Algoritmos y Programación I (75.40) – Curso Wachenchauzer – 1.er parcialito - 1.er recuperatorio – 29/06/2015

Resolvé los siguientes problemas en forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Si te parece que los comentarios no son suficientemente explicativos, podés agregar una descripción de funcionamiento del código. Podés escribir tantas funciones auxiliares como creas necesarias.

1. Escribir una función que dado un número entero `n` mayor a 0 y una lista de números enteros, devuelva una nueva lista con los divisores de `n` que se encuentren en la primera. Si `n` no cumple las condiciones pedidas, debe devolver una lista vacía.

   Ejemplo: Se recibe `(8, [1, 7, 2, -4, 6, 9])`, debe devolver `[1, 2, -4]`

2. Escribir un programa que le pida al usuario que ingrese letras de a una, hasta que ingrese una cadena vacía. Si ingresa algo que no es una letra, o ingresa más de una, debe ignorarse esta entrada. Luego debe imprimir todas las letras ingresadas en orden alfabético.

   Ejemplo: Se ingresa `g, n, a, v, n, p, n`; se debe imprimir `agnnnpv`.

3. a) Hacer el seguimiento paso por paso de la búsqueda del número `5` en la lista `[1,2,4,5,6,8,9,11,14]`, utilizando el algoritmo de búsqueda binaria.

   b) ¿Qué condiciones debe cumplir sí o sí una lista para que se pueda realizar una búsqueda binaria sobre ella?

   c) Si una lista cumple las condiciones del punto (b) y tiene elementos repetidos, ¿puede aplicarse igual una búsqueda binaria sobre ella?

¡Suerte! :-)

Algoritmos y Programación I (75.40) - Curso Wachenchauzer - 1º Parcialito - 20/04/15

Resolvé los siguientes problemas en forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Si te parece que los comentarios no son suficientemente explicativos, podés agregar una descripción del funcionamiento del código. Podés escribir tantas funciones auxiliares como creas necesarias.

1) Una *ruta* es una cadena donde se indica la ubicación de un archivo. Cada carpeta se separa por un carácter `'/'`, y al final se encuentra el nombre del archivo con su extensión luego del carácter `'.'`. Ejemplo: `'/home/user/tp1.py'`; siendo la extensión `'py'`.

   Realizar una función que reciba una ruta y una lista de extensiones válidas, y verifique que la extensión del archivo corresponda a alguno de los formatos admitidos.

   Ejemplo:
   `validar_extension('/home/user/listado.txt', ['mp3', 'wav', 'mpeg'])` → `False`

2) Escribir una función que reciba un número entero n, y devuelva una matriz triangular superior de dimensión nxn cuyos elementos son números aleatorios del 0 al 9. Por ejemplo, para $n = 4$ debe devolver la siguiente lista de listas:

```
[[3,6,2,6],
 [0,5,9,1],
 [0,0,6,8],
 [0,0,0,2]]
```

   Ayuda: para generar números aleatorios en un rango determinado se puede utilizar la función `randint(a, b)` del módulo random (devuelve un número aleatorio `x` tal que $a \leq x \leq b$).

3) Escribir un programa que pida al usuario que ingrese el valor de un ángulo en grados, entre 0 y 360, e imprima su conversión a radianes. Si el usuario no ingresa un número válido, se le debe pedir que lo ingrese nuevamente (y repetir hasta que el usuario ingrese un número válido).

Suerte! :-)

Algoritmos y Programación I (75.40) – Wachenchauzer – 1.er recuperatorio 2.º parcial – 29/6/2015

Resolvé los siguientes problemas en forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Si te parece que los comentarios no son suficientemente explicativos, podés agregar una descripción de funcionamiento del código. Podés escribir tantas funciones auxiliares como creas necesarias.

1. Se tiene un diccionario con las fechas de nacimiento de un conjunto de personas. Las fechas están en formato `DD/MM/YYYY`.

   Ejemplo: `{"Juan": "27/05/1993", "Alicia": "30/11/1990", "Elena": "27/05/1985"}`

   Se pide escribir una función que tome como parámetro este diccionario y agrupe en una lista las personas que celebran su cumpleaños el mismo día.

   La función debe devolver un diccionario en el que las claves son fechas en formato `DD/MM`, y los valores listas de nombres.

