Programmazione Orientata agli Oggetti e Scripting in Python

Prova Scritta del 22 Giugno 2004

Esercizio 1 (5 punti)

Scrivere la funzione *limits*, che rende il più grande e il più piccolo dei valori passati in ingresso come "named parameters" (in caso di chiamata senza parametri rende None).

Per esempio le chiamate:

```
limits( a=1.2, b=0.5, c=3.4, d=0.2 )
limits( w=1.0, t=1.5, z=-3.2 )
```

danno come risultato rispettivamente: `(0.2, 3.4)` e `(-3.2, 1.5)`

Esercizio 2 (5 punti)

Dato il file di testo "risultati.txt", contenente un insieme di linee nel seguente formato.

cognome nome matricola voto

dove *cognome* e *nome* sono delle stringhe non contenenti spazi e *matricola* e *voto* sono due numeri interi (il primo di 5 cifre e il secondo avente valore compreso tra 0 e 30)

scrivere un programma in Python che legge dal file "risultati.txt" tutte le righe e le scrive nel file "risultati2.txt", dopo averle ordinate in base ai seguenti criteri: i risultati devono essere presentati in ordine decrescente di voto (dal più alto al più basso); in caso di votazione identica, le righe dovranno essere ordinate (in senso crescente) prima per cognome, poi per nome e infine per matricola.

Esempio:

risultati.txt

```
ROSSI ALBERTO 23456 30
BIANCHI STEFANO 56789 30
ROSSI MARIO 12345 30
ROSSI ALBERTO 45678 30
VERDI GIUSEPPE 52483 27
VERDI FRANCESCO 56463 27
```

risultati2.txt

```
BIANCHI STEFANO 56789 30
ROSSI ALBERTO 23456 30
ROSSI ALBERTO 45678 30
ROSSI MARIO 12345 30
VERDI FRANCESCO 56463 27
VERDI GIUSEPPE 52483 27
```

Nota: trascurare eventuali errori/eccezioni di I/O; evitare l'uso di classi definite dall'utente (ricorrere a funzioni e/o strutture elementari come liste, tuple, dizionari).
Suggerimento: data una stringa contenente un insieme di parole separate da spazi, il metodo split() richiamato su di essa rende la lista delle parole che la compongono.

Esercizio 3 (6 punti)

Spiegare brevemente che cosa stampa il seguente codice Python (motivare la risposta):

```
class B(object):
  default=0
  def __init__(self,a=None):
    if a!=None: self.A=a
    else:  self.A=B.default
  def method(self,a):
    self.A+=a

class D(B):
  def method(self,a):
    self.A*=a

b = B(100)
b.method(20); print b.A
b = B('Ciao')
b.method(' a tutti'); print b.A
d = D(5)
d.method(3); print d.A
D.default = 10
d = D(20); print d.A
```

Esercizio 4 (6 punti)

Spiegare brevemente il meccanismo di assegnamento degli attributi di un oggetto durante la sua costruzione, e fare un semplice esempio di procedura di inizializzazione di un oggetto (__init__).

Esercizio 5 (5 punti)

Spiegare brevemente che cosa stampa il seguente codice Python (motivare la risposta):

```
def zap(f1,f2):
  return lambda x, y: f1(f2(x,y),f2(y,x))
def sub(x,y): return x-y
def mul(x,y): return x*y
foo = zap(mul,sub)
print foo(10,20)
```

Esercizio 6 (6 punti)

Spiegare brevemente che cosa stampa il seguente codice Python (motivare la risposta):

```
class zot:
  def __init__(self,function):
    self.function = function
  def __call__(self,*args):
    return self.function(*args)

def foo(x): return x

z = zot(foo)
print z(1)
```