Problemes Programació Avançada – Curs 2018-19
Tema 0: Repàs de Fonaments de Programació

EXERCICIS DE FITXERS

Exercici 1 (nivell bàsic)

Implementeu una funció que es digui `analitzaFitxer` que rebi com a paràmetre el nom d'un fitxer de text i llegeixi el fitxer i retorni el número de línies del fitxer, el número de paraules i el número de caràcters totals del fitxer (sense comptar els espais en blanc).

Exercici 2 (nivell bàsic)

Implementeu una funció que es digui `analitzaFixer` que rebi com a paràmetre el nom d'un fitxer de text que conté números enters separats per espais en blanc i salts de línia i retorni una única llista que contingui quants números hi ha a cada línia del fitxer i la suma dels números de cada línia. A la llista resultat cada element ha de ser una altra llista amb dos elements, el primer amb el nº de valors a la línia i el segon amb la suma dels valors.

EXERCICIS DE LLISTES

Exercici 3 (nivell mig)

Donada una llista de nombres enters, que es passa com a paràmetre, escriure les funcions següents:

a) `sumaAcumulada`: retorna una llista amb la suma acumulada de la llista original a cada element, és a dir, una llista on el primer element sigui el mateix, el segon sigui la suma del primer i el segon, el tercer la suma dels tres primers elements i així successivament. Per exemple si la llista d'entrada és `[1,2,3,4]` la llista de sortida hauria de ser `[1,3,6,10]`
b) `factorialLlista`: retorna una nova llista amb el factorial de cadascun dels nombres de la llista original.
c) `primers`: retorna una nova llista amb tots els nombres primers de la llista original.
d) `eliminaDuplicats`: retorna una nova llista sense elements duplicats.
e) `binariADecimal`: suposant que la llista original només conté 0's i 1's representant un número binari (amb el dígit més significatiu a la primera posició de la llista), retorna el valor decimal corresponent.

Exercici 4 (nivell mig)

Donades dues llistes de nombres enters, que es passen com a paràmetre, escriure les funcions següents:

a) `interseccio`: retorna una nova llista amb la intersecció de les dues llistes, és a dir, una nova llista que contingui els elements de la primera llista que també estan a la segona, en el mateix ordre en què estan a la primera llista.
b) `unio`: retorna una nova llista amb la unió de les dues llistes, és a dir, una nova llista amb tots els elements de les dues llistes, però sense elements repetits, en el mateix ordre que tenien a les llistes originals, primer els de la primera llista i a continuació, els de la segona llista.
c) `multiplicacioLListes`: retorna una nova llista en què cada element ha de ser la multiplicació dels dos elements corresponents de les dues llistes. Si les dues llistes tenen mida diferent, la mida de la llista resultat ha de ser la mida de la més petita, fent només la multiplicació dels primers elements de la llista més gran que sí que tenen correspondència a l'altra llista.
d) `multiplicacioElements`: retorna una nova llista amb la multiplicació de cadascun dels elements de la primera llista per tots els elements de la segona llista. Els elements de la nova llista seran llistes i cada llista contindrà totes les multiplicacions d'un element de la primera llista amb tots els elements de la segona.

Exercici 5 (nivell mig) – EXERCICI AVALUABLE

Escriure una funció que es digui `masterMind` que simuli el funcionament d'una versió simplificada del joc del Master Mind. El joc del Master Mind consisteix en endevinar una combinació de colors amb un número màxim d'intents. Després de cada intent se'ns informa de quins colors de la combinació hem encertat.

En el nostre cas, el joc consistirà en endevinar una seqüència de 5 dígits del 0 al 9 sense valors repetits. La funció `masterMind` rebrà tres valors com a paràmetre:

- Una llista amb la combinació inicial de 5 dígits del 0 al 9.
- Una llista on cada element serà un dels intents per endevinar la combinació. Cada intent (element de la llista) serà una llista amb els 5 valors de la seqüència.
- El número màxim d'intents permesos per endevinar la combinació. Podeu suposar que a la llista d'intents hi haurà sempre com a mínim tants elements com el número màxim d'intents.

La funció haurà d'anar processant cadascun dels intents. Per cada intent es generarà una nova seqüència de 5 valors que indicaran quins valors hem encertat de forma exacta (amb un 1), quins valors apareixen a la combinació, però en una posició equivocada (amb un 0), i quins valors no apareixen a la combinació (amb un -1). Per exemple, si la seqüència generada de forma aleatòria és [2, 8, 3, 5, 4], si s'introdueix la seqüència [2, 4, 6, 8. 0] s'ha de generar la seqüència [1, 0, -1, 0, -1] per indicar que el 2 està a la posició correcta, el 4 i el 8 estan, però no a la seva posició correcta i 6 i el 0 no estan a la combinació a encertar.