   Ejemplo: `{"30/11": ["Alicia"], "27/05": ["Juan", "Elena"]}`

2. Se pide implementar una clase `CalendarioMes` con los siguientes métodos:

   - `__init__()`: toma como parámetro un entero que representa el número de días del mes (entre 28 y 31).
   - `agregar_evento()`: toma como parámetro un día (número entero) y el nombre de un evento (cadena) y lo almacena en el calendario.
   - `eliminar_evento()`: toma como parámetro un día y el nombre de un evento y lo elimina del calendario. La función debe devolver `True` si se pudo borrar el evento, o `False` si no se encontró el evento en la fecha indicada.
   - `obtener_eventos_dia()`: toma como parámetro un día y devuelve una lista con los eventos programados para ese día, o la lista vacía si no hay eventos en ese día.

   Si algún parámetro no es válido, se debe lanzar la excepción `ValueError`.

3. Se tiene un archivo CSV de cuatro columnas, "Equipo", "Jugador", "Fecha" y "Goles". El archivo está ordenado por equipo y, dentro de cada equipo, por jugador.

   Se pide implementar una función que reciba el nombre del archivo como parámetro, e imprima el total de goles de cada jugador, y de cada equipo.

Algoritmos y Programación I (75.40) – Curso Wachenchauzer – 2.° parcialito – 18/05/2015

Resolvé los siguientes ejercicios de forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Podés escribir tantas funciones auxiliares como creas necesarias.

1. Se pide escribir una función que calcule el ganador de una liga de fútbol a partir de los resultados de los partidos realizados. La función recibe una lista de tuplas de 4 elementos con el siguiente formato:

   ```
   (nombre_equipo1, goles_equipo1, nombre_equipo2, goles_equipo2)
   ```

   Ejemplo: `("Barcelona", 1, "Real Madrid", 2)`.

   La función devuelve el nombre del equipo ganador. En caso de haber más de un equipo con puntuación máxima, devolver uno cualquiera de entre ellos.

   Reglas de puntaje de la liga: el equipo ganador de un partido recibe 2 puntos, y el perdedor 0 puntos. En caso de empate, cada uno de los dos equipos recibe 1 punto.

2. Se tiene una lista de los alumnos de una materia, y se desea dividirlos en dos grupos de prácticas: grupo 1 para los alumnos con padrón impar, grupo 2 para los alumnos con padrón par.

   La lista de alumnos se encuentra en un archivo en formato CSV. Se desconoce el número de columnas, pero se sabe que la primera columna es siempre el padrón, y que no hay cabecera.

   Se pide escribir una función que reciba como parámetro el nombre del archivo de alumnos y escriba dos archivos, `grupo1.txt` y `grupo2.txt`, con la lista de padrones correspondientes, uno por línea.

   Si ocurre un error al abrir cualquiera de los archivos, la función debe imprimir el detalle del error y finalizar su ejecución. Ante cualquier otra situación de error, la función debe permitir que el error se propague mediante el mecanismo estándar de excepciones.

3. Se pide implementar una clase *Cuenta* que se comporte de la siguiente manera:

   ```
   >>> c = Cuenta("Adolfo Suarez")
   >>> c.depositar(100)
   >>> c.extraer(60)
   >>> c.saldo()
   40
   ```

   ```
   >>> c.extraer(50)
   Traceback (most recent call last):
       ...
   ValueError: fondos insuficientes
   >>> print c
   Cuenta de Adolfo Suarez: $40 de saldo
   ```

Algoritmos y Programación I (75.40) – Wachenchauzer – 3.er parcialito - 1.er recuperatorio – 03/07/2015

Resolvé los siguientes problemas en forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Si te parece que los comentarios no son suficientemente explicativos, podés agregar una descripción de funcionamiento del código. Podés escribir tantas funciones auxiliares como creas necesarias.

1. Teniendo una `ListaEnlazada` con los métodos `append`, `pop`, `insert` y `remove` se quiere implementar la clase `ListaEnlazadaOrdenada`. Esta clase tiene sólo los métodos `insert` y `remove`, y asegura mantener ordenados los elementos mientras se agregan. Una posible implementación sugiere heredar el nuevo tipo de la `ListaEnlazada` construida en clase. Responder sin implementar:

   a) ¿Qué desventaja tiene la estrategia propuesta?
   b) Proponer una solución alternativa que permita reutilizar la clase ya programada. Explicar por qué esta solución no tiene los problemas del diseño anterior.

2. Implementar el método `eliminar_posiciones` para la `ListaEnlazada` que reciba una secuencia de números ordenados y sin repeticiones (por ejemplo, la tupla `(0, 2, 6, 8)`), y elimine los elementos de la lista enlazada en dichas posiciones, recorriendo la lista enlazada una única vez. Si la secuencia de índices a eliminar contiene una posición no válida se deberá lanzar una excepción.

