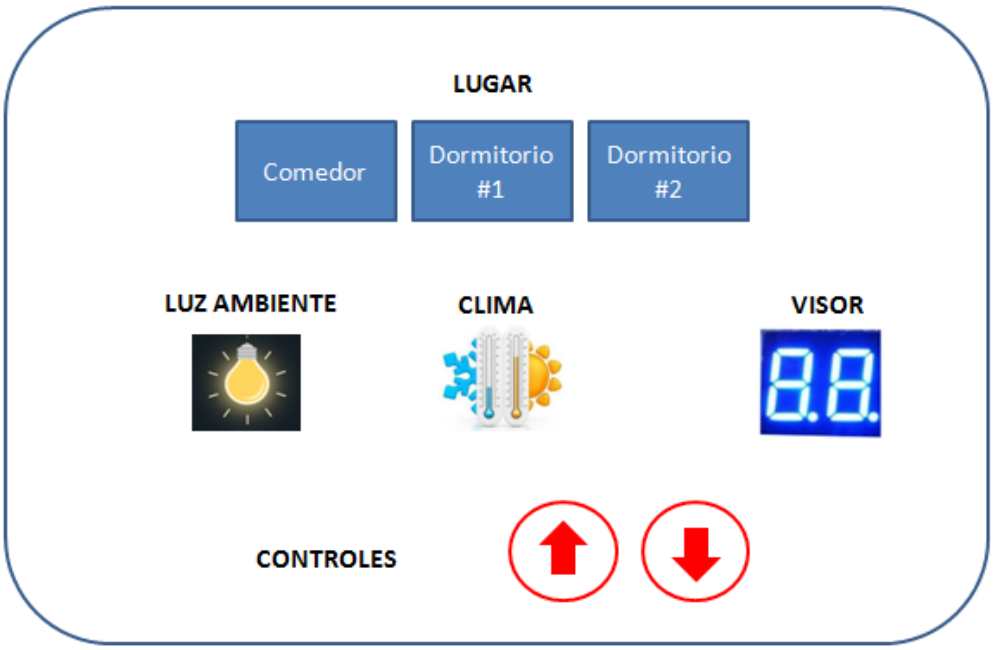
Final INFORMATICA II	Total Hojas	Duración	Fecha	Diciembre 2020
		17.00 a 19.30hs	# 3	

Nombre y Apellido del alumno	Nº Legajo	Calificación	Docente a cargo / Firma	

Una empresa dedicada a la automatización se encuentra desarrollando paneles de control del hogar. En esta etapa, lo convocan para desarrollar un panel para el control de temperatura y nivel de luz de viviendas de 4 ambientes. El objetivo del sistema es poder controlar la luz ambiente y el clima dentro del comedor y los dos dormitorios.



El sistema a desarrollar consta de:

- Panel de control con:
  - Tecla para seleccionar el comedor (T_COM)
  - Tecla para seleccionar el dormitorio #1 (T_DOM1)
  - Tecla para seleccionar el dormitorio #2 (T_DOM2)
  - Tecla de control de luz (T_LUZ)
  - Tecla de control de temperatura de la vivienda (T_CLIMA)
  - Tecla de control de aumento (T_MAS)
  - Tecla de control de reducción (T_MENOS)
- Visor compuesto por 2 display de 7 segmentos (ánodo común).
- Módulo de comunicación con servidor (mediante comunicación serie).
- Módulo para el control de temperatura
- Módulo para el control de luz

Forma de uso:

- Se toca el botón del lugar / ambiente.
- A continuación se selecciona la variable que se desea controlar (luz o clima)
- Por último, se selecciona la acción (subir o bajar).
  - En el caso de temperatura se muestra la temperatura (valor entero). En el caso de la intensidad de luz se muestra el valor 1, 2, 3 o 4 siendo los correspondientes a la graduación de

luz. Posterior al valor máximo si se presiona la flecha de subir no tendrá ningún efecto. Lo mismo sucederá con el límite inferior.
    - En el caso del clima, se mostrará la temperatura en un rango de 16 a 30 grados. Una vez alcanzados los límites sucederá la misma situación que frente al control de luz.
- Cualquier otra combinación de teclas debe descartarse

Por cada operación exitosa, es decir, una secuencia correcta de uso, se enviará una trama serie con el siguiente formato:

Inicio 1 byte	Lugar 0 1 o 2	Control LUZ – CLIMA	Accion SUBIR - BAJAR	Finalización 1 byte
$	1 byte	1 byte	1 byte	#

Cada 5 minutos se deberá medir la temperatura y si se encuentra en el valor indicado +/- 1 grado no deberá realizar acción. En caso de estar por debajo encenderá la calefacción mediante la primitiva **EncenderCalefaccion**. En el caso de estar por arriba del rango deberá invocar a la primitiva **EncenderRefrigeracion**. En el caso de estar en el rango de control apagará la calefacción o la refrigeración utilizando las primitivas equivalente (ApagarCalefaccion, ApagarRefrigeracion).

