"Los ordenadores son buenos siguiendo instrucciones,
no leyendo tu mente".
Donald Knuth

Práctica 1

Ejercicio 1:
Diseña un programa que lea la edad de dos personas y diga quién es más joven, la primera o la segunda. Ten en cuenta que ambas pueden tener la misma edad. En tal caso, hazlo saber con un mensaje adecuado.

Ejercicio 2:
Diseña un programa que lea un carácter de teclado y muestre por pantalla el mensaje "Es paréntesis" sólo si el carácter leído es un paréntesis abierto o cerrado. En caso contrario muestra el mensaje “No es paréntesis”.

Ejercicio 3:
Diseña un programa que, dado un número entero, muestre por pantalla el mensaje "El número es par." cuando el número sea par y el mensaje "El número es impar." cuando sea impar. (Una pista: un número es par si el resto de dividirlo por 2 es 0, e impar en caso contrario.)

Ejercicio 4:
Diseña un programa que, dado un número entero, determine si éste es el doble de un número impar. (Ejemplo: 14 es el doble de 7, que es impar.)

Ejercicio 5:
Escribir un programa que permita ingresar dos números enteros y escribirlos en orden creciente.

Ejercicio 6:
Escribir un programa que permita ingresar tres números enteros y escribirlos en orden creciente.

Ejercicio 7:
Escribir un programa que permita ingresar el mes y el año y muestre la cantidad de días del mes. Por ejemplo, si el usuario ingresa mes 2 y año 2000, el programa debe responder que Febrero 2000 tiene 29 días. Si el usuario ingresa mes 3 y año 2005, el programa responderá que Marzo 2005 tiene 31 días.

Ejercicio 8:
Escribir un programa que permita ingresar las coordenadas (x,y) de un punto en el plano y verifique si el punto está dentro del círculo con centro en (0,0) y radio 10, o está fuera o en la circunferencia. Mostrar los mensajes adecuados.

"La mayoría de los buenos programadores
programan no porque esperan que les paguen
o que el público los adore, sino porque programar es divertido".
Linus Torvalds

Práctica 2

Ejercicio 1:
Implementa un programa que muestre todos los múltiplos de 6 entre 6 y 150, ambos inclusive.

Ejercicio2:
Diseña un programa que solicite la lectura de un número entre 0 y 10 (ambos inclusive). Si el usuario teclea un número fuera del rango válido, el programa solicitará nuevamente la introducción del valor cuantas veces sea menester.

Ejercicio 3:
Escribir un programa que muestre, de a diez números por línea y separados por un blanco, todos los números entre 100 y 1000 que sean divisibles por 5 y 6.

Ejercicio 4:
Construir un programa que permita multiplicar dos números enteros positivos empleando el método denominado *multiplicación rusa*. Este método permite calcular el producto de N por M dela siguiente forma:

Ejercicio 5:
En pasos sucesivos se divide M por 2 (división entera) y se multiplica N por 2. Este proceso se repite hasta que M es 0. El resultado de la multiplicación deseada se obtiene acumulando aquellos valores sucesivos de N para los cuales el valor correspondiente de M es impar.

Ejemplo: 31 * 27

N	M
* 31	27
* 62	13
124	6
* 248	3
* 496	1
992	0

Si sumamos los valores marcados con un asterisco (que son los que corresponden a un valor de M impar) obtenemos: 31 + 62 + 248 + 496 = 837 = 31 * 27

Ejercicio 6:
Diseñar un programa que genere una lista de números usando el siguiente proceso: comenzar con un entero n que deberá ingresar el usuario. Si n es par, dividirlo por 2. Si n es impar, multiplicarlo por 3 y sumarle 1. Repetir este proceso hasta que el nuevo valor obtenido para n sea 1.
Ejemplo, para n = 22, la secuencia que se obtiene es:
22 11 33 17 52 26 13 40 20 10 5 16 8 4 2 1

"Programa siempre tu código como
si el tipo que va a tener que mantenerlo
en el futuro fuera un violento psicópata que sabe dónde vives".
Martin Goldin

Práctica 3

Ejercicio 1:
Escribir una función que reciba una cadena que contiene un número entero largo y devuelva una cadena con el número y las separaciones de miles. Por ejemplo, si recibe '1234567890', debe devolver '1.234.567.890'.

