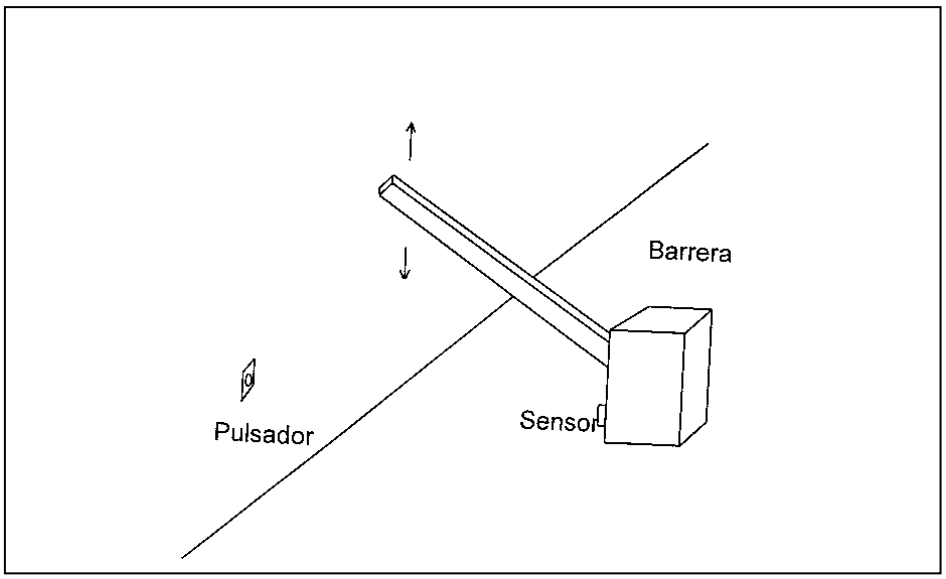
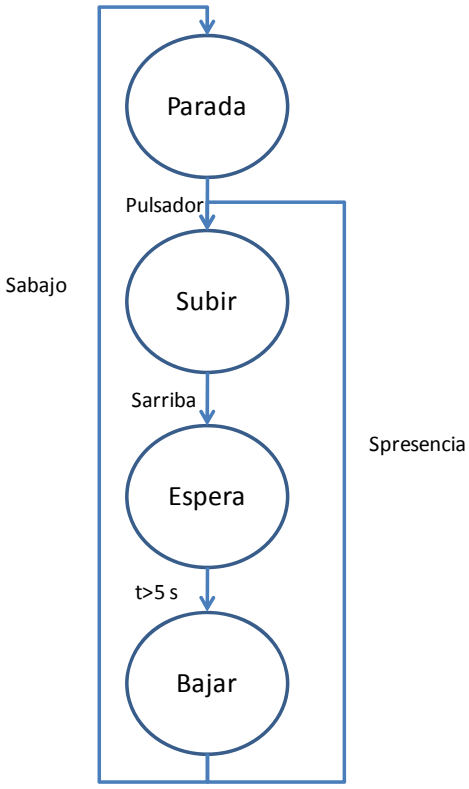
PROBLEMAS RESUELTOS EN LENGUAJE LITERAL ESTRUCTURADO (ST)

Problema: "Barrera de Garaje"

Se pretende automatizar el siguiente funcionamiento:

Al oprimir el pulsador, la barrera sube. Cuando llega arriba permanecerá subida durante 20 segundos, transcurrido ese tiempo inicia el descenso hacia su posición inicial. Si durante el proceso de bajada de la barrera se detecta la presencia de algún vehículo por medio del sensor, la barrera vuelve a subir, espera a que el vehículo se retire y después baja.





