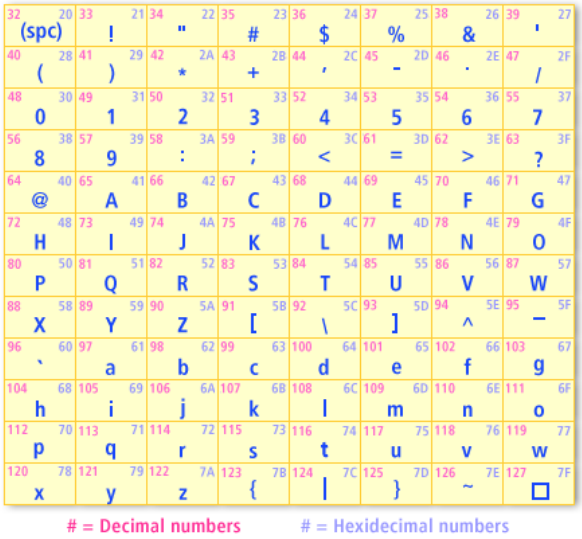
CLASE STRING – EJERCICIOS PROPUESTOS

1. Pedir al usuario su nombre y saludarlo.
2. Programa que diga si una cadena dada por el usuario está o no vacía.
3. Pedir al usuario una cadena y decir si empieza por el carácter 'H'.
4. Como el anterior, pero sin importar si es mayúscula o minúscula.
5. Método que pida un carácter c y un número n y que devuelva una cadena con c repetido n veces.
6. Elabora un método que escriba todos los caracteres de una cadena cada uno en una línea.
7. Pedir al usuario una frase y un carácter, y decir en qué posiciones está ese carácter, o indicar que no está si procede. No se puede usar indexOf o similar.
   a. Repite el ejercicio anterior, pero usando indexOf.
8. Método que reciba una cadena y retorne el número de vocales que contiene.
9. Método que reciba una cadena y la transforme sustituyendo todos los espacios por * (sin usar replace ni replaceAll).
10. Método que reciba una cadena u un carácter y la retorne pero sólo hasta la primera aparición del carácter dado (sin incluirlo). No se puede utilizar indexOf.
    a. Repite el ejercicio anterior usando indexOf.
11. Pedir una frase y eliminar las vocales. No se puede usar replace ni replaceAll.
    a. Repite el ejercicio anterior usando replace .
12. Método que sustituya mayúsculas por minúsculas y minúsculas por mayúsculas en una frase. Utilizar Character.toUpperCase(char c) y Character.toLowerCase(char c).
13. Método que reciba una cadena y comprueba si el balanceo de paréntesis es correcto (se abren y se cierran correctamente).
14. Método que retorne cuántas palabras de menos de 3 letras tiene una frase, considerando como separador de palabras únicamente el inicio de cadena, el espacio y el fin de cadena.
15. Método que reciba dos cadenas y las junte carácter a carácter. Es decir, si recibe "pata" y "coco" el resultado es "pcaotcao". Las cadenas tienen que ser del mismo tamaño.
    a. Mejorar el anterior para que no importe que las palabras sean de distinto tamaño. Es decir, si recibe "patata" y "coco" el resultado sería "pcaotcaota", añadiendo la cadena sobrante directamente al final.
16. Método que cuente el número de veces que aparece un carácter en una cadena.
17. Método que elimina los caracteres en blanco al principio de una cadena. No puedes usar trim.
    a. Otro que elimine los caracteres en blanco finales de una cadena. No puedes usar trim.
    b. Otro que quite los caracteres en blanco a la izquierda y a la derecha de una cadena. No puedes usar trim.
18. Método que reciba una cadena de caracteres, donde en lugar de ñ se han utilizado los caracteres ny. Crear una nueva cadena de caracteres sustituyendo ny por ñ. No puedes usar replace.
19. Método que reciba una cadena y dos caracteres y reemplace en la cadena todas las ocurrencias del primer carácter por el segundo. Sin utilizar replace.

20. Implementar las operaciones de inserción, borrado, búsqueda y copia de caracteres dentro de una cadena. Sólo se permite utilizar la función Subcadena(cadena,pos,pos) para una posición, no para varios caracteres.
21. Dada una cadena de entrada, comprobar si es una contraseña FUERTE o DÉBIL. Se considera que una contraseña es fuerte si contiene 8 o más caracteres, y entre ellos al menos hay una mayúscula, una minúscula, un signo de puntuación y un dígito.
    Tener en cuenta los valores decimales de los caracteres UNICODE.



22. Desarrollar una Clase StringUtil, que implemente los siguientes métodos:
    a. public static String insertar (String ori, String insertado)
    b. public static String borrar (String ori, String borrar)
    c. public static String reemplazar (String ori, String buscado, String reemplazo)
    d. public static String capitalize (String ori)
    e. public static int cuentaPalabras (String ori)
    f. public static int cuentaParrafos (String ori)
23. Desarrollar dos métodos para codificar y decodificar un texto utilizando el método de cifrado de César. Supondremos que el texto solo contiene letras mayúsculas o minúsculas. Las letras serán las correspondientes al alfabeto inglés (26 caracteres, excluimos la ñ y Ñ).
    En este método de cifrado cada letra del texto se sustituye por otra letra que se encuentra **n** posiciones adelante en el alfabeto, la **n** se recibe también como parámetro.
    Se considera que el alfabeto es circular, es decir, la letra siguiente a la 'z' es la 'a'.
    Por ejemplo, si **n** es 3, las transformaciones serían:
    la 'a' se transformaría en 'd',
    la 'b' en 'e',
    la 'c' en 'f',
    la 'w' en 'z',
    la 'x' en 'a',
    la 'y' en 'b', etc. De la misma manera se comportarían las mayúsculas.
    Ejemplo de cifrado César: si el texto es "capaz" y **n** = 3 el texto cifrado es "fdsdc".
    Para descifrar un texto se realiza la operación contraria. Se calcula la letra que está **n** posiciones por detrás en el alfabeto. Como el alfabeto es circular, la letra anterior a la 'a' es la 'z'.
    Trabajar con los valores decimales de los UNICODE.