Ejercicio 2:
Una palabra es "alfabética" si todas sus letras están ordenadas alfabéticamente. Por ejemplo, "amor", "chino" e "himno" son palabras "alfabéticas". Diseña un programa que lea una palabra y nos diga si es alfabética o no.

Ejercicio 3:
Hay un tipo de pasatiempos que propone descifrar un texto del que se han suprimido las vocales. Por ejemplo, el texto ".n .j.mpl. d. p.s.t..mp.s", se descifra sustituyendo cada punto con una vocal del texto. La solución es "un ejemplo de pasatiempos". Diseña un programa que ayude al creador de pasatiempos. El programa recibirá una cadena y mostrará otra en la que cada vocal ha sido reemplazada por un punto.

Ejercicio 4:
Haz un programa que lea dos cadenas que representen sendos números binarios. A continuación, el programa mostrará el número binario que resulta de sumar ambos (y que será otra cadena). Si, por ejemplo, el usuario introduce las cadenas '100' y '111', el programa mostrará como resultado la cadena '1011'.
Nota: El procedimiento de suma con acarreo que implementes deberá trabajar directamente con la representación binaria leída.

Ejercicio 5:
Escribir un programa que verifique si un string es una password correcta. Las reglas para determinar si es correcta son:

- Debe tener como mínimo 8 caracteres.
- Sólo puede tener letras y dígitos.
- Como mínimo debe tener dos dígitos.

Ejercicio 6:
Escribir un programa que retorne el número que le corresponde a una letra en mayúscula de acuerdo al teclado telefónico (ver figura):

Ejercicio 7:
Escribir un programa que ingrese un número telefónico como un string y convierta los caracteres a el dígito correspondiente, ejemplo:

1-800-FLOWERS ---> 1-800-3569377

Ejercicio 8:
Un ISBN-10 (International Standard Book Number) consiste de 10 dígitos: $d_1d_2d_3d_4d_5d_6d_7d_8d_9d_{10}$.
El último dígito, d_{10}, es el dígito verificador que se calcula como sigue:

$$(d_1 \times 1 + d_2 \times 2 + d_3 \times 3 + d_4 \times 4 + d_5 \times 5 + d_6 \times 6 + d_7 \times 7 + d_8 \times 8 + d_9 \times 9) \% 11$$

Si el dígito verificador es 10, el último dígito es x, de acuerdo a las normas ISBN. Escribir un programa que permita ingresar los primeros 9 dígitos como una cadena y muestre el número ISBN.
Ejemplo: 013601267 ---> 0136012671
013031997 ---> 013601267X

Ejercicio 9:
ISBN-13 es un nuevo estandar para identificar libros. Usa 13 dígitos: $d_1d_2d_3d_4d_5d_6d_7d_8d_9d_{10}d_{11}d_{12}d_{13}$.
El último dígito, es el dígito verificador y se calcula con la siguiente fórmula:

$$10 - (d_1 + 3d_2 + d_3 + 3d_4 + d_5 + 3d_6 + d_7 + 3d_8 + d_9 + 3d_{10} + d_{11} + 3d_{12}) \% 10$$

Si el dígito verificador es 10 se reemplaza por 0. El programa deberá permitir ingresar un número como un string y mostrar el ISBN-13, ejemplo:

978013213080 ---> 9780132130806

978013213079 ---> 9780132130790

Ejercicio 10:
Escribir una programa que permita ingresar un texto y un caracter e imprima la palabra más larga en la que se encuentra ese carácter.