Se pide realizar:

- Diagramas de estado
- Implementación de las máquina/s de estado del sistema en lenguaje C
- Realización de la función *main* en donde quede claramente incluida la lógica general de funcionamiento del sistema.
- La/s rutinas para la visualización de los datos y manejo del display
- La/s rutinas para el control del teclado de 2 filas y 4 columnas.
- La rutina de atención para la comunicación serie
- La/las función/es necesaria/s para el envío de datos por puerto serie (UART0)
- La función de atención del ADC – utilizando el canal 5.

Importante:

- Si Ud. considera que necesita una base de tiempo, puede usar el Systick sabiendo que **YA ha sido configurado** en 10 ms.
- Ya se han realizado todas las inicializaciones, por lo tanto solo se debe invocar a la función InicializarHW().
- Puede utilizar la primitiva **ModificarLuz**, la cual recibe el ambiente a controlar y el nivel de intensidad.
- Puede utilizar la primitiva **ModificarTemperatura**, la cual recibe el ambiente y un valor entero correspondiente a la temperatura a ajustar.
- Puede utilizar las primitivas getPin() y setPin()
- Para el control del display puede elegir la configuración de pines en función de sus necesidades
- Para el control del teclado puede asignar los pines en función de su necesidad, como notará habrá teclas no asignadas, las cuales están presentes en el pinout pero no en el panel.
- El rango de temperatura del sensor es de 0 a 50 grados siendo los 50 grados correspondiente a la cantidad de cuentas máxima del ADC. Para el uso del adc debe entenderse un comportamiento lineal del sensor.

Teclado 7 teclas → 3x3.

Encender_calefacción()

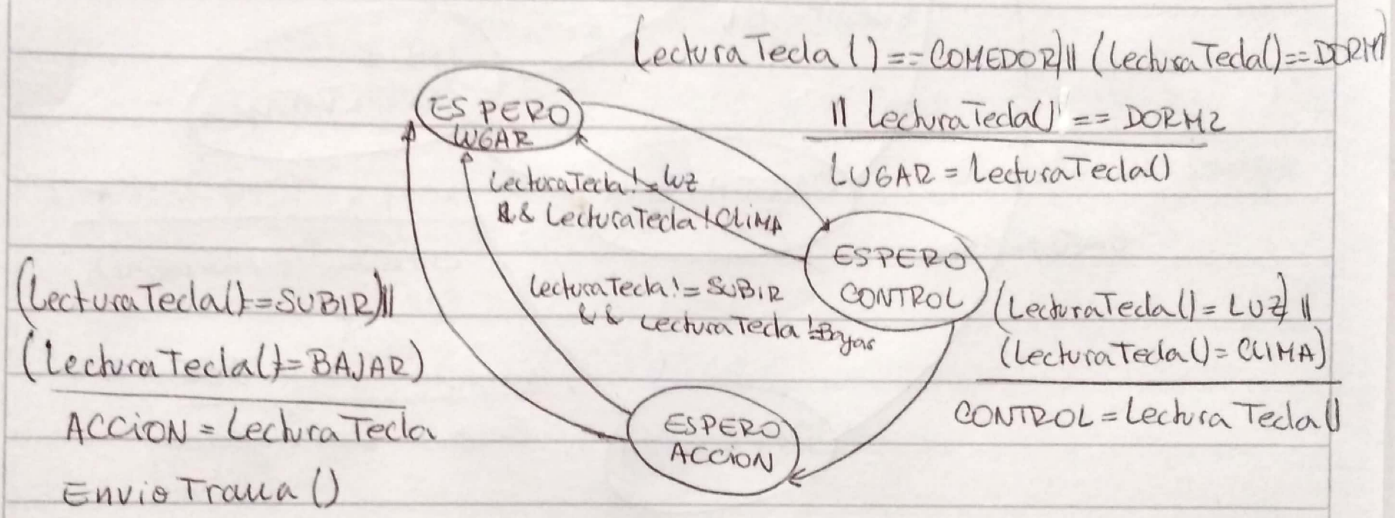
Encender Refrigeración()

Apagar Calefacción()

