Facultad de Ciencia

Departamento de Matemática y Ciencia de la Computación

Paradigmas de programación

Semestre 2020-01

Ejercicios de ayudantía

Nicolás Honorato Droguett
Matías Hurtado Carrasco

Pablo Perez Lantero
12 de septiembre de 2020

Índice

1 2020-05-11

1.1 Ayudantía Nicolás

Ejercicios

Pregunta 1 Programe en `Python` la función `bloques(g,key)` donde `g` es un generador de elementos, y `key` es una función de un argumento que recibe elementos de `g` y devuelve valores comparables entre sí usando los operadores habituales de comparación. La función genera los bloques de elementos consecutivos generados por `g` que están ordenados ascendentemenre de acuerdo al valor de `key`. Cada vez que se rompe el orden se genera un nuevo bloque. Cada bloque es generado como una lista. Por ejemplo, si `g` genera los elementos `1,5,6,4,8,9,2,3,7` en este orden y `key` es la función identidad, entonces se generarán los bloques `[1,5,6],[4,8,9],[2,3,7]` en este mismo orden. Tenga presente en la implementación que `g` podría generar infinitos elementos, o una cantidad finita pero extremadamente grande.

Pregunta 2 La función `merge(A,B)` recibe dos listas `A` y `B` de números, cada una ordenada ascendentemente, y devuelve una tercera lista con la mezcla ordenada de `A` y `B`. Usando la función `merge`, realice dos implementaciones distintas de la función `mega_merge(*L)` que recibe varias listas de números como parámetros, cada lista ordenada, y devuelve la mezcla de todas las listas. La primera implementación debe usar la función `reduce`, mientras que la segunda implementación no puede usarla.

1.2 Ayudantía Matías

Ejercicios

Pregunta 1 Implemente en `Python` la función `merge(A,B,comp=cmp)`, pero en lugar de devolver explícitamente la mezcla en una lista, devuelve una generación de (o define una forma de generar o iterar) los elementos de la mezcla.

Pregunta 2 Implemente en `Python` la función `modulo(A,k)` que recibe una lista `A` y un entero positivo `k`, y devuelve una lista compuesta por `k` listas. La lista número `i` de la lista resultado contiene de la lista original los elementos que se encuentran en las posiciones que dejan resto `i` en la división por `k`. Por ejemplo, para `A = [0,1,2,3,4,5,6,7,8,9,10]` y `k = 3`, la función debe devolver la lista `[[0,3,6,9], [1,4,7,10], [2,5,8]]`.

2 2020-05-18

2.1 Ayudantía Nicolás

Ejercicios

Pregunta 1 La **sucesión de Thue-Morse** es una sucesión de números binarios que al concatenarlos se produce una secuencia con segmentos iniciales alternos. Implemente en `Python` una función que entregue los n primeros números de la sucesión, siendo n un **parámetro opcional**. La secuencia comienza con los números 0, 01, 0110, 01101001, ..., si tomamos t_n como el binario n de la sucesión, el binario t_{n+1} es una **concatenación** de la forma $t_n + C_1(t_n)$, tal que $C_1(x)$ es la función complemento 1.

Pregunta 2 Implemente en `Python` una función que reciba una matriz `M` y entregue la sub-matriz de suma máxima de `M`. Muestre el valor de la suma.

2.2 Ayudantía Matías

Ejercicios

Pregunta 1 Implemente en `Python` un generador `fun_stream(s,e)`, que devuelva el los resultados de la ecuacion `fun(x)= 2*`x^3 `+ 3*`x^2 `+ 4` para todos los naturales entre `s` y `e` incluyente, utilizando el generador `exp_stream(s,e,exp)` que entrega x^{exp} siendo x todos los naturales entre `s` y `e` incluyente.

Pregunta 2 Implemente en `Python` la función `recorrer(M)` donde `M` es una matriz rectangular llena de 1 y 0, la funcion recorrer debe generar un camino desde la posición `(0,0)` de la matriz `M` hasta la esquina contraria, con los 0 como camino valido y los 1 no, el orden de preferencia de movimiento es `(x,y)->(x+1,y+1)` , `(x,y)->(x+1,y)`, `(x,y)->(x,y+1)`

$$\begin{bmatrix} \$ & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} \$ & \$ & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} \$ & \$ & 0 & 0 & 0 & 0 \\ 1 & 1 & \$ & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \overset{*}{\rightarrow} \begin{bmatrix} \$ & \$ & 0 & 0 & 0 & 0 \\ 1 & 1 & \$ & \$ & 0 & 0 \\ 0 & 0 & 0 & 1 & \$ & 0 \\ 0 & 0 & 0 & 1 & 0 & \$ \\ 0 & 0 & 0 & 1 & 0 & \$ \\ 0 & 0 & 0 & 1 & 0 & \$ \end{bmatrix}$$

3 2020-05-08

3.1 Ayudantía Nicolás

Ejercicios

Pregunta 1 Implemente en `Python` la función `paridad(M,i,j)` donde `M` es una matriz rectangular y `i,j` es una celda de la matriz. Entregue la matriz `M'`, que contiene $*$ por cada impar en `M` con la misma posición, y $-$ por cada par. Comience el cómpute desde la posición `i,j`.

$$\begin{bmatrix} 2 & 2 & 0 \\ 7 & 1 & 4 \\ 9 & 0 & 3 \end{bmatrix} \rightarrow \begin{bmatrix} - & - & - \\ * & * & - \\ * & - & * \end{bmatrix}$$

4 2020-08-03

4.1 Ayudantía Matías

Ejercicios

Pregunta 1 Implemente en `Python` clases con jerarquía para representar a un estudiante que hereda de una persona e implementar métodos para mostrar la informacion de los objetos.

