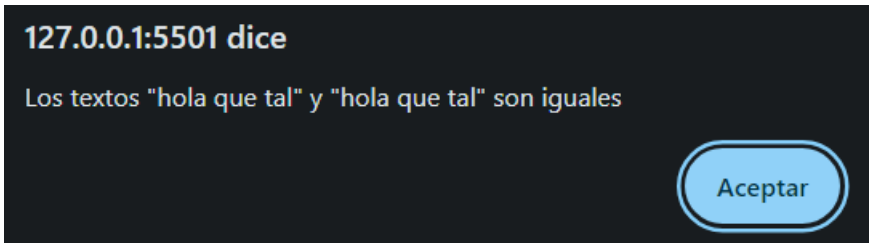
DWEC 2º DAW

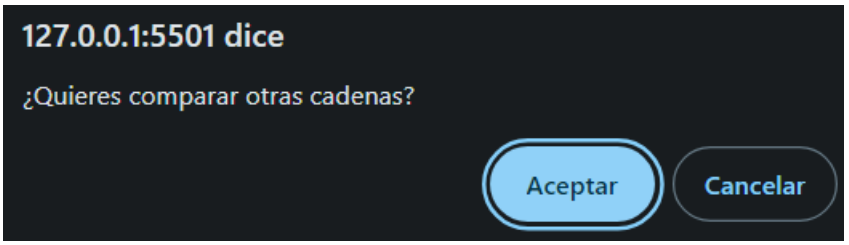
UNIDAD 2: MANEJO DE SINTAXIS. USO DE DATOS PRIMITIVOS

PRÁCTICA STRINGS

1. Crea un programa que, tras pedir dos cadenas de texto al usuario, las compare y de información sobre la comparación. Deberemos
    a. Eliminar los posibles espacios delanteros y traseros de las cadenas que el usuario haya podido incluir de manera fortuita (o no).
    b. Usaremos una arrow function para la comparación de las cadenas
    c. Mostraremos un mensaje por pantalla diciendo al usuario si sus textos son iguales o no. El mensaje debe incluir los textos.



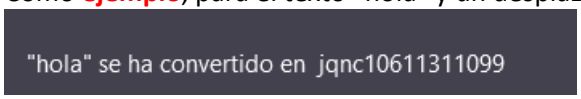
    d. Debemos seguir ofreciendo esta comparación de textos hasta que el usuario decida cancelar.



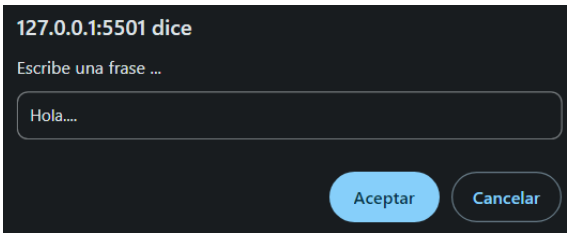
2. Crea un programa que cifre un texto al estilo César, pero basado en la tabla Unicode: El cifrado César consiste en tomar una letra y desplazarla en el alfabeto un número de posiciones concretas (lo determina una clave). Para realizar nuestra versión de cifrado Unicode deberemos:
    a. Pedir al usuario el texto a cifrar y un número entero (será el desplazamiento).
    b. Deberemos asegurarnos de que lo que nos da es un número y, si no lo es, lo seguiremos pidiendo hasta que lo sea.
    c. Deberemos eliminar posibles decimales del número recogido.
    d. Para cada carácter de nuestro texto, guardaremos la posición numérica correspondiente al código Unicode y le sumaremos el desplazamiento que el usuario nos dio.

