Laboratorio 4

Informatica@DSS 2020/2021

Massimo Lauria <massimo.lauria@uniroma1.it>*

Lo scopo del laboratorio è di esercitarsi e misurare la propria preparazione: gli esercizi non sono troppo difficili, se si sono seguite le lezioni. Non vi viene comunque messo alcun voto.

Modalità di lavoro: gli studenti devono lavorare in autonomia o in piccoli gruppi, senza disturbare i colleghi. Il lavoro di gruppo è fruttuoso solo se tutti partecipano e se ognuno scrive una propria soluzione per tutti gli esercizi.

Raccomandazioni: leggete bene il testo degli esercizi prima di chiedere chiarimenti. In ogni caso sarò in aula con voi. Alla fine della lezione per favore rispondete al questionario, disponibile al link alla fine di questo documento.

Uso dei file di test: per aiutarvi a completare questa esercitazione avete a disposizione dei programmi di test per testare la vostra soluzione. Questi sono simili a quelli che avrete in sede di esame, pertanto vi consiglio di imparate ad usarli. Per portare a termine l'esercizio è necessario

- scrivere un file di soluzione **col nome specificato**;
- dare alla funzione **il nome specificato**;
- porre il file di test corrispondente **nella stessa cartella**;
- eseguire in quella cartella il comando

```
$ python3 <fileditest>
```

dove <fileditest> va ovviamente sostituito con il nome del file di test appropriato per l'esercizio su cui state lavorando.

*http://massimolauria.net/informatica2020/

Per ogni esercizio ci sta un file di test indipendente, così da poter lavorare sugli esercizi uno alla volta con più agio. Controllate i parametri passati in input ed eventualmente sollevate le eccezioni `ValueError` o `TypeError`.

Il risultato di ogni test è una schermata (o più schermate) nella quale si mostra:

- se è stato trovato il file con il nome corretto,
- se il file contiene la funzione con il nome corretto,
- se chiamate alla funzione con diversi valori dei parametri terminano restituendo risultati corretti.

Per ogni funzione scritta vengono eseguite chiamate con diversi valori dei parametri. L'esito dei test viene riportato con il carattere

- `E` se la chiamata non può essere eseguita,
- `F` se la funzione non restituisce il risultato corretto,
- `.` se la funzione restituisce il risultato corretto.

1 Conteggio delle vocali

Scrivete una funzione `conteggiovocali(testo)` che restituisca il numero di vocali presenti nella stringa `testo`. Ad esempio

```
print(conteggiovocali("Contate le vocali."))
```

```
7
```

File della soluzione: `conteggiovocali.py`

File di test: `test_conteggiovocali.py`

2 Conteggio delle parole

Scrivete una funzione `conteggioparole(testo)` che restituisca il numero di parole presenti nel stringa `testo`. Ad esempio

```
print(conteggioparole("Python4Dummy? È, tra i manuali python,il peggiore."))
```

```
8
```

Notate che `Python4Dummy` è una parola sola, e che il pezzo di testo `python,il` contiene due parole. Per questo esercizio potreste usare i metodi di stringa

- `str.isalpha` che restituisce `True` quando una stringa è **non vuota** ed è fatta di soli caratteri alfabetici;
- `str.isdigit` che restituisce `True` quando una stringa è **non vuota** ed è fatta di soli cifre numeriche.

In particolare potete usare questi metodi per verificare se un carattere è una cifra o un carattere alfabetico.

Ad esempio:

```
print("Massimo".isalpha())
print("342".isdigit())
print("Massimo!".isalpha())
print("3 42".isdigit())
```

```
True
True
False
False
```

File della soluzione: `conteggioparole.py`

File di test: `test_conteggioparole.py`

3 Terne pitagoriche

Scrivete una funzione `ternepitagoriche1(N)` che dato il parametro N restituisca il numero di terne pitagoriche, ovvero le terne (a, b, c) tali che

- $a^2 + b^2 = c^2$
- $1 \leq a < b < c \leq N$

File della soluzione: `ternepitagoriche1.py`

File di test: `test_ternepitagoriche1.py`