Ejercicio 11:
La ley de Chargaff dice que en el ADN de un organismo la cantidad de Adenina es la misma que la de Tiamina, y la de Citosina es la misma que la de Guanina. Dada una secuencia de nucleótidos del estilo de ATTACCAGTACA... podemos comprobar si cumple dicha ley de la siguiente forma:
Contamos la cantidad de A, T, C y G presentes en la cadena y calculamos los coeficientes

$$a = \frac{N_A - N_T}{N_A + N_T} \qquad y \qquad c = \frac{N_C - N_G}{N_C + N_G}$$

donde N_X indica la cantidad de nucleótidos del tipo X presentes en la secuencia.
Partiremos de una cadena que contiene una cantidad indeterminada de caracteres, que solo pueden ser A, T, G ó C. Calcula a partir de dicha cadena los coeficientes ***a*** y ***c***.

Ejercicio 12:
Escribe un programa que lea dos cadenas y devuelva el prefijo común más largo de ambas. Ejemplo: las cadenas "politécnico" y "polinización" tienen como prefijo común más largo a la cadena "poli".

Ejercicio 13:
Necesitamos buscar en diversas secuencias de ARN las posibles apariciones del codón iniciador 'AUG', que marca el inicio de un gen. Como en una secuencia de ARN puede haber más de un gen, deseamos conocer todas las posiciones en las que aparece dicho codón. Para ello elaboraremos un programa que ingresará una cadena de caracteres que representa una secuencia de ARN y comprobará que la secuencia de ARN contiene únicamente los caracteres 'A','U', 'G' ó 'C'. En caso contrario, la secuencia es inválida y se deberá imprimir un mensaje adecuado.

Ejercicio 14 (opcional):
Los biólogos usan una secuencia de letras A, C, T, y G para modelar un genoma. Un gen es un subcadena de un genoma que comienza después de la tripleta ATG y termina con una tripleta TAG, TAA, ó TGA. La longiutd de una cadena de gen es un múltiplo de 3 y el gen no contiene a las tripletas ATG, TAG, TAA y TGA. Escribir un programa que permita ingresar un genoma y muestre todos los genes en el genoma. Si en la cadena no se encuentran genes, el programa mostrará un mensaje acorde. Ejemplo:
si la cadena es TTATGTTTTAAGGATGGGGCGTTAGTT, el programa mostrará:
TTT

GGGCGT
Si la cadena es TGTGTGTATAT entonces se deberá mostrar "no se encontraron genes".

Ejercicio 15:

Dos palabras son anagramas si tienen las mismas letras pero en otro orden. Por ejemplo, «torpes» y «postre» son anagramas, mientras que «aparta» y «raptar» no lo son, ya que «raptar» tiene una r de más y una a de menos.

Escriba la función `sonAnagramas(p1, p2)` que indique si las palabras `p1` y `p2` son anagramas:

```
sonAnagramas('torpes', 'postre') ---> True
sonAnagramas('aparta', 'raptar') ---> False
```

Ejercicio 16:

Las palabras **panvocálicas** son las que tienen las cinco vocales. Por ejemplo: centrifugado, bisabuelo, hipotenusa. Escriba la función `esPanvocalica(palabra)` que indique si una palabra es panvocálica o no:

```
esPanvocalica('educativo') ---> True
esPanvocalica('pedagogico') ---> False
```

"Existen 2 formas de desarrollar un diseño de software: una es hacerla tan simple que obviamente no hay deficiencias, y la otra es que sea tan complicada que no existan deficiencias obvias. El primer método es mucho más difícil".

C.A.R. Hoare

Práctica 4

Ejercicio 1:
Escribir funciones que, tomando como entrada una lista de números enteros, decida si la lista:

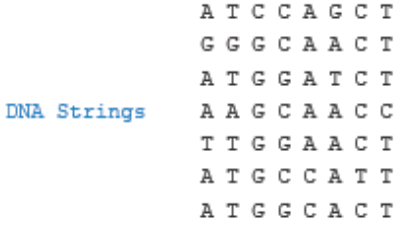
- está ordenada crecientemente.
- está ordenada decrecientemente.
- es *gaspariforme*. Se dice que un lista de n elementos es *gaspariforme* si todas sus *sumas parciales* son no negativas, y la suma total es igual a 0. Las *sumas parciales* de una lista a de n elementos está definida por

$$s_k = \sum_{i=0}^{k} a_i \quad \text{para k = 0,...,n-1}$$

Ejemplo:
Si a = [10, 5, 2, 20, 6], n = 5
$s_0 = 10$
$s_1 = 10 + 5 = 15$
$s_2 = 10 + 5 + 2 = 15 + 2 = 17$
$s_3 = 10 + 5 + 2 + 20 = 17 + 20 = 37$
$s_4 = 10 + 5 + 2 + 20 + 6 = 17 + 20 + 6 = 37 + 6 = 43$

- es *melchoriforme*. Se dice que una lista es *melchoriforme* si alguno de sus elementos es un *centro* . Un elemento es un *centro* si su valor coincide con la suma de los otros elementos de la lista.