   Ejemplos:

   ```
   [ a b c d e ] → l.eliminar_posiciones([1, 3]) → l = [ a c e ]
   [ a c e ] → l.eliminar_posiciones([0, 2]) → l = [ c ]
   ```

3. Implementar el método `distribuir_cola(k)` para la clase `Cola` que reciba por parámetro un número entero `k` mayor a 1, cuyo valor debe ser validado. Este nuevo método debe devolver `k` nuevas colas con los elementos de la original distribuidos de forma alternada, respetando el orden relativo de los elementos.

   Ejemplos (en todos los casos el primer elemento de la cola en desencolar es el primero de izquierda a derecha):

   ```
   c: [ a b c d e f g ] → c.distribuir_cola(3) → c1 = [ a d g ], c2 = [ b e ], c3 = [ c f ]
   c: [ a b c ] → c.distribuir_cola(4) → c1 = [ a ], c2 = [ b ], c3 = [ c ], c4 = [ ]
   ```

¡Suerte! :-)

Algoritmos y Programación I (75.40) – Wachenchauzer – 3.er parcialito - 2.do recuperatorio – 06/07/2015

Resolvé los siguientes problemas en forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Si te parece que los comentarios no son suficientemente explicativos, podés agregar una descripción de funcionamiento del código. Podés escribir tantas funciones auxiliares como creas necesarias.

1. Implementar la clase `ListaDeEnteros` que herede de `ListaEnlazada` y tenga su misma funcionalidad, excepto que los métodos `append(elemento)` e `insert(posición, elemento)` deben verificar que el elemento a agregar sea un entero usando `isinstance(elemento, int)`. Si no se verifica esta condición se debe lanzar una excepción de tipo `ValueError`.

   Considerar que la `ListaEnlazada` está implementada sólo con una referencia al primer nodo.

2. Implementar el método `merge()` para la `ListaEnlazada` que recibe otra `ListaEnlazada` por parámetro. Con la precondición de que ambas tienen a todos sus elementos ordenados, el método debe devolver una nueva lista enlazada que contenga a los elementos de las dos en orden.

   Considerar que la `ListaEnlazada` está implementada sólo con una referencia al primer nodo.

3. Usando una Pila, implementar una función que verifique si una cadena de paréntesis y corchetes es correcta.

   Ejemplos:

   `'[ ( ) [ ] ] ( )' → True`

   `'[ ( ] [ ] ] ( )' → False`

   `'[ ] ( ) [ ] ] ( )' → False`

   `'[ ] ( ) [ ] ( )' → True`

¡Suerte! :-)

Algoritmos y Programación I (75.40) – Curso Wachenchauzer – 3.er parcialito – 05/06/2015

Resolvé los siguientes problemas en forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Si te parece que los comentarios no son suficientemente explicativos, podés agregar una descripción de funcionamiento del código. Podés escribir tantas funciones auxiliares como creas necesarias.

1. Se desea modelar a los personajes de un juego de rol. Cada `Personaje` tiene una cantidad de vida y de daño de ataque (sendos valores numéricos entre 0 y 100). Estos atributos deben ser establecidos en el constructor. Todos los Personajes deben implementar un método `atacar` que recibe como parámetro a otro `Personaje` al que le descontará de su vida el daño de ataque. Cuando uno es atacado y llega a 0 de vida, se considera que está muerto y no podrá realizar ninguna actividad en el juego.

   Implementar la clase `Druida` con el método `curar` que reciba una lista de `Personajes` y les restaure la vida al valor inicial correspondiente. Implementar la clase `Vampiro` que tiene un comportamiento diferente de ataque: además de descontar vida al `Personaje` atacado, recupera 15 puntos de vida.

   Crear todos los métodos auxiliares que creas necesarios para lograr un buen diseño orientado a objetos. Realizar, además, todas las validaciones necesarias usando excepciones.

2. Implementar el método `downsample(k)` para la `ListaEnlazada`. Este método debe eliminar todo elemento de la lista que ocupe una posición que no sea múltiplo del número `k` pasado por parámetro, que debe ser validado (`k > 1`).