Els resultats de processar tots els intents s'han de guardar en una llista que s'ha de retornar com a resultat de la funció. Si en algun moment s'encerta la combinació inicial, s'ha de parar sense acabar de processar la resta d'intents que quedin a la llista. Si arribem al número màxim d'intents sense encertar la combinació també hem de parar i deixar sense processar la resta d'intents que quedin a la llista.

EXERCICIS DE DICCIONARIS

Exercici 6 (nivell mig)

a) Escriure una funció que es digui `indexCaracter` que rebi com a paràmetre el nom d'un fitxer i retorni un diccionari que contingui, per cada caràcter de l'alfabet, una llista on el primer element és el nº de vegades que apareix el caràcter al fitxer i el segon element és una llista amb totes les paraules del fitxer que contenen aquest caràcter, sense repeticions. Si algun caràcter de l'alfabet no apareix al fitxer no ha de tenir cap entrada al diccionari. Abans de construir l'índex convertiu tots el caràcters a minúscules amb el mètode `lower` dels strings. Elimineu també els possibles signes de puntuació: ',', '.', ':', ';', '!', '?'. Ho podeu fer amb el mètode `replace` dels strings per substituir tots aquests símbols per espais en blanc abans de dividir el text en paraules. Indexeu només els caràcters de l'alfabet. Ignoreu dígits o qualsevol altre símbol especial.

b) Escriure una funció que es digui `maxParaulaCaracter` que rebi com a paràmetre el diccionari creat a l'apartat anterior, i retorni una llista què cada element de la llista contingui un caràcter de l'alfabet i la paraula més llarga en què apareix. Els elements d'aquesta llista han d'estar ordenats alfabèticament.

Exercici 7 (nivell mig)

Volem crear un diccionari de sinònims on, per cada paraula (clau) del diccionari, el valor que hi guardem és una llista de tots els seus sinònims.

a) Implementeu una funció per afegir un sinònim al diccionari amb la capçalera següent:

```
afegeixSinonim(diccionari, paraula, sinonim)
```

La funció ha d'afegir el sinònim que es passa com a paràmetre al diccionari. Si la paraula no existeix al diccionari, s'hi ha d'afegir amb el seu sinònim. Si la paraula ja existeix al diccionari, el nou sinònim s'afegeix al final de la llista de sinònims actual de la paraula. Si el sinònim ja existeix dins de la llista de sinònims de la paraula, no s'ha de tornar a afegir.

b) Implementeu una funció que agafi com a entrada un diccionari de sinònims com el que es crea a l'apartat anterior i una llista amb paraules i retorni una nova llista transformada utilitzant el diccionari de sinònims de la forma següent:

- Per les paraules de la llista original que apareixen al diccionari hem de seleccionar aleatòriament un dels sinònims de la paraula que hi hagi al diccionari i substituir a la nova llista la paraula original pel seu sinònim.
- Les paraules de la llista original que no apareixen al diccionari han d'afegir-se directament a la nova llista sense modificar-les.

La funció tindrà aquesta capçalera:

```
def conversioSinonims(frase, diccionari)
```

Per fer aquesta funció podeu utilitzar la funció `random.randrange(n)`, que retorna un número aleatori entre 0 i n-1.

Exercici 8 (nivell mig) – EXERCICI AVALUABLE

Volem crear un índex de les paraules que apareixen en un fitxer utilitzant diccionaris i llistes en Python. Per aconseguir-ho us demanem:

1. Escriure una funció que es digui `indexParaules` rebi com a paràmetre el nom d'un fitxer i retorni un diccionari que contingui, per cada paraula que apareix al fitxer una llista amb tots els nº de línia on podem trobar aquesta paraula i quantes vegades apareix aquesta paraula a cada línia. Abans de construir l'índex convertiu tots el caràcters a minúscules amb el mètode `lower` dels strings. Elimineu també els possibles signes de puntuació: ',', '.', ':', ';', '!', '?'. Ho podeu fer amb el mètode `replace` dels strings per substituir tots aquests símbols per espais en blanc abans de dividir el text en paraules.