SOLUCIÓN:

Entradas		**Salidas**		**Estados**	
Psubida	%I1.1	Motor subir	%Q2.0	Parada	%M0
Sarriba	%I1.13	Motor bajar	%Q2.1	Subir	%M1
Sabajo	%I1.8			Espera	%M2
Spresencia	%I1.2			Bajar	%M3

Configuración del Temporizador:

%TM0.P:=200;

%TM0.B:=100ms;

```
! (* Inicialización de los estados *)
  %L0:
  IF %S13 THEN
  SET %M0;
  RESET %M1;
  RESET %M2;
  RESET %M3;
  RESET %M4;
  END_IF;
```

- **La sentencia %L0 es una inicialización de los estados, en este caso hemos utilizado el bit de sistema %S13, este bit (según el manual del autómata) toma valor 1 en el primer ciclo después de la puesta a RUN, es decir, vale 1 cuando se enciende el autómata. Por eso nos puede servir para determinar la condición de inicio del programa.**

  Las instrucciones indican que si estamos en el primer ciclo, se acaba de pasar a RUN el autómata (IF %S13 THEN), ponemos a 1 el bit de memoria que indica el primer estado al que hemos denominado "Parada" (SET %M0) y a 0 el resto de los estados (RESET %M1; RESET %M2).

  Indicar que todas las instrucciones deben finalizarse con ";".

  Al finalizar la estructura de control "END_IF".

```
! (* La barrera sube cada vez que se oprime un pulsador *)
  %L1:
  IF %I1.1 AND %M0 THEN
  SET %M1;
  RESET %M0;
  END_IF;
```

- **La sentencia %L1 indica la transición del estado "Parada" (%M0) al estado " Subir" (%M1).**
  - **Cuando está en el estado de "Parada" y se oprime el pulsador "Psubida" (IF %I1.1 AND %M1 THEN), entonces se pone a 1 el bit de memoria que indica el estado "Subir" (SET %M1) y se pone a 0 el bit de memoria que indica el estado "Parada" (RESET %M0).**

```
! (* Cuando llega al final de su recorrido se para durante 5 segundos *)
  %L2:
  IF %I1.13 AND %M1 THEN
  SET %M2;
  RESET %M1;
  END_IF;
```

- **La sentencia %L2 indica la transición del estado "Subir" al estado "Espera", al llegar al final del recorrido, lo detecta el sensor "Sarriba", la barrera debe parar.**
    - **Cuando está en el estado de "Subir" y el sensor "Sarriba" detecta que ha llegado arriba (IF %I1.13 AND %M1) , entonces se pone a 1 el bit de memoria que indica el estado "Espera" (SET %M2) y se pone a 0 el bit de memoria que indica el estado "Subir" (RESET %M1).**

```
! (* Inicialización del temporizador *)
  IF RE %M2 THEN
  START %TM0;
  ELSIF FE %M2 THEN
  DOWN %TM0;
  END_IF;
```

- **La sentencia %L5 gestiona el temporizador, que se inicia con el flanco de subida del bit que indica el estado "Espera" (RE %M2), y se desactiva al salir del estado (FE %M2).**

```
! (* Detención de la barrera durante 5 segundos *)
  %L21:
  IF %M2 AND %TM0.Q AND(NOT %I1.2)THEN
  SET %M3;
  RESET %M2;
  END_IF;
```

- **La sentencia %L21 indica la transición del estado "Espera" al estado "Bajar", para lo cual deben haber transcurrido 20 segundos. Según el enunciado la barrera debe permanecer arriba 5 segundos parada, para esto utilizaremos un temporizador. Cuando el temporizador haya terminado de medir los 20 segundos, su salida se pondrá a 1 (%TM0.Q). Además se comprueba que no hay ningún coche debajo de la barrera, por medio del sensor de presencia "Spresencia" antes de iniciar su bajada.**
    - **Cuando está en el estado "Espera" (%M2) y han transcurrido los 5 segundos (%TM0.Q) y no hay coche debajo de la barrera (NOT %I1.2), todas las condiciones a la vez, es decir , con operación AND**

      IF %M2 AND %TM0.Q AND (NOT %I1.2)
      entonces se pone a 1 el bit de memoria que indica el estado "Bajar" (SET %M3) y se pone a 0 el bit de memoria que indica el estado "Espera" (RESET %M2).

```
! (* Cuando llega abajo, la barrera se para *)
  %L3:
  IF %M3 AND %I1.8 THEN
  SET %M0;
  RESET %M3;
  END IF;
```