   Ejemplos:

   ```
   l: [0, 1, 2, 3, 4, 5] → l.downsample(2) → l = [0, 2, 4]
   l: ['a', 'b', 'c', 'd', 'e', 'f', 'g'] → l.downsample(4) → l = ['a', 'e']
   ```

3. Sabiendo que se cuenta con una clase `Cola` con las primitivas `encolar`, `desencolar` y `esta_vacia`, implementar la clase `ColaConPrioridad`. Este nuevo tipo debe tener los métodos `encolar(elemento)`, `encolar_prioritario(elemento)`, `desencolar()` y `esta_vacia()`; y al desencolar debe priorizar aquellos elementos que fueron encolados con prioridad; es decir, no deben salir elementos comunes de la estructura si no salieron previamente todos los elementos con prioridad. ¿Qué tipo de estrategia de diseño utiliza? ¿Por qué?

¡Suerte! :-)

Algoritmos y Programación I (75.40) – Curso Wachenchauzer – 4.to parcialito – 26/06/2015

Resolvé los siguientes problemas en forma clara y legible, respetando sangrías e incluyendo la documentación necesaria. Si te parece que los comentarios no son suficientemente explicativos, podés agregar una descripción de funcionamiento del código. Podés escribir tantas funciones auxiliares como creas necesarias.

1. Escribir en C la función `int obtener_valor(const int vector[], int len, int pos)`. La función debe devolver el valor que se encuentra en `vector[pos]`, interpretando `pos` como en Python. Es decir, `pos` puede tomar valores entre `-len` y `len - 1`; y para los valores negativos busca los elementos comenzando desde la última posición del vector. Si `pos` no es válida, devolver la constante `INT_MIN` (asumir que la constante ya fue declarada).

2. La suma digital de un número `n` es la suma de sus dígitos. Escribir la función recursiva `suma_digital(n)` que recibe un número entero positivo y devuelve su suma digital. No se permite convertir el número a cadena. Ejemplo: `suma_digital(2019) -> 12` (porque `2+0+1+9 = 12`).

3. Mostrar los pasos del ordenamiento de la lista: `[6, 1, 4, 7, 2, 0, 3, 5]` con Quicksort. En cada paso tomar como pivote el primer elemento del arreglo.

¡Suerte! :-)

Ejercicios de primer parcialito
Taller 15/04/15

Del primer parcialito del primer cuatrimestre de 2009

1. Escribir una función que reciba una lista de cadenas y devuelva la cadena de menor orden alfabético y su posición. Por ejemplo, si se recibe: [“Luis”, “Alberto”, “Spinetta”] debe devolver (“Alberto”, 1).

Del segundo parcialito del primer cuatrimestre de 2010

2. Escribir una función que reciba una cadena con palabras y devuelva una lista con solo aquéllas palabras que son palíndromas. Por ejemplo, si se recibe la cadena “neuquen es para alla”, debe devolver [“neuquen”, “alla”].

Del primer parcialito del primer cuatrimestre de 2011

1. Un número refactorizable o número tau es un número natural n que es divisible por el número de divisores que tiene; por ejemplo, el número 18 es refactorizable porque sus divisores son seis: 1, 2, 3, 6, 9, 18 y el número 6 está entre ellos. Escribir una función que reciba un número y devuelva True si es refactorizable o False si no lo es.

Del primer recuperatorio, segundo parcialito del primer cuatrimestre de 2011

3. Escribir una función que reciba una lista de cadenas, y devuelva la cadena más larga de la lista. En caso de que la lista esté vacía debe devolver una cadena vacía.

Del primer parcialito del primer cuatrimestre de 2013

1. Escribir una función que reciba por parámetro una lista de números y devuelva otra lista con los números de la ingresada que terminan en cero. Por ejemplo, si se recibe la lista: [4, 23, 40, -7, 0, 14, 1000, -760] debe devolver [40, 0, 1000, -760].

2. Escribir una función que reciba por parámetro un número y le pida números al usuario hasta que la suma de los números ingresados sea mayor al número recibido. La función debe devolver una lista con los números ingresados.

Ejercicios de segundo parcialito

Del segundo parcialito, primer cuatrimestre de 2013

1. Escribir la clase Polinomio, utilizando una lista para guardar los coeficientes del mismo. Implementar los métodos:

- __init__: Constructor. Recibe una lista con los coeficientes.
- __str__: Devuelve una representación de cadena del polinomio, por ejemplo: "3x^3 - x + 1".
- __eq__ : Recibe otro Polinomio y devuelve True si son iguales o False si no.
- derivar: Devuelve un nuevo Polinomio que es la derivada del actual.

Del segundo recuperatorio del segundo parcialito, primer cuatrimestre de 2013

2. Escribir un programa que abra un archivo separado por comas y le pida al usuario dos números. El programa debe escribir en otro archivo el contenido del primero con las columnas del archivo CSV intercambiadas según los números del usuario. Por ejemplo, si una linea del archivo es "lunes, martes, miércoles, jueves" y los números del usuario son 1 y 3, el archivo de salida será "miércoles, martes, lunes, jueves" (las columnas 1 y 3 están intercambiadas).