Ejercicio 2:
Dada una matriz de n x m elementos (una lista que contiene n listas de tamaño m) donde cada fila representa una cadena de ADN de longitud m como la siguiente:



Obtener otra matriz de la siguiente forma:

		A	5	1	0	0	5	5	0	0
	Profile	C	0	0	1	4	2	0	6	1
		G	1	1	6	3	0	1	0	0
		T	1	5	0	0	0	1	1	6

donde en cada fila tenemos la cantidad de repeticiones de cada letra por columnas que forma un codón, además obtener una lista donde se muestra por columnas cuál es el símbolo que se repite más veces, para el ejemplo anterior se tiene como resultado:

Consensus	A	T	G	C	A	A	C	T

Ejercicio 3:
Diseñar una función que determine si una lista de números enteros está ordenada de menor a mayor y cada número i entre 1 y n aparece exactamente i veces.
Ejemplo:
para n = 5
esTelescopio(5,[1,2,2,3,3,3,4,4,4,4,5,5,5,5,5]) --> verdadero

Ejercicio 4:
Diseñar una función que genere una lista en forma de telescopio .
Ejemplo: genTelescopio(5) --> [1,2,2,3,3,3,4,4,4,4,5,5,5,5,5]

Ejercicio 5:
Diseñar una función que determine si una lista de números enteros, cada número i entre 1 y n aparece i veces consecutivas.
Ejemplo: sonConsecutivos(5,[3,3,3,1,2,2,5,5,5,5,5,4,4,4,4]) --> verdadero

Ejercicio 6:
Diseña una función que reciba una lista de cadenas y devuelva el prefijo común más largo. Por ejemplo, la cadena "pol" es el prefijo común más largo de esta lista: ['poliedro', 'policia', 'polifona', 'polinizar', 'polaridad', 'politica']

Ejercicio 7:
La ciudad de Babilonia tiene una alta congestión de vehículos circulando por sus calles. Las autoridades han decidido aplicar restricción vehicular para descongestionar la ciudad en base a las patentes de los vehículos.

La patente de un vehículo es un string de cuatro letras y dos dígitos, y la restricción depende sólo del penúltimo dígito. Por ejemplo, para la patente GEEA78, el dígito 7 es el utilizado para evaluar la restricción.

La restricción vehícular de los días hábiles de la semana se encuentra ingresada en el diccionario digitos, cuyas llaves son los días de la semanas, y cuyos valores son tuplas de cuatro enteros que representan los dígitos con restricción de ese día.

- Implemente la función estaConRestriccion(digitos,dia, patente), que indique si el vehículo está o no con restricción.
- Implemente la función diasConRestriccion(digitos,patente), que retorne la lista de los días en que el vehículo no puede circular.
- Implemente la función diasSinRestriccion(digitos,patente), que retorne el conjunto de los días en que el vehículo sí puede circular.

```
digitos = {'lunes':  (3, 4, 5, 6), 'martes': (7, 8, 9, 0),...,'miercoles': (1, 2, 3, 4), 'jueves':
                                (5, 6, 7, 8), ... ,   'viernes':   (9, 0, 1, 2)}
estaConRestriccion(digitos, 'lunes', 'BBDT35')  ---> True
diasConRestriccion(digitos, 'BBDT35') ---> ['lunes', 'miercoles']
diasSinRestriccion(digitos, 'BBDT35') ---> set(['jueves', 'martes', 'viernes'])
```

Ejercicio 8:

La empresa RawInput S.A. desea hacer una clasificación de sus clientes según su ubicación geográfica. Para esto, analizará su base de datos de correos electrónicos con el fin obtener información sobre el lugar de procedencia de cada cliente.