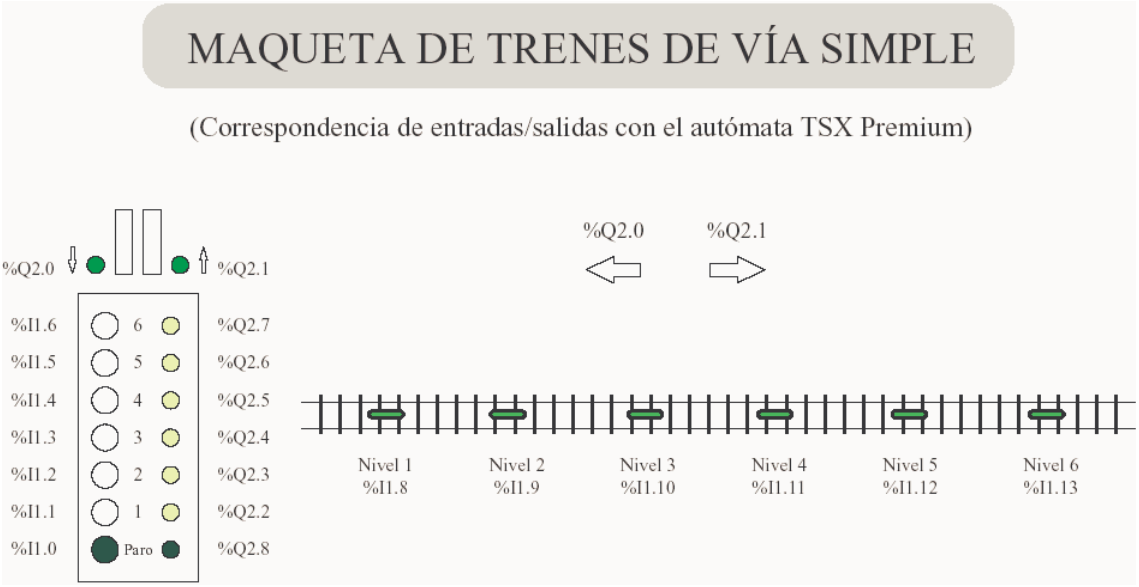
- **La sentencia %L3 indica la transición del estado "Bajar" al estado "Parada", al llegar al final del recorrido, lo detecta el sensor "Sabajo", la barrera debe parar.**
  - **Cuando está en el estado de "Bajar" y el sensor "Sabajo" detecta que ha llegado abajo (IF %I1.8 AND %M3) , entonces se pone a 1 el bit de memoria que indica el estado "Parada" (SET %M0) y se pone a 0 el bit de memoria que indica el estado "Bajar" (RESET %M3).**

```
! (* En el caso de que un vehículo este atravesando, la barrera permanecerá subida
     y si estaba bajando subirá *)
  %L4:
  IF %M3 AND %I1.2 THEN
  SET %M1;
  RESET %M3;
  END_IF;
```

- **La sentencia %L4 indica la transición del estado "Bajar" al estado "Subir" que se puede producir si el sensor de presencia detecta un vehículo atravesando la barrera cuando esta está bajando**
  - **Cuando está en el estado de "Bajar" y el sensor "Spresencia" detecta que hay un vehículo debajo (IF %M3 AND %I1.2) , entonces se pone a 1 el bit de memoria que indica el estado "Subir" (SET %M1) y se pone a 0 el bit de memoria que indica el estado "Bajar" (RESET %M3).**

```
! (* Asigno las salidas correspondientes a los estados *)
  %Q2.0:=%M3;
  %Q2.1:=%M1;
```

- **La sentencia %L6 asigna las salidas a los estados correspondientes:**
  - **%Q2.0:=%M3; La salida %Q2.0 (motor de bajada) solo está activada cuando %M3 toma valor 1, es decir, cuando está activado el estado "Bajar"**
  - **%Q2.1:=%M1; La salida %Q2.1 (motor de subida) solo está activada cuando %M1 toma valor 1, es decir, cuando está activado el estado "Subir"**



Se quiere que la vagoneta efectúe el siguiente proceso cada vez que pulsemos el botón de arranque:

Debe cargar material en el área de carga (nivel 1) en un proceso que dura 5 segundos. Tras cargar, dirigirse al puesto de trabajo situado en el nivel 2 y efectuar la descarga del materia, este proceso dura 4 segundos. Una vez concluido el proceso de descarga volver a situarse en el área de carga. Repetir el mismo proceso para los puestos de trabajo situados en los niveles 3, 4 y 5.