Del segundo parcialito, primer cuatrimestre de 2012

3. Un grupo de amigos quiere ponerse de acuerdo para realizar un asado, para ello cuenta con un diccionario con los nombres de los amigos como claves, y como valor una tupla con los días del mes (1 a 31) en los que cada uno podría asistir. Implementar una función que, dado un diccionario de esa forma, devuelva una lista con los días del mes en los que todos podrían ir al asado.

Del tercer parcialito, primer cuatrimestre de 2012

4. Implementar una clase Colectivo con los siguientes métodos:

- Un constructor, que crea un Colectivo vacío.
- Un método subir, que dado un destino (en forma de cadena) y un objeto de la clase Persona, lo agrega al Colectivo.
- Un método bajar, que dado un destino, saca del Colectivo a las Personas que tienen dicho destino asignado, devolviéndolas en una lista.

Del tercer parcialito, primer cuatrimestre de 2010

5. Escribir una función que reciba un nombre de un archivo con datos y un nombre de un archivo para guardar errores (los errores se agregan al final). El archivo con datos tiene el siguiente formato: año,mes,vendedor,cliente,monto. La función debe devolver un diccionario con los vendedores como clave y los montos totales por vendedor como valor. Cuando una línea no sea válida, la ejecución no se puede detener, pero debe guardar la línea en el archivo de errores.

6. Escribir una función que reciba un diccionario con vendedor $\rightarrow$ monto, otro diccionario con vendedor $\rightarrow$ comisión, y un archivo en el cual grabar los datos. La función debe validar que todos los vendedores del primer diccionario se encuentren en el segundo diccionario, y de no ser así levantar una excepción; calcular el monto que le corresponde a cada uno (monto*comision), y guardarlo en el archivo de salida como vendedor:monto.

Ejercicios de los temas del 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 ésta sólo 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 estén 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]).

4. Implementar un Iterador para la clase ListaEnlazada que saltee los nodos cuyos datos sean de largo 0 o no esté definido.
Aclaración: La función len no está definida para todos los tipos de datos y en esos casos lanza TypeError. Escribir y utilizar una función auxiliar llamada "largo" que siempre devuelva un número.

Del segundo recuperatorio del tercer parcialito del primer cuatrimestre de 2013

5. Para una lista simplemente enlazada de números (que sólo 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

6. 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 sólo quede un elemento. La función "reducir" debe devolver el único elemento restante en la cola.

Ejercicios de los temas del cuarto parcialito

Del parcialito del primer cuatrimestre de 2008

1) Escribir una función que calcule recursivamente cuántos elementos hay en una pila sin alterar su contenido. Suponer que la pila sólo tiene los métodos apilar y desapilar.

Del recuperatorio del quinto parcialito del segundo cuatrimestre de 2008

2) Escribir una función `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 función `merge(lista_1, lista_2, lista_3)`. ¿Cómo te parece que se va a comportar este método con respecto al merge sort original?

Del cuarto parcialito del primer cuatrimestre de 2012

3) a) ¿Cómo debería ser un arreglo con números del 1 al 10 para obtener la peor performance en una implementación de quicksort que elige siempre como pivote al último elemento de la lista en vez del primero? Justifique.

b) ¿Qué método de ordenamiento elegiría para ordenar ascendentemente un arreglo que ya está ordenado pero en forma descendente? ¿Por qué? Asumiendo que fueran muchos elementos, ¿elegiría este método de ordenamiento o utilizaría una función para invertirlos in-place (en el mismo arreglo)?

Del recuperatorio del tercer parcialito del primer cuatrimestre de 2013

4) Escribir una función recursiva que reciba una cadena y devuelva True si es un palíndromo (se lee igual hacia adelante que hacia atrás) o False si no lo es.

Del parcialito del primer cuatrimestre de 2013

5) Escribir en C un programa que le pida un número entero al usuario y muestre por pantalla si el número ingresado es un número primo o no.

Del segundo recuperatorio del tercer parcialito del primer cuatrimestre de 2013

6) Escribir en forma recursiva una función que reciba una lista de números y devuelva el valor del elemento máximo.

Del segundo recuperatorio del cuarto parcialito del primer cuatrimestre de 2013

7) Escribir en C un programa que solicite al usuario el ingreso de diez números enteros e imprima el mínimo, el máximo y el promedio.