En una dirección de correo electrónico, el dominio es la parte que va después de la arroba, y el TLD es lo que va después del último punto. Por ejemplo, en la dirección fulano.de.tal@alumnos.usm.cl, el dominio es alumnos.usm.cl y el TLD es cl.

Algunos TLD no están asociados a un país, sino que representan otro tipo de entidades. Estos TLD genéricos son los siguentes:

```
genericos = {'com', 'gov', 'edu', 'org', 'net', 'mil'}
```

1. Escriba la función obtenerDominios(correos) que reciba como parámetro una lista de correos electrónicos, y retorne la lista de todos los dominios, sin repetir, y en orden alfabético.
2. Escriba la función contarTld(correos) que reciba como parámetro la lista de correos electrónicos, y retorne un diccionario que asocie a cada TLD la cantidad de veces que aparece en la lista. No debe incluir los TLD genéricos.

```
c = ['fulano@usm.cl', 'erika@lala.de', 'li@zi.cn', 'a@a.net', 'gudrun@lala.de',
'otto.von.d@lorem.ipsum.de', 'org@cn.de.cl', 'yayita@abc.cl', 'jozin@baz.cz', 'jp@foo.cl',
'dawei@hao.cn', 'pepe@gmail.com', 'ana@usm.cl', 'polo@hotmail.com', 'fer@x.com',
'ada@alumnos.usm.cl', 'dj@foo.cl', 'jan@baz.cz', 'd@abc.cl']

obtenerDominios(c)  se obtiene:
['abc.cl', 'alumnos.usm.cl', 'baz.cz', 'cn.de.cl', 'foo.cl',
'hao.cn', 'lala.de', 'lorem.ipsum.de', 'usm.cl', 'zi.cn']

contarTld(c) se obtiene:  {'cz': 2, 'de': 3, 'cn': 2, 'cl': 8}
```

Ejercicio 9:
Debido a la gran cantidad de crímenes y casos sin resolver, la policía local ha decidido implementar su propio sistema de reconocimiento de sospechosos con la técnica basada en el uso del DNA.

Para esto la policía mantiene dos listas de información; la primera contiene el nombre de las personas registradas en la región (nombre y apellido), la otra, un cromosoma representativo del DNA de cada una de esas personas.

Por simplicidad, un cromosoma se considera como una cadena de 0 (ceros) y 1 (unos), de largo 20.

El método para determinar el sospechoso, es el siguiente:

- Se obtiene una muestra del cromosoma del autor del delito (20 caracteres)
- Se busca en la lista de cromosomas, aquel cromosoma que esmás parecido a la muestra. El más parecido se define como el cromosoma que tiene más genes (caracteres) iguales a la muestra.
- Al terminar la búsqueda, se muestra el nombre de la persona cuyo cromosoma es sospechoso, con el porcentaje de parentesco.

La información inicial del programa se debe ingresar directamente en el código, es decir, nombres y cromosomas, en cambio la secuencia encontrada en la escena del crimen, debe ser ingresada por el usuario.

Desarrolle un programa que lleve a cabo la búsqueda descrita a partir de una muestra de entrada.

Consideremos, personas como Pedro, Juan y Diego. Sus secuencias serán

- 00000101010101010101
- 00101010101101110111
- 00100010010000001001

Ingrese secuencia: 01010101000011001100

El culpable es Pedro con un parentesco de 60%

Ejercicio 10:
En el juego Scrabble™, cada letra tiene asociado un valor numérico. El puntaje total de una palabra es la suma de los valores numéricos de cada letra. Las letras más comunes tienen menor valor que las letras menos comunes. Los puntos asociados a cada letra se muestran en la siguiente tabla:

puntos	**letras**
1	A,E,I,L,N,O,R,S,T, U
2	D,G
3	B,C,M,P
4	F,H,V,W,Y
5	K
8	J,X
10	Q,Z

Escribir un programa que calcule y muestre el puntaje de una palabre. Crear un diccionario para almacenar la tabla anterior.