Durante la etapa de carga debe mantenerse encendido el indicador luminoso correspondiente al área de carga, de igual forma durante la descarga se mantendrá encendida la luz correspondiente al puesto de trabajo donde se efectúa la descarga.

Notas:

Se pide programar en lenguaje literal estructurado.

Utilizar una palabra interna para indicar en cada instante el puesto de trabajo sobre el que se está actuando.

SOLUCIÓN

Programa MAIN

```
!(* Inicializacion *)
IF(NOT %M0 AND NOT %M1 AND NOT %M2 AND NOT %M3 AND NOT %M4)THEN
SET %M0;
END_IF;
!(* Puesta en marcha *)
IF %M0 AND %I1.0 THEN
SET %M10;
%MW0:=2;
END_IF;
!(* Bucle para abastecer todos los niveles *)
IF %MW0<=5 THEN
        (* inicio del proceso de carga *)
        IF %M0 AND %M10 THEN
        SET %M1;
        RESET %M0;
        END_IF;
        (* proceso de carga *)
        IF RE %M1 THEN
        START %TM0;
        ELSIF FE %M1 THEN
        DOWN %TM0;
        END_IF;
```

```
(* Paso al estado de movimiento hacia derecha *)
IF %TM0.Q THEN
SET %M2;
RESET %M1;
END_IF;
(* Elijo hasta donde moverme *)
IF(%MW0=2)THEN
        IF %M2 AND %I1.9 THEN
                SET %M3;
                RESET %M2;
        END_IF;
ELSIF(%MW0=3)THEN
        IF %M2 AND %I1.10 THEN
                SET %M3;
                RESET %M2;
        END_IF;
ELSIF(%MW0=4)THEN
        IF %M2 AND %I1.11 THEN
                SET %M3;
                RESET %M2;
        END_IF;
ELSE
        IF %M2 AND %I1.12 THEN
                SET %M3;
                RESET %M2;
        END_IF;
END_IF;
```

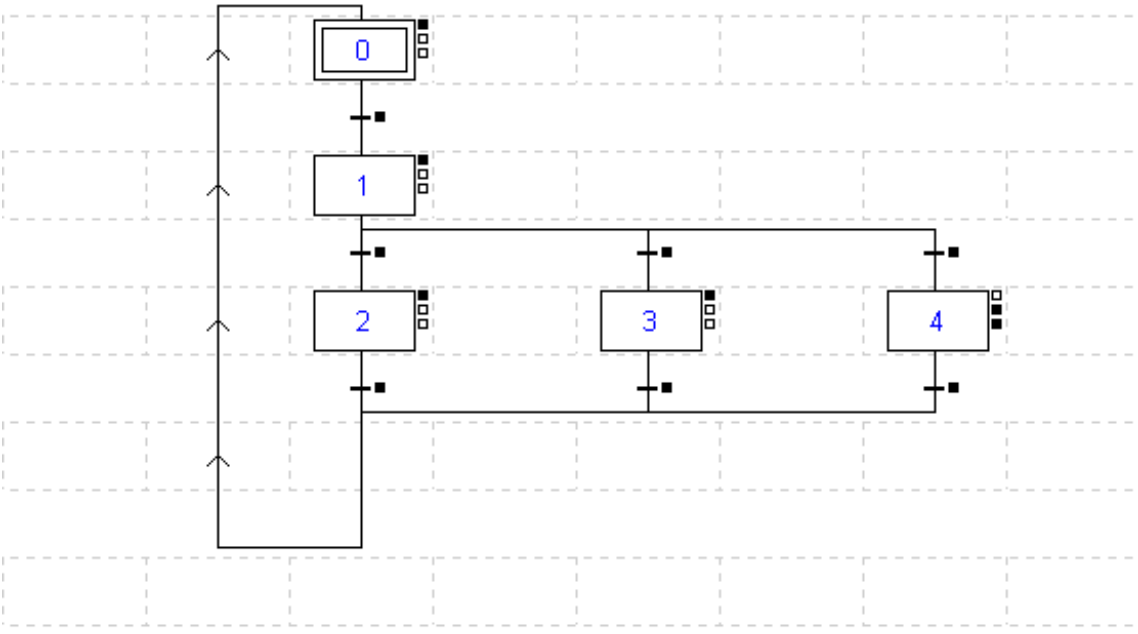
```
        (* Etapa de descarga *)
        IF RE %M3 THEN
        START %TM1;
        ELSIF FE %M3 THEN
        DOWN %TM1;
        END_IF;
        (* Vuelta a zona de carga *)
        IF %M3 AND %TM1.Q THEN
        SET %M4;
        RESET %M3;
        END_IF;
        (* Acciones a realizar al llegar a la zona de carga *)
        IF %M4 AND %I1.8 THEN
        SET %M1;
        RESET %M4;
        %MW0:=%MW0+1;
        END_IF;
END_IF;
!(* Activacion de las salidas *)
%Q2.1:=%M2;
%Q2.0:=%M4;
!(* Vuelta al estado de reposo *)
IF %MW0>5 THEN
SET %M0;
RESET %M1;
RESET %M10;
END_IF;
```