5 2020-08-10

5.1 Ayudantía Nicolás

Ejercicios

Pregunta 1 Implemente en `Python` la estructura de datos para el siguiente problema utilizando clases: Una empresa de servicios de transporte, con nombre, dirección e email, posee una lista de vehículos, específicamente posee automóviles, buses y trenes. Se desea modelar tal estructura, tal que la lista de vehículos se almacene en una única lista. Especifique atributos únicos para cada tipo de vehículo y encapsule la lista. Entregue las siguientes implementaciones:

```
    class EmpresaTransporte():
        def trenConCapacidad(self,capacidad: int): #Entrega la lista con todos los trenes que
            poseen tal capacidad
        def autoDisponible(self): #Entrega la lista de autom\'oviles disponibles.
    class Vehiculo():
        def __str__(self): #Muestra la informacion del vehiculo
    class Automovil():
        def __eq__(self,other): #compara la igualdad de dos automoviles
```

5.2 Ayudantía Matías

Ejercicios

Pregunta 1 Implemente en `Python` clases con jerarquía para representar grafos dirigidos y no dirigidos e implementar métodos para determinar caminos entre vértices, determinar cuales conjuntos de grafos conexos hay y encontrar el camino continuo más largo.

6 2020-08-17

6.1 Ayudantía Nicolás

Ejercicios

Pregunta 1 Implemente en `Python` la clase `Interval` y la clase `IntervalSet`, solicitadas en el segundo trabajo del curso, para implementar los siguientes métodos:

```
    class IntervalSet():
        def from_random(cls,min,max,n):
        def max_container(self):
```

Donde `from_random(cls,min,max,n)` es un **método constructor** para IntervalSet, que genera `n` intervalos aleatorios entre [`min`,`max`]. El método `max_container(self)` entrega el intervalo C que contiene la mayor cantidad de intervalos dentro de él, que también pertenecen al set.

Observación: Implemente los atributos y métodos necesarios para implementar los dos métodos solicitados, apegándose a las reglas del trabajo 2.

6.2 Ayudantía Matías

Ejercicios

Pregunta 1 Implemente en `Python` un verificador de forma para las funciones solicitadas en el segundo trabajo del curso:

```
class Interval:
    def __init__(self, start, end):
        ...
class IntervalSet:
    def __init__(self):
        ...
    def add(self, interval):
        ...
    def remove(self, interval):
        ...
    def cover(self, interval):
        ...
    def free(self, interval):
        ...
    def max_subset(self):
        ...
class WeightedInterval:
    def __init__(self, start, end):
        ...
class WeightedIntervalSet:
    def __init__(self):
        ...
    def max_subset(self):
        ...
```

7 2020-08-31

7.1 Ayudantía Matías

Ejercicios

Pregunta 1 Implemente en `Scheme (add L)` tal que `add` es un procedimiento que recibe una lista de números `L` y devuelve la suma de estos. ¿Cómo se implementaría con y sin recursión de cola?, ¿De qué formas distintas se pueden conseguir métodos con similar objetivo?.

8 2020-09-07

8.1 Ayudantía Nicolás

Ejercicios

Pregunta 1 Implemente en `Scheme` la función `(rellenar S I)` tal que `S` es una lista de intervalos ordenada e `I` es un intervalo. La función inserta, de **ser posible**, a `I` dentro de `S`. Se asume que el intervalo `I` está entre dos intervalos como **máximo**. Si `I` se sobrepone a tales intervalos donde se rellanará el espacio, generar un intervalo nuevo tal que contemple toda el área.

8.2 Ayudantía Matías

Ejercicios

Pregunta 1 Implemente en `Scheme (overlap? ia ib)` tal que `overlap?` es un procedimiento que recibe 2 intervalos (listas de 2 números) ia e ib y devuelve el valor de verdad correspondiente a la aseveración "existe intersección entre ia e ib".

Pregunta 2 Implemente en `Scheme (filterlist condit lis)` tal que `filterlist` es un procedimiento que recibe un procedimiento de devolución booleana (ej: even?) `condit` y una lista de elementos, apropiados al input esperado por `condit`, `lis` y retorne la sublista maximal de `lis` de los elementos para los que se cumple:

```
> (condit? element_from_lis)
#t
```

¿Cómo se implementaría con y sin recursión de cola?