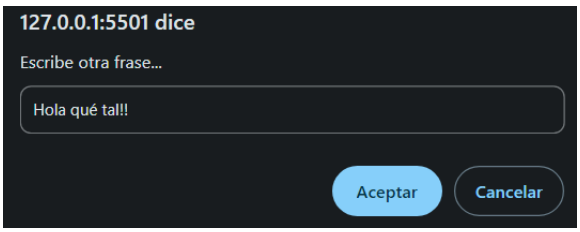
e. Por otro lado, y también para cada carácter de nuestro texto, guardaremos el nuevo carácter correspondiente al código Unicode según la posición numérica anterior.
f. El texto cifrado será la unión de estas dos cadenas.
g. Por último, mostraremos por pantalla un mensaje en el que informe que su texto inicial se ha convertido (cifrado) en el texto final.

Como **ejemplo**, para el texto "hola" y un desplazamiento de 2, este sería el resultado:

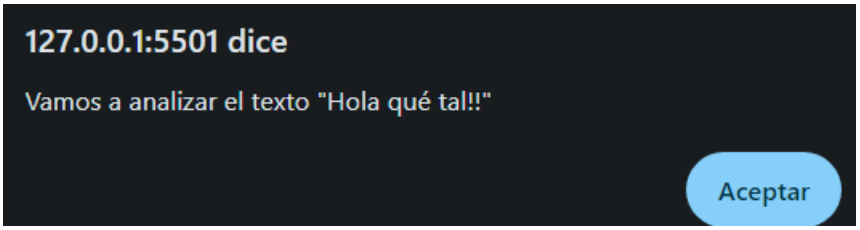


3. Crea un programa en el que, partiendo de una cadena, se extraigan todos sus caracteres (sin repetirlos) y pasen a otra cadena. Deberemos:
   a) Pedir dos cadenas y trabajar sobre la más larga. Ejemplo:

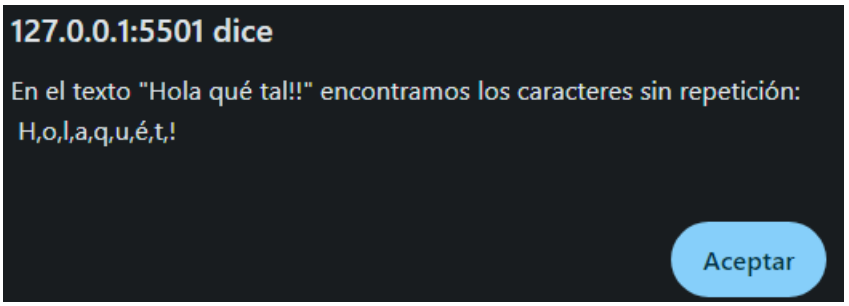




   b) Mostraremos por pantalla sobre cuál de las dos cadenas vamos a trabajar.



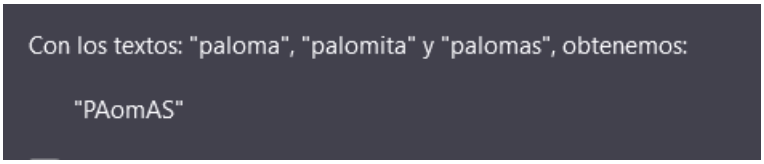
   c) Mostrar por pantalla los caracteres NO repetidos (separados por comas) que contiene la cadena inicial (que aparecerá en el mensaje.



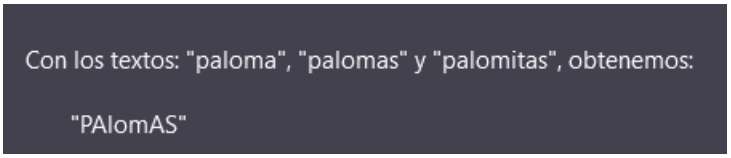
4. Crea un programa que, partiendo de 3 palabras, obtenga una cuarta compuesta por el inicio de la 1ª, la parte central de la 2º y el final de la 3ª. Para ello:
    a) Las palabras las pediremos al usuario y lo haremos hasta que se cumplan las condiciones de cada una de ellas:
        - La 1ª palabra deberá contener mínimo 4 caracteres,
        - La 2ª palabra deberá contener mínimo 5 caracteres y
        - La 3ª mínimo 6 caracteres.
        - Los espacios NO se consideran caracteres

    b) La nueva palabra estará compuesta por:
        - **Los dos primeros caracteres de la 1ª palabra** en **mayúsculas**.
        - **La parte central de la segunda en minúsculas:**
            - Si el número de caracteres de la palabra es **par**, la parte central serán dos caracteres y,
            - Si el número de caracteres de la palabra es **impar**, la parte central serán tres caracteres.
        - Los dos últimos caracteres de la 3ª cadena en mayúsculas.