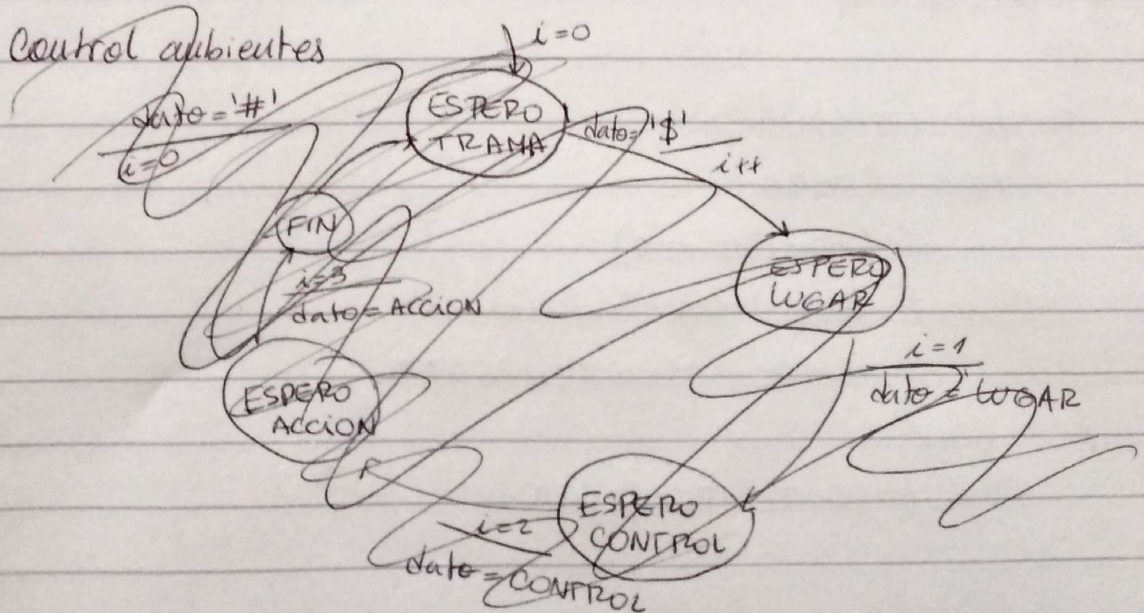
Apagar refrigeración()

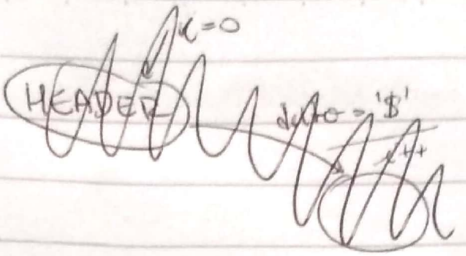
ModificarLuz(int aubiente, int ~~temperatura~~ intensidad)

ModificarTemp(int aubiente, int temperatura)