   Per exemple, si el contingut del fitxer és el següent:

   ```
   Python is an easy to learn, powerful programming language. It is
   an ideal language for scripting and rapid application
   development in many areas on most platforms. It has
   efficient high-level data structures and a simple but effective
   approach to object-oriented programming.
   ```

   el diccionari contindria, entre d'altres, els valors següents:

   ```
   'and': [[1, 1], [3, 1]],
   'simple': [[3, 1]],
   'is': [[0, 2]],
   'powerful': [[0, 1]],
   'an': [[0, 1], [1, 1]],
   ...
   ```

2. Escriure una funció que permeti cercar una paraula en el fitxer utilitzant el diccionari que es crea amb la funció de l'apartat anterior. La funció haurà de tenir aquesta capçalera:

   ```
   def cercaParaula(paraula, index, nomFitxer)
   ```

   on index és el diccionari que es crea amb la funció de l'apartat anterior. La funció ha de retornar en una llista totes les línies del fitxer on apareix la paraula que es passa com a paràmetre, així com el número de vegades que apareix a cada línia.

   Per exemple si a partir d'un índex creat sobre el fitxer de l'apartat anterior es fa la crida:

   ```
   cercaParaula('and', index, 'fitxerParaules')
   ```

   la llista que hauríem de retornar és la següent:

```
[['an ideal language for scripting and rapid application development', 1],
['data structures and a simple but effective approach to', 1]]
```

EXERCICIS D'ARRAYS

Exercici 9 (nivell mig)

Donada una matriu, que es passa com a paràmetre en un array de *numpy*, escriure les funcions següents:

a) `canviaValorForaRang`: retorna una nova matriu en què tots els valors fora del rang *[n,m*], que es passa com a paràmetre, es substitueix pel valor *x,* que també es passa com a paràmetre.
b) `sumaNoDiagonal`: suposant que la matriu és quadrada, retorna la suma de tots els elements que no estan a la diagonal.
c) `diagonalDominant`: suposant que la matriu és quadrada, retorna si la matriu és 'diagonal dominant', és a dir, si tots els elements de la diagonal són més grans que la resta dels elements de la seva fila.
d) `maxFila`: retorna una llista que conté, per cada fila, una tupla amb el valor més gran i la posició que ocupa dins de la
e) `intercanviFiles`: retorna una nova matriu amb les files *i* i *j* (que es passen com a paràmetres) intercanviades.

Exercici 10 (nivell mig) – EXERCICI AVALUABLE

Suposem que tenim una matriu *mxn* on cada fila guarda els valors de les notes obtingudes pels *m* estudiants d'una assignatura en cadascun dels *n* lliuraments d'exercicis obligatoris de l'assignatura. Si no s'ha fet algun lliurament, el valor de la nota serà -1. Escriure funcions per:

a) `mitjanaLliuraments`: retornar la mitjana de tots els lliuraments de l'estudiant que correspon a la fila *k* (que es passa com a paràmetre), si ha fet tots els lliuraments. Si no els ha fet tots, retornarà 0.
b) `nAprovats`: retorna quants alumnes han lliurat tots els exercicis i la mitjana dels seus lliuraments és superior a 5.
c) `resultatExercici`: Donat un número d'exercici que es passa com a paràmetre, retorna el número d'estudiants que l'han lliurat, i la nota mínima, mitjana i màxima de tots els lliuraments fets.
d) `abandonamentsSetmanals`: suposant que els lliuraments són setmanals (un per setmana), retorna en una llista el nº d'estudiants que han abandonat l'assignatura en cada setmana. Suposarem que un estudiant abandona l'assignatura a la setmana *k* si a partir d'aquella setmana ja no ha fet cap més lliurament (si un estudiant no ha fet mai cap lliurament, considerarem que abandona a la setmana zero).

EXERCICIS DE CLASSES

Exercici 11 (nivell bàsic) – EXERCICI AVALUABLE

Crear una classe `Vector` que permeti representar un vector en un espai de N dimensions, $v = (x_1.x_2, \dots, x_N)$. El número de dimensions ha de poder ser diferent per cada vector.

1. El constructor ha de rebre com a paràmetre una llista numèrica amb les coordenades del vector.
2. Redefiniu el mètode __str__ perquè es mostri per pantalla el seu contingut en format $[x_1, x_2, \dots, x_N]$.
3. Sobrecarregueu l'operador de suma perquè retorni un nou vector que sigui la suma de dos vectors component a component, si tenen la mateixa dimensió. Si no, ha de generar una excepció de tipus `ValueError`.
4. Sobrecarregueu l'operador de resta perquè calculi la distància euclidiana entre els dos vectors: $d(u, v) = \sqrt{\sum_{i=1}^{N}(u_i - v_i)^2}$. Si els dos vectors no tenen la mateixa longitud ha de generar una excepció de tipus `ValueError`.
5. Sobrecarregueu l'operador de multiplicació perquè retorni un nou vector que sigui la multiplicació de totes les components del vector per un número enter que es passa com a paràmetre.