
CLASE STRING – EJERCICIOS PROPUESTOS

1. Pedir al usuario una cadena y decir si empieza por el carácter 'H'.
2. Como el anterior, pero sin importar si es mayúscula o minúscula.
3. Elabora un método que escriba todos los caracteres de una cadena cada uno en una línea.
4. Pedir al usuario una frase y un carácter, y decir si la frase contiene ese carácter.
5. Método que pida un carácter c y un número n y que devuelva una cadena con c repetido n veces. Ej: repite('a', 4) debe retornar "aaaa"
6. Método que cuente el número de veces que aparece un carácter en una cadena.
7. Método que reciba una cadena y retorne el número de vocales que contiene.
8. Desarrollar un método que reciba una cadena y retorne otra con la original invertida.
9. Desarrollar un método que indique si una palabra es un palíndromo.
 - a. Modificarlo para que indique si una frase es un palíndromo.
10. Desarrollar un método que, dada una cadena, retorne su sigla, es decir, la primera letra de cada palabra en mayúsculas. Por ejemplo, si recibe 'Universal serial bus' debe devolver 'USB'.
11. Desarrollar un método que reciba una cadena y un carácter e inserte el carácter entre cada letra de la cadena. Ej: "separar" y '-' debería retornar "s-e-p-a-r-a-r"
12. Desarrollar un método que reciba una cadena e inserte un carácter cada 3 caracteres. Por ejemplo: "1921680012" y '.', debe retornar "192.168.001.2"
13. Haga otra versión del método, pero que reciba una cadena, un carácter y la cantidad de dígitos entre las que se debe insertar el carácter.
14. Desarrolle un método llamado capitalize que ponga en mayúsculas cada palabra de una frase recibida como parámetro. Por ej: si se recibe "Congreso internacional de programación de String", debe devolver, "Congreso Internacional De Programación De String". Utilice Character.toUpperCase(char c)
15. Desarrolle un método que reciba una cadena con una expresión de una operación simple (+, -, *, /) con dos operandos y retorne el resultado como double.
- 16.
17. 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.
18. Método que reciba una cadena y la transforme sustituyendo todos los espacios por * (sin usar replace ni replaceAll).
19. Método que reciba una cadena y 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.
20. Pedir una frase y eliminar las vocales. No se puede usar replace ni replaceAll.
 - a. Repite el ejercicio anterior usando replace.
21. Método que sustituya mayúsculas por minúsculas y minúsculas por mayúsculas en una frase. Utilizar Character.toUpperCase(char c) y Character.toLowerCase(char c).
22. Método que reciba una cadena y compruebe si el balanceo de paréntesis es correcto (se abren y se cierran correctamente).
23. 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.

24. 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.
- 25.
26. Método que elimine 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.
27. 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.
28. 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.
29. Este ejercicio
30. 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.

32	20	33	21	34	22	35	23	36	24	37	25	38	26	39	27
(spc)	!	"	#	\$	%	&	'								
40	28	41	29	42	2A	43	2B	44	2C	45	2D	46	2E	47	2F
()	*	+	,	-	.	/								
48	30	49	31	50	32	51	33	52	34	53	35	54	36	55	37
0	1	2	3	4	5	6	7								
56	38	57	39	58	3A	59	3B	60	3C	61	3D	62	3E	63	3F
8	9	:	;	<	=	>	?								
64	40	65	41	66	42	67	43	68	44	69	45	70	46	71	47
@	A	B	C	D	E	F	G								
72	48	73	49	74	4A	75	4B	76	4C	77	4D	78	4E	79	4F
H	I	J	K	L	M	N	O								
80	50	81	51	82	52	83	53	84	54	85	55	86	56	87	57
P	Q	R	S	T	U	V	W								
88	58	89	59	90	5A	91	5B	92	5C	93	5D	94	5E	95	5F
X	Y	Z	[\]	^	_								
96	60	97	61	98	62	99	63	100	64	101	65	102	66	103	67
,	a	b	c	d	e	f	g								
104	68	105	69	106	6A	107	6B	108	6C	109	6D	110	6E	111	6F
h	i	j	k	l	m	n	o								
112	70	113	71	114	72	115	73	116	74	117	75	118	76	119	77
p	q	r	s	t	u	v	w								
120	78	121	79	122	7A	123	7B	124	7C	125	7D	126	7E	127	7F
x	y	z	{		}	~									

= Decimal numbers

= Hexidecimal numbers

31. Desarrollar una Clase StringUtil, que implemente los siguientes métodos:
 - a. public static String insert(String cadena, String insertado, int pos)
 - b. public static String delete(String cadena, int desde, int hasta)
 - c. int indexOf(String cadena, String buscado)
 - d. public static String replace(String cadena, String buscado, String reemplazo)
 - e. public static String capitalize(String cadena)
 - f. public static int cuentaPalabras(String cadena)
 - g. public static int cuentaParrafos(String cadena)
32. 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.