COLEGIO LA CRUZ
SEMPER ALTIUS
Departamento de Matemática

EJERCICIOS – LISTAS PARTE 2

Pensamiento computacional y programación

---

0. En español al escribir una secuencia de elementos usualmente separamos cada uno por una coma. Además, separamos el último elemento con un "y" en vez de una coma, a menos que la secuencia tenga un solo elemento. Escribe un programa que reciba una lista de strings y retorne un string que contenga todas las palabras de la lista usando el formato mencionado al comienzo. Por ejemplo:

Input	Output
[manzanas]	manzanas
[manzanas, naranjas]	manzanas y naranjas
[manzanas, naranjas, plátanos, limones]	manzanas, naranjas, plátanos y limones

1. Para ganar el premio mayor en un juego de lotería se deben acertar 6 números escogidos entre 1 y 49 sin repetirlos. Escribe un programa que pida al usuario 6 números entre 1 y 49. Luego, tu programa debe generar la combinación ganadora y mostrar en pantalla: los números jugados por el usuario, la combinación ganadora y la cantidad de aciertos. Debes asegurarte que los números ingresados por el usuario sean válidos y que la combinación ganadora no tenga números repetidos.

2. Una baraja de cartas inglesa común contiene 52 cartas. Cada carta se puede representar usando dos caracteres, el primero indicando el valor de la carta y el segundo, su pinta. Para representar el valor usamos los números del 2 al 9, una T para el 10 y J, Q, K, A, para la Jota, Reina, Rey y As respectivamente. Para las pintas usaremos t, c, p, d para representar trébol, corazón, pica y diamante, respectivamente. Por ejemplo:

Carta	Representación
Jota de trébol	Jt
Diez de diamante	Td
7 de corazón	7c

Para este ejercicio debes comenzar escribiendo una función llamada *crearMazo*, la cual debe usar loops para crear y retornar el mazo de cartas completo, guardando en una lista las abreviaciones de cada carta. Esta función no debe recibir ningún parámetro.

Luego escribe una función llamada *barajar* , la cual recibe como parámetro una lista de cartas y retorna la baraja revuelta. Un modo de hacerlo es ir carta por carta y cada una cambiarla con otra al azar. Lo importante es que debes escribir tu propio código para barajar, sin usar el comando shuffle.

Opcionalmente, puedes imprimir en pantalla la baraja ordenada que creaste al comienzo y el resultado de barajarla.

3. En muchos de juegos de cartas se debe repartir una mano de cartas a cada jugador, luego de que el mazo ha sido barajado. Escribe una función de nombre *repartir* que reciba tres parámetros: el número de manos a repartir, la cantidad de cartas por mano y un mazo de cartas. Tu función deberá retornar **una** lista que contenga todas las manos que fueron repartidas, cada una representada a su vez por una lista.
   Al repartir las manos, tu función debe modificar el mazo de cartas recibido como parámetro y eliminar aquellas que se han repartido. Además, tu función deberá entregar una carta a cada jugador antes de entregar otra al mismo jugador.

4. Escribe un programa que utilice el código que escribiste en los ejercicios 2 y 3 que realice las siguientes acciones:
   - Cree un mazo de cartas
   - Baraje el mazo 3 veces
   - Reparta 4 manos de 5 cartas cada una
   - Muestre en pantalla las 4 manos de cartas
   - Muestre en pantalla las cartas que han quedado en el mazo

5. Una sub lista es una lista que forma parte de otra lista más grande. Una sub lista puede contener uno, varios o ningún elemento. Por ejemplo [1], [2], [3], y [4] son sub listas de [1, 2, 3, 4]. La lista [2, 3] también es una sub lista de [1, 2, 3, 4], pero [2, 4] no lo es, por que el 2 y el 4 no van juntos en la lista más grande. Una sub lista vacía [] es sub lista de cualquier lista. También, una lista es sub lista de si misma, por ejemplo [1, 2, 3] es sub lista de [1, 2, 3].
   En este ejercicio debes crear una función llamada esSubLista, que determine si una lista es o no sub lista de otra. Tu función debe recibir dos listas, “larga” y “corta”, como parámetros, y retornar True si “corta” es sub lista de “larga” y False en caso contrario.

6. Usando el código anterior, crea un programa que imprima en pantalla una lista con todas las posibles sub listas de una lista. Por ejemplo, las sub listas de [1, 2, 3] son [], [1], [2], [3], [1, 2], [2, 3] y [1, 2, 3].

* Cada uno de los ejercicios de esta guía que realices de forma satisfactoria y cumpliendo todos los requerimientos propuestos, otorga 3pp. Si tu código presenta una solución novedosa/original o destaca por incluir funcionalidades que no se piden, puedes obtener como máximo 5pp.