Sapienza Università di Roma
Corso di Laurea in Ingegneria Informatica e Automatica A.A. 2021-22

Esercitazione Python n. 3 -- 19 Ottobre 2021

Obiettivo dell'esercitazione è scrivere i primi programmi che fanno uso dell'istruzione while. Per risolvere gli esercizi descritti di seguito dovete modificare i file "esercizio1.py", "esercizio2.py", ecc., che trovate nella cartella dell'esercitazione.

Per **consegnare l'esercitazione svolta**, comprimete nuovamente la cartella LabPython03 **in un file .zip** e caricatela sulla pagina del corso dal link https://classroom.google.com/u/0/w/MTY5MzU2NjY1MTg2/t/all (Dalla sezione 'Esercitazioni su Python', selezionate 'Esercitazione 3' e successivamente 'Visualizza Compito'; poi cliccate su 'Aggiungi o crea' e scegliete il file da caricare). Al termine dell'operazione cliccate su 'Contrassegna come completato'. Vi chiediamo di effettuare la consegna entro le *23:59 di Mercoledì 20 ottobre*.

Si rammenta che le esercitazioni che non rispettano le indicazioni per la consegna non possono essere considerate per l'assegnazione del punto bonus. In particolare, ricordate di comprimere e riconsegnare l'intera cartella dell'esercitazione, e non singolarmente i file degli esercizi, di NON usare formati di compressione diversi da .zip, di NON rinominare i file o metterli in sottocartelle. Inoltre, prestate attenzione ad input ed output degli esercizi: gli **input** devono essere richiesti all'utente **UNO PER VOLTA** e **NELL'ORDINE INDICATO** nel testo dell'esercizio; l'**output** restituito (quando richiesto) deve essere **STAMPATO ESATTAMENTE** nel formato indicato dall'esercizio, che potete verificare anche facendo riferimento agli esempi riportati (ad esempio, nell'esercizio 1, se vengono inseriti "-5" "-5" e "*" il vostro programma deve stampare a schermo "0", e non "sono stati inseriti 0 interi positivi", oppure "il risultato è 0" o altri messaggi del genere).

In ogni esercizio, se non diversamente richiesto, potete sempre assumere che gli input forniti siano coerenti con la traccia (cioè, se l'esercizio chiede di dare in input al programma un numero intero, potete assumere che l'input sia sempre convertibile in intero, e non è necessario nel codice effettuare controlli in tal senso).

Esercizi

1) Scrivere un programma python che chiede in input all'utente un intero **n>2** e stampa tutti i numeri *pari* compresi tra **2** e **n** uno per riga (includendo **2** ed **n**). *Esempio:*
   - Inserendo l'intero "4", il programma stampa "2" e a capo "4"
   - Inserendo l'intero "7", il programma stampa "2", a capo "4" e infine a capo "6"
2) Scrivere un programma python che chiede in input all'utente un numero intero maggiore di zero e stampa a schermo tutti i suoi divisori interi positivi. *Esempio:*
   - Inserendo l'intero "6", il programma stampa "1", a capo "2", a capo "3", a capo "6"
3) Scrivere un programma python che chiede in input all'utente una sequenza di numeri interi, uno per uno in maniera iterativa terminando la richiesta di inserimento in input quando viene immesso un numero divisibile per 5. Prima di terminare, il programma stampa a schermo la *divisione intera* fra l'ultimo numero inserito e 5. *Esempio:*
   - Inserendo in questo ordine le stringhe "13", "-19", "11", "7" e "-10", il programma termina stampando "-2"
4) Scrivere un programma python che chiede in input all'utente due numeri interi x e y compresi nell'intervallo [0,10] (assumete che i numeri immessi siano contenuti nell'intervallo) e stampa tutti i numeri fino a 10 esclusi x e y uno per riga. *Esempio:*
   - Inserendo gli interi "1" e "2", il programma stampa "0" a capo "3" a capo "4" a capo "5" a capo

"6" a capo "7" a capo "8" a capo "9" a capo "10"

5) Scrivere un programma python che chiede in input all'utente una sequenza di stringhe, una per una in maniera iterativa, terminando la richiesta di inserimento in input quando viene immessa una stringa fatta esclusivamente da caratteri alfabetici minuscoli (quindi anche senza spazi) e stampa per ogni stringa il primo e l'ultimo carattere. Si assuma che l'utente non inserisca mai la stringa vuota. Si ricorda che, data una stringa `s`, *(i)* l'istruzione `s.isalpha()` restituisce `True` se `s` contiene solo caratteri alfabetici, altrimenti restituisce `False`, e che *(ii)* l'istruzione `s.islower()` restituisce `True` se `s` contiene almeno un carattere alfabetico minuscolo e nessun carattere alfabetico maiuscolo, altrimenti restituisce `False`. *Esempio:*
   - Inserendo in questo ordine le stringhe "Casa", "esercitazione3" e "abaco", il programma stampa "Ca", e poi, dopo aver chiesto di inserire una nuova stringa, stampa "e3", e, dopo aver chiesto di inserire una nuova stringa, stampa "ao" e termina.
6) Scrivere un programma python che chiede in input all'utente una stringa composta da almeno un carattere e la scandisce carattