Ejercicios de tercer parcialito

Del tercer parcialito del primer cuatrimestre de 2012

1. Escribir una función *reemplazar* que tome una Pila, un valor_nuevo y un valor_viejo y reemplace en la Pila todas las ocurrencias de valor_viejo por valor_nuevo. Considerar que la Pila tiene las primitivas apilar(dato), desapilar() y esta_vacia().

Del recuperatorio del tercer parcialito del primer cuatrimestre de 2012

2. Escribir un método que invierta una ListaEnlazada utilizando una Pila como estructura auxiliar y considerando que lista solo tiene una referencia al primer nodo.

Del tercer parcialito del primer cuatrimestre de 2013

3. Escribir una función que reciba una pila de números y elimine de la misma los elementos consecutivos que están repetidos. Se pueden usar estructuras auxiliares. La función no devuelve nada, simplemente modifica los elementos de la pila que recibe por parámetro.
Por ejemplo: remover_duplicados_consecutivos(Pila([2, 8, 8, 8, 3, 3, 2, 3, 3, 3, 1, 7])) Genera: Pila([2, 8, 3, 2, 3, 1, 7]).

Del segundo recuperatorio del tercer parcialito del primer cuatrimestre de 2013

4. Para una lista simplemente enlazada de números (que solo mantiene una referencia al primer nodo) implementar la primitiva suma_acumulativa() que devuelva una nueva lista (del mismo largo) tal que el nodo i de la nueva lista contenga la suma acumulativa de los elementos de la lista origina hasta el nodo i.

Del segundo recuperatorio del tercer parcialito del primer cuatrimestre de 2014

5. Escribir una función *reducir* que reciba por parámetro una cola y una función f de dos parámetros, y aplique sucesivamente la función f a los dos primeros elementos de la cola (luego de desencolarlos) y encole el resultado, hasta que solo quede un elemento. La función *reducir* debe devolver el único elemento restante en la cola.

Del primer recuperatorio del tercer parcialito del primer cuatrimestre de 2014

6. Escribir una función que reciba una cola y la cantidad de elementos en la misma, y devuelva True si los elementos forman un palíndromo o False si no.
Por ejemplo:
`es_palindromo([n, e, u, q, u, e,n], 7)` $->$ True.

Del primer recuperatorio del tercer parcialito del primer cuatrimestre de 2013

7. Escribir una funcion recursiva que reciba una cadena y devuelva True si es un palındromo (se lee igual hacia adelante que hacia atras) o False si no lo es.

Del cuarto parcialito del primer cuatrimestre de 2012

8. a) ¿Como deberıa ser un arreglo con numeros del 1 al 10 para obtener la peor performance en una implementacion de quicksort que elige siempre como pivote al ultimo elemento de la lista en vez del primero? Justifique.

b) ¿Que metodo de ordenamiento elegirıa para ordenar ascendentemente un arreglo que ya esta ordenado pero en forma descendente? ¿Por que?. Asumiendo que fueran muchos elementos, ¿elegirıa este metodo de ordenamiento o utilizarıa una funcion para invertirlos in- place (en el mismo arreglo)?

Del recuperatorio del quinto parcialito del segundo cuatrimestre de 2008

9. Escribir una funcion merge sort 3 que implemente un algoritmo similar al de merge sort pero que en lugar de dividir los valores en dos partes iguales, los divida en tres. Considerar que se cuenta con la funcion merge(lista 1, lista 2, lista 3)). ¿Como te parece que se va a comportar este metodo con respecto al merge sort original?

Del primer recuperatorio del quinto parcialito del primer cuatrimestre de 2009

10. Escribir una funcion recursiva que reciba una lista y un parametro n, y devuelva otra lista con los elementos de la lista replicados esa cantidad n de veces. Por ejemplo, replicar ([1, 3, 3, 7], 2) debe devolver ([1, 1, 3, 3, 3, 3, 7, 7]) .

Del quinto parcialito del primer cuatrimestre de 2010

11. a) Mostrar paso a paso como funcionarıan los algoritmos de insercion y quicksort sobre la siguiente lista de numeros: [9, 12, 1, 4, 2, 8, 15, 21]

b) ¿Cual de los dos algoritmos es mas eficiente para ordenar una gran cantidad de elementos? ¿Por que?

Ejercicios extra de recursividad

12. Escribir una funcion recursiva en Python que ordene una Pila (la cual contiene unicamente numeros) de menor a mayor (quedando el elemento mas grande en el tope).

13. Escribir una funcion recursiva en Python que cuente la cantidad de apariciones de un elemento en una lista recibidos por parametro.