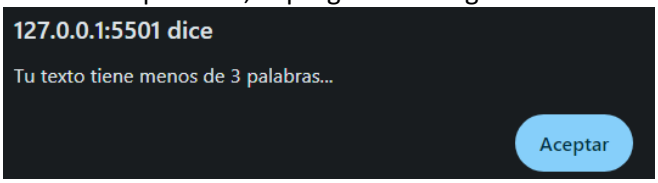
    Como ejemplo1, para las cadenas “paloma”, “palomita” y “palomitas”, este sería el resultado → la 2ª cadena es par

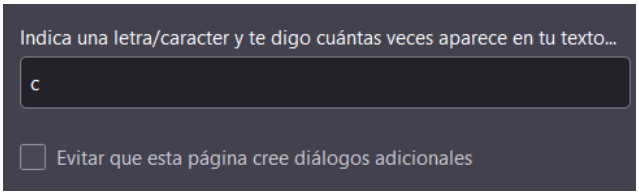
    Como ejemplo2, para las cadenas “paloma”, “palomas” y “palomitas”, este sería el resultado → la 2ª cadena es impar

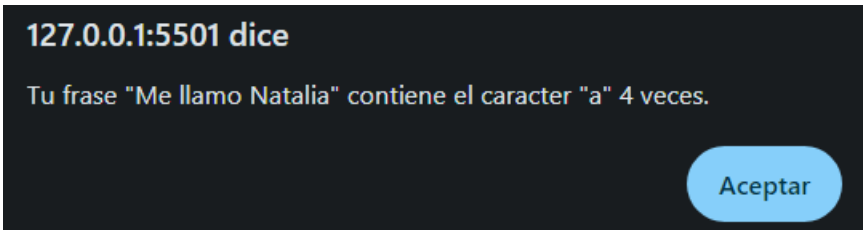
5. Haz un programa que cuente cuántas veces aparece en un texto un carácter determinado. Para ello:
    a) El texto deberá tener como mínimo 3 palabras. Lo pediremos al usuario. Si tiene menos de 3 palabras, el programa no sigue.

    b) Lo que debemos buscar será una cadena con un solo carácter. Lo pediremos al usuario.

c) Mostraremos un mensaje por pantalla diciendo que en el texto (debe aparecer) el carácter (debe aparecer) aparece XX veces.



6. Crea un programa que, partiendo de 1 cadena **construida por tí mismo**, genere cadenas de caracteres aleatorios y muestre por consola esas subcadenas y el número de cadenas vacías, si las hay. Para ello:
    a) Teniendo ya la cadena de caracteres, pediremos al usuario el número de cadenas que desea.
    b) Generamos ese número de cadenas de longitudes aleatorias (de 0 a 10 elementos máximo)
    c) Mostraremos por consola las cadenas y el número de cadenas vacías.

Como ejemplo, partiendo de:

- Una **cadena con el abecedario en may, min y los nºs de 0 al 9**

```
const ABCNUM = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabecdefghijklmnopqrstuvwxyz0123456789';
```

- El número de cadenas que ha dado el usuario es 5

```
Cadena nº 1: JIzH4Uil
Cadena nº 2: T
Cadena nº 3: aBElN7An
Cadena nº 4: Ojc7PyU8H
Cadena nº 5:
En el conjunto de cadenas aleatorias encontramos 1 elemento/s vacío/s
```