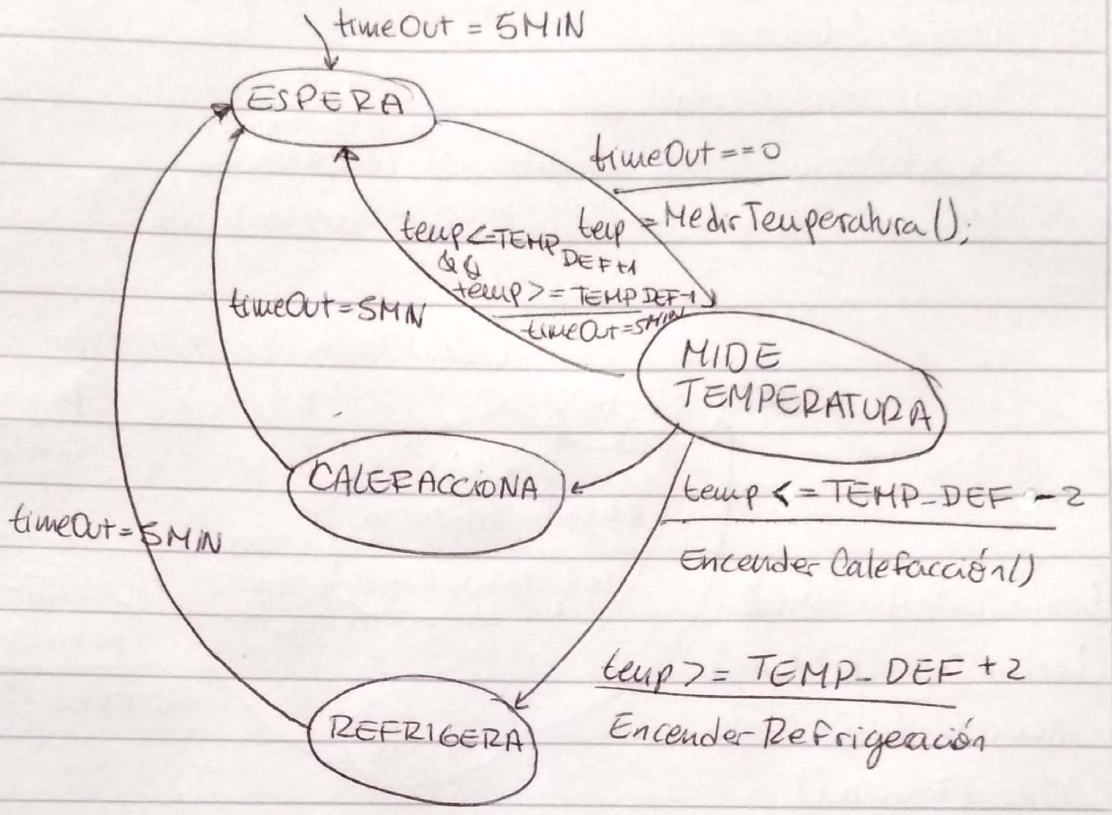


Si no sigue la secuencia requerida, vuelve al estado inicial: ESPERO LUGAR





CONTROL Temperatura




```
void Control Temperatura (void)
{   uint8_t estado = ESPERA;
    uint8_t temp;

    switch (estado){
        case ESPERA:
            if (timeOut==0)
            { temp = MedirTemperatura();
              estado = MIDE_TEMPERATURA;
            }
        break;
        Case MIDE_TEMPERATURA:
```

NOTA

```
if (temp <= (TEMP_DEF - 2))   // Temp demasiado baja
{ EncenderCalefaccion();
   timeOut = 5MIN;
   estado = ESPERA;
}
if (temp >= (TEMP_DEF + 2))   // Temp demasiado Alta
{  EncenderRefrigeracion();
   timeOut = 5MIN;
   estado = ESPERA;
if (temp <= (TEM_DEF + 1) && temp >= (TEMP_DEF - 1))
{  ApagarCalefaccion();              // Temp en rango
   ApagarRefrigeracion();
   timeOut = 5MIN;
   estado = ESPERA;
}
break;
}

}
```

```
void MdEstadoPanel (void) {
    static uint8_t estado = ESPERO_LUGAR;

    switch (estado) {
        case ESPERO_LUGAR:
            key = LecturaTecla();
            if(key == COMEDOR || key == DORM1 || key == DORM2)
            { LUGAR = key;
              estado = ESPERO_CONTROL;
            }

            break;
        case ESPERO_CONTROL:
            key = LecturaTecla();
            if(key == LUZ || key == CLIMA)
            { CONTROL = key;
              estado = ESPERO_ACCION;
            }
            else
            { estado = ESPERO_LUGAR}
            break;
        case ESPERO_ACCION:
            key = LecturaTecla();
            if(key == SUBIR || key == BAJAR)
            { ACCION = key;
              estado = ESPERO_LUGAR
              EnvioTrama()
            }
            else
              estado = ESPERO_LUGAR;
            break;
    }
}
```

→ if(key==LUZ) Display Show(intensidad);
if(key==CLIMA) Display Show(TEMP_DEF)

// Lectura Teclas 2 filas 4 columnas

COM	D1	D2		fila 1
Luz	Cima	sub	bajar	fila 2
col 1	col 2	col 3	col 4	

```
uint8_t LecturaTecla (void)
{
  setPIN(COL1,0);
  SetPIN(col2,1);
  setPIN(col3,1);
  setPIN(col4,1);
  if (!GetPIN(FILA1)) return COMEDOR
  if (!GetPIN(FILA2)) return LUZ;

  setPIN(col1,1);
  setPIN(col2,0);
  setPIN(col3,1);
  setPIN(col4,1);
  if(!GetPIN(FILA1)) return DORM1;
  if(!GetPIN(FILA2)) return CLIMA;

  setPIN(col1,1);
  setPIN(col2,1);
  setPIN(col3,0);
  setPIN(col4,1);
  if(!GetPIN(FILA1)) return DORM2;
  if(!GetPIN(FILA2)) return SUBIR;

  setPIN(col1,1);
  setPIN(col2,1);
  setPIN(col3,1);
  setPIN(col4,0);
  if(!GetPIN(FILA1)) return NO_KEY;
  if(!GetPIN(FILA2)) return BAJAR;
} return NO_KEY
```

NOTA

Para teclado también sería conveniente implementar una función Antirrebote.

