Desarrollo de Aplicaciones Telemáticas, examen práctico, 18 de mayo de 2026
Grado en Ingeniería en Tecnologías de la Telecomunicación
Grado en Ingeniería en Sistemas de la Telecomunicación
Universidad Rey Juan Carlos

Instrucciones

- Emplea lo visto en clase y solo lo visto en clase (que puedes consultar en las transparencias). No será válido ni JavaScript más antiguo ni más avanzado. Usa el modo estricto.

Ejercicio 1. (4.0 puntos)

Ya conoces la *paradoja del cumpleaños*. Definiremos ahora una idea relacionada pero distinta: si dos personas en un grupo cumplen años en la misma semana, consideramos que se cumple la *paradoja laxa*. Lo que entendemos aquí por *semana* lo veremos posteriormente. Escribe un programa en JavaScript llamado *laxa.js* según la siguiente especificación.

1. Escribe todas las funciones que consideres conveniente, pero la principal se llamará *laxa()*. Recibirá un argumento, una cadena como esta:

   `"3 enero,3 enero,12 enero,26 diciembre,31 diciembre,14 mayo,16 mayo"`

   Esto es:

   a) La cadena tendrá al menos una fecha, no podrá ser la cadena vacía.

   b) Entre fecha y fecha, habrá una coma. Tras la última fecha, no.

   c) Cada fecha estará compuesta por el día del mes, un espacio y el nombre del mes.

   d) El mes estará escrito en español, todo con minúsculas.

   e) Ignoraremos los años bisiestos.

   f) No debes preocuparte por lo que haga el programa si la cadena incumple este formato. Suponemos que los datos ya están previamente filtrados, y si el filtro es incorrecto, no es responsabilidad de esta función. Ejemplos de fechas erróneas que puedes ignorar:

   ```
   ""                Fecha vacía.
   "21abril"         Falta el espacio
   "30 febrero"      Es una fecha errónea.
   "21  abril"       Sobra un espacio entre la fecha y el mes.
   "21 abril "       Sobra un espacio al final.
   "21 Abril"        Hay una mayúscula.
   "21 april"        No está en español.
   "29 febrero"      Es válido solo en bisiesto.
   ... entre otros casos.
   ```

2. El propósito de esta función es saber, en cada semana del año, qué cumpleaños hay. La estructura de datos principal será un array llamada *semanas* que contendrá 53 elementos. El elemento i de este array contendrá un array anidado con todas las fechas correspondientes a esa semana. Para el ejemplo anterior:

   ```
   Elemento      Contenido
   1             ["3 enero","3 enero"]
   2             ["12 enero"]
   20            ["14 mayo","16 mayo"]
   52            ["26 diciembre","31 diciembre"]
   ```

   El resto de elementos, los correspondientes a semanas sin cumpleaños, contendrán lo que consideres más adecuado: *undefined*, *null*, un hueco o cualquier otro valor similar.

   Observa que deseamos que la primera semana sea la semana 1, no la semana 0. Pero recuerda que en JavaScript, el primer elemento de un array es el 0, no el 1. Y la especificación pide que

el elemento 1 (el segundo) contenga los cumpleaños de la primera semana. Así que habrá 53 posiciones (0–52), pero la posición 0 quedará vacía para que el índice coincida con el número de semana.

3. Observa que para saber a qué semana corresponde una fecha (según nuestra definición de *semana*), basta aplicar la expresión

$$\text{semana} = \left\lfloor \frac{\text{díaDelAño} - 1}{7} \right\rfloor + 1$$

   Por ejemplo, el 7 de enero se corresponde con la semana 1, y el 8, con la semana 2.

   Con una excepción: un año no tiene exactamente 52 semanas, sino 52 semanas y un día (52 * 7 = 364). Este día *de más*, el último del año, lo incluiremos en la semana 52 que será una semana especial de 8 días (la alternativa sería una semana especial de 1 día).

   Por tanto el array tendrá 53 posiciones: la 0, vacía. De la 1 a la 51, semanas de 7 días, y la posición 52, una semana de 8 días.

4. La función *laxa()* devolverá este array *semanas*.

5. Si una misma fecha aparece varias veces, cada aparición se considera una persona distinta.

6. Escribe además una función llamada *muestra_semanas()*, o *muestraSemanas()* si lo prefieres, que reciba la lista *semanas* y escriba en la consola los datos de todas las semanas donde se da la *paradoja laxa*, esto es, donde haya 2 o más personas. Las semanas con 0 o 1 cumpleaños, no se escribirán. La salida será similar a esta:

```
Semana      Cumpleaños
1           3 enero, 3 enero
20          14 mayo, 16 mayo
52          26 diciembre, 31 diciembre
```

7. Si la función *laxa()* no recibe exactamente un argumento, de tipo cadena, el programa lanzará una excepción de forma similar a como lo hace el motor de JavaScript. Recuerda que el formato de la cadena con la serie de cumpleaños no debes revisarlo, basta comprobar que es una cadena.

8. La función *muestra_semanas()* debe comprobar que recibe exactamente un argumento, de tipo array, con 53 elementos y lanzar una excepción similar a la descrita anteriormente, si procede. Sin comprobaciones adicionales sobre estos 53 elementos.

9. En el fichero *laxa.js*, escribe una invocación a *laxa()*, pasándole el ejemplo del enunciado, que encontrarás en el aula virtual. Con el resultado, llama a *muestra_semanas()*.

Ejercicio 2. (6.0 puntos)

Modifica tu aplicación de auto-pedido según la siguiente especificación:

- En la tabla donde se muestran los productos aparecerá una columna adicional: el precio en euros.
- En la tabla de pedidos, aparecerán dos columnas adicionales: *precio unitario* y *total*. Por ejemplo, 2 cafés con un precio unitario de 1.50 EUR hacen un total de 3.00 EUR.
- Debajo de la tabla de pedidos, aparecerá una nueva tabla con el importe total de todo el pedido.

- Este total se actualizará cada vez que se añada o elimine un artículo al pedido.
- Todos los precios son finales. No consideres el IVA en ningún caso.
- El local tendrá dos tipos de precios: *normal* y *reducido*. Supón que el primero se corresponde con el fin de semana, con las noches y los festivos. Y el segundo con comidas a diario. Habrá un botón para elegir entre un modo y otro. Naturalmente no es realista que este botón esté al alcance del cliente, pero ignora esto.
- En modo *reducido*, todos los precios serán más baratos. Según lo indicado en la constante global `PorcentajeReducido`, definida en el código fuente de tu programa. Ponle el valor 20, esto es, un precio un 20 % inferior al precio normal. No es un descuento: el cliente no verá el precio original y el descuento. Simplemente, en modo reducido verá precios más baratos. Posiblemente con cantidades no *redondas*, lo cual tampoco es realista, ignóralo.
- Todos los precios aparecerán con dos decimales. Por ejemplo 11.20 EUR.
- En modo *Normal*, el texto del botón será *Reducido*. Y viceversa. Esto es, el botón indica a qué modo se pasará al pulsarse, no el estado actual.
- Si el botón normal/reducido se pulsa en mitad de un pedido, podrán pasar *cosas raras*. No es problema. Basta con que tengas claro qué sucede, si el profesor te lo pregunta.