Ejercicio 1 (0.5 puntos): Diseñar un comparador con capacitación que tenga la siguiente interfaz de entrada/salida:

```
entity comparador is
  port (x: in std_logic_vector(3 downto 0);
        y: in std_logic_vector(3 downto 0);
        c: in std_logic;
        s: out std_logic
  );
end comparador;
```

- x: Primera cadena de 3 bits de entrada.
- y: Segunda cadena de 3 bits de entrada.
- c: Señal de capacitación del comparador.
- s: Salida.

Cuando c valga '1', la salida s deberá devolver '1' si y es mayor que x y además, difiere con x en más 1 bit. Nota: Para comprobar esto, podéis utilizar la operación binaria XOR. Si c es '0', el comparador siempre devolverá '0'.

Ejercicio 2 (0.5 puntos): Integrar el componente diseñado en el ejercicio anterior en vuestro sistema para que, cuando el usuario pida introducir una nueva clave, ésta sólo se escriba en la memoria si la nueva clave es mayor que la antigua clave, y ambas difieren en más de 1 bit (en otras palabras, cuando el comparador desarrollado en el ejercicio anterior devuelva '1'). **Nota: No necesitáis modificar vuestro diagrama ASM en absoluto para realizar este ejercicio.**

Ejercicio 3 (1 punto): Modificar el diseño básico para que la memoria RAM de vuestra práctica sea directamente modificable desde el exterior. La interfaz de entrada/salida del sistema en esta nueva versión deberá seguir siendo la misma.

Con esta modificación, el usuario podrá introducir una nueva clave cuando el sistema esté en su estado inicial. Para ello, activará la señal *escribir* a '1', pondrá una dirección en un **nuevo puerto** llamado *dir_in* y el valor de la nueva clave, en la señal *clave*. El resto del funcionamiento del sistema debe permanecer intacto. Por tanto, **en este ejercicio no deberéis introducir ningún cambio en vuestro diagrama ASM, sino sólo en la ruta de datos y/o en la unidad de control.**