PROBLEMA

Desarrollar un programa utilizando GRAFCET y lenguaje literal estructurado que realice las siguientes operaciones:

- Una vez pulsada la entrada %I1.0, lea el valor de las entradas y las guarde en una palabra (%MW10) y cuente cuántas están activas en %MW4.
- Si se pulsa el interruptor %I1.1, debe activar el número de salidas indicadas en %MW4, empezando por la primera.
- Si se pulsa el interruptor %I1.2, debe activar el número de salidas indicadas en %MW4, empezando por la última.
- Si se pulsa %I1.3, debe activar las salidas almacenadas en %MW10 hasta que se pulse el interruptor %I1.4.
- Una vez actualizadas las salidas, se vuelve a esperar la entrada %I1.0.

Esquema de GRAFCET



Programación de las transiciones:

X0 -> X1 : %I1.0

X1 -> X2 : %I1.1

X1 -> X3 : %I1.2

X1 -> X4 : %I1.3

X2 -> X0 : TRUE

X3 -> X0 : TRUE

X4 -> X0 : %I1.4

Programación de las etapas

Etapa 0

```
!%MW4:=0; (* Inicializamos el contador de entradas activadas *)
```

Etapa 1

```
! %Q2.0:16:=0;  (* Ponemos a cero las salidas *)
%MW10:=%I1.0:16; (* Copiamos la tabla de entradas en la palabra MW10 *)
(* Contamos entradas activas *)
!FOR %MW0:=0 TO 15 DO                (* Bucle para recorrer todas las entradas *)
        IF %I1.0[%MW0] THEN     (* Si la entrada está activa
                INC %MW4;               aumentamos el valor de MW4 *)
        END_IF;
END_FOR;
```

Etapa 2

```
!FOR %MW0:=0 TO %MW4 DO    (* Recorremos las %MW4 primeras salidas *)
        SET %Q2.0[%MW0];          (* las activamos *)
END_FOR;
```

Etapa 3

```
! FOR %MW0:=0 TO %MW4 DO   (* Recorremos las %MW4 últimas salidas *)
        %MW1:=15-%MW0;     (* para empezar por la última *)
        SET %Q2.0[%MW1];   (* las activamos *)
END_FOR;
```

Etapa 4

Acción continua:

```
! %Q2.0:16:=%MW10;          (* Copiamos en la tabla de salidas el contenido de
                            MW10 , donde se ha guardado la tabla de entrada*)
```

Acción desactivación:

```
! %Q2.0:16:=0;
```