Laboratorio di Python

Dizionari, Esercizi su dizionari

Università di Bologna

23 aprile 2015

Sommario

Esercizi

1. Scrivere una funzione che presa una lista e un valore *i* appartenente a tale lista restituisca due liste. La prima lista contiene i valori minori o uguali a *i* e la seconda lista i valori maggiori di *i* [non utilizzare il metodo sort].
2. Scrivere una funzione che presi come parametri due liste ordinate ne crei una sola anch'essa ordinata.
3. Scrivere una funzione che presa una lista controlli se al suo interno sono presenti valori uguali in posizioni successive, e in tal caso sostituire il valore uguale con il numero di occorrenze di valori uguali. (es. L=['a','b',1,1,'c'] $\rightarrow$ O=['a','b',2,'c'])

Esercizio 1

```
def double():
    L=eval(input("Inserire una lista: "))
    i=eval(input("Scegliere un valore all'interno della lista: "))
    if i not in L:
        print("Il valore considerato deve essere compreso nella lista"
        return double()
    N1=[]
    N2=[]
    for e in range(len(L)):
        if L[e]<=i:
            N1.append(L[e])
        else:
            N2.append(L[e])
    print (N1)
    print (N2)
    return None
```

È corretto?

Esercizio 2

```
def una(a,b):
    if type(a)!=list or type(b)!=list:
        return ("Le due variabili devono essere liste (ordinate)")
    c=[]
    c+=a
    c+=b
    c.sort()
    return c
```

È corretto?

Esercizio 3

```
def sost_occ(S):
    # inizializzazione delle variabili almeno_due=False n=1 R=[]
    for i in range(len(S)-1):
        if i==(len(S)-2) and S[-1]==S[-2]:
            almeno_due=True
            n+=1
            R+=[n,]
        if S[i]==S[i+1]:
            almeno_due=True
            n+=1
        else:
            if almeno_due==True:
                R+=[n,]
                n=1
                almeno_due=False
            else:
                R+=[S[i],]
    if S[-1]!=S[-2]:
        R+=[S[-1],]
    return R
```

È corretto?

Definizione e Inizializzazione

I dizionari sono sequenze mutabili. I dizionari hanno un indice chiamato chiave. La chiave è definibile da un qualunque tipo immutabile.
Per inizializzare un dizionario *d* si usa il comando:

- $d = \{\}$

Definizione e Assegnazione

$d = \{chiave : val, chiave2 : val, \cdots\}$

chiave può essere di tipo stringa, intero, tupla, ecc... qualsiasi tipo immutabile

Val può essere di qualsiasi tipo anche un dizionario stesso

Ad esempio:
$d = \{'vocali' : ('a','e','i','o','u'),'consonanti' : ('b','c',...),'punteggiatura' : (':',';')\}$

chiavi del nostro dizionario sono: 'vocali', 'consonanti' , 'punteggiatura'

valori referenziati dalle rispettive chiavi sono ('a','e','i','o','u'); ('b','c', ...); (',',';')

Operazioni sui dizionari

Modifica : d['vocali'] = ('a','e','i','o','u', 'A','E', 'I', 'O','U') $\rightarrow$ modifica la chiave già presente con i valori a destra dell'uguale

Assegnazione : d['alfabeto']=('a','b', ...)$\rightarrow$ associa alla variabile *d* una nuova *chiave* con i valori a destra dell'uguale

Cancellazione : del d['consonanti'] $\rightarrow$ cancella da *d* la *chiave* e i *valori* ad essa associati

Numero di coppie chiave valore len(d) $\rightarrow$ conta il numero di coppie *chiave valore* presenti nel dizionario *d*

Metodi key and values versione 3.x

keys : d.keys() $\rightarrow$ ritorna la vista dinamica delle chiavi

values d.values() $\rightarrow$ ritorna la vista dinamica dei valori

Python 3.x

```
>>>d={'vocali': ('a','e','i','o','u'), 'consonanti': ('b','c', ...)}
>>> ks = d.keys()
>>> kv = d.values()
>>> print(ks)
(['vocali', 'consonanti'])
>>> print(kv)
([('a','e','i','o','u'),('b','c', ...) ])
>>> d['punteggiatura'] = (':',';')
>>> print(ks)
([vocali', 'consonanti' 'punteggiatura'])
>>> print(kv)
([('a','e','i','o','u'),('b','c', ...), (':',';')])
```

Esercizio

1. Scrivere una funzione che dato un insieme di studenti e voti ad essi associati restituisca un dizionario degli studenti e dei voti ad essi associati, suddivisi per lettera del cognome. Le chiavi sono i gruppi (a-f), (g-o), (p-z); i valori sono le liste degli studenti e i voti ad essi associati.
2. Si definisca la funzione di inserimento e cancellazione di un dato studente nel dizionario appena creato.
3. Si definisca la funzione che inserisca a uno studente presente nel dizionario una lista di voti.
4. Si definisca la funzione che preso uno studente e un voto aggiunga il voto alla lista dei voti.

Esercizio 1.1

```
def dizionario(s):
    if type(s)==tuple:
        d={}
        k1=('A','F')
        k2=('G','O')
        k3=('P','Z')
        d={k1:[],k2:[], k3:[]}
        itero= list(d.keys()) #  versione 3.x
        for i in range(len(s)):
            for k in itero:
                t=str(s[i][0])
                if t[0] >= k[0] and t[0]<=k[len(k)-1]:
                    d[k].append(s[i])
        return (d)
```

Se stiamo usando la versione 2.x, il codice diventa: itero= d.keys()

Esercizio 1.2a

```
def inser_studente(s,d):
    itero= list(d.keys()) #  versione 3.x
    for k in itero:
            t=str(s[0])
            if t[0] >= k[0] and t[0]<=k[len(k)-1]:
                  d[k].append(s)
    return (d)
```

Se stiamo usando la versione 2.x, il codice diventa: $itero = d.keys()$

Esercizio 1.2b

```
def canc_studente(s,d):
    itero= list(d.keys()) #  versione 3.x
    for k in itero:
        t=str(s[0])
        if t[0] >= k[0] and t[0]<=k[len(k)-1]:
            ite=d[k]
          lung=len(ite)
            for l in range(lung):
                if t == ite[l][0]:
                    del ite[l]
                    d[k]=ite
                    return (d)
```

Cerca studente

```
def cerca_studente_n(s,d):
     itero= list(d.keys()) #  versione 3.x
    for k in itero:
        if type(s)==list:
            t=str(s[0])
            if t[0] >= k[0] and t[0]<=k[len(k)-1]:
                ite=d[k]
                for l in range(len(ite)):
                    if t == ite[l][0]:
                        return (k,l)
        elif type(s)==str:
            if s[0] >= k[0] and s[0]<=k[len(k)-1]:
                ite=d[k]
                for l in range(len(ite)):
                    if s == ite[l][0]:
                        return (k,l)
    return(None,None)
```

Esercizio 1.2b che richiama cerca_studente

```
def canc_studente_n(s,d):
    k, l=cerca_studente_n(s,d)
    if k is not None:
        ite=d[k]
        del ite[l]
        d[k]=ite
    return (d)
```

Esercizio 1.3

```
def insert_voto(s,d,n):
    k, l=cerca_studente_n(s,d)
    ite=[]
    if k is not None:
        ite=d[k]
        if type(n) == list:
            for i in n:
                ite[l][1].append(i)
                d[k]=ite
            return d
        elif type(n)==int:
            ite[l][1].append(n)
            d[k]=ite
            return d
    return d
```