Para display 7 segmentos asumo que tengo definida una tabla: uint32_t tabla7seg[] = { ?
donde para cada valor que quiero mostrar están definidos los segmentos a encender.

```
// Driver
void BarridoDisplay(void) // Función llamada en Systick

{ static uint8_t digito = 0; // display de 2 dígitos
  uint8_t dato;

  // Apagar display
  setPIN(DIGITO0, OFF);
  setPIN(DIGITO1, OFF);

  // Convierto el valor a 7 segmentos
  dato = tabla7seg[gdigit_buffer[digito]];

  // Pongo el dato en el bus de datos
  setPIN(segA, dato & 0x01);
  setPIN(segB, (dato << 1) & 0x01);
  setPIN(segC, (dato << 2) & 0x01);
         segD
         segE
         segF
  setPIN(segG, (dato << 6) & 0x01);

  // Enciendo el display correspondiente
  switch(digito){
    case 0
      setPIN(DIGITO0, ON);
```

```
        case 4 :
            setPIN(DIGITO4, ON);
        }
        // Incremento el indice del display
        digito++;

    }
// Primitiva        void DisplayShow (uint16_t Valor)
                    { for (i= Displays-1; i>=0; i--)
                        { gdigitbuffer[i] = Valor%10;
                          valor/=10;
                        }
                    }


//Rutina atención Serie
    UART0_Handler (void)
    {   uint8_t iir, dato;
        do { iir = U0IIR;
             if (iir & 0x04) // Hay dato disponible
              { dato = U0RBR // registro del que llega la
                PushRx (dato);          interrupción
              }
            while (!(iir & 0x01));
        {
```

// Funciones para envio de datos por puerto serie.

Transmisión: PushTx → Buffer → PopTx

Recepción: PopRx ← Buffer ← PushRx

```
void main (void) {
    Inicializar HW

    while (1) {
    MdEstadoPanel();
    ControlTemperatura();
     }

}
```

```
void PushTx (uint8_t dato)
 { BufferTx[iTxIn] = dato
    iTxIn ++;
    iTxIn %= SIZE-BUFFER;
  if (TxStart == 0)
   { TxStart = 1
    U0THR = BufferTx[iTxOut];
   }
}
```

```
uint8t PopTx (void)
 { int aux = -1;
   if (iTxIn != iTxOut)
    { aux = BufferTx [iTxOut];
      iTxOut ++;
      iTxOut %= SIZE_BUFFER;
    }
  return aux
}
```

NOTA

```
void Envio Trava (void)
{   uint8_t i;

    buffer [0] = '$';
    buffer [1] = LUGAR;
    buffer [2] = CONTROL;
    buffer [3] = ACCION;
    buffer [4] = '#';

    for (i=0; buffer [i]; i++)
        PushTx (buffer [i]);

}
```

Estudiante:
Corrigió: Nahuel Gonzalez
Nota: 7 (siete)

- **Diagramas de estado**
  - Realiza 2 diagramas de estado.
  - El control de teclas y validación está realizado adecuadamente
    - El diagrama de secuencia no tiene estado inicial
  - El diagrama de control de temperatura no contempla lo que sucede dentro de los 5 minutos.
  - Falta un diagrama de estado que controle la luz

- **Implementación de las máquina/s de estado del sistema en lenguaje C**
  - Implementa las 2 máquinas de estado
  - En la maquina del panel: No se observa donde se modifica la variable **intensidad** y entiendo que debería modificarse en el siguiente estado correspondiente a la modificación de la tecla de subir o bajar.
  - En la maquina de control de clima la implementación es correcta.
  - Falta una máquina que controle la luz
  - No realiza temporización

- **Realización de la función *main* en donde quede claramente incluida la lógica general de funcionamiento del sistema.**
  - Realiza la función main en forma correcta llamando a la inicialización y a las máquinas de estado.

- **La/s rutinas para la visualización de los datos y manejo del display**
  - El desplazamiento para el control de la información es sentido inverso. El resto está ok.
  - Implementa la primitiva en forma correcta.

- **La/s rutinas para el control del teclado de 2 filas y 4 columnas.**
  - Realiza correctamente la función de barrido del teclado
  - Menciona que sería bueno una función de antirebote pero no lo implementa.

- **La rutina de atención para la comunicación serie**
  - Implementa la ISR en forma adecuada
  - Implementa las funciones de push y pop para transmisión

- **La/las función/es necesaria/s para el envío de datos por puerto serie (UART0)**
  - Desarrolla una función de envío llamando a los módulos previamente creados

- **La función de atención del ADC – utilizando el canal 5.**
  - No resuelve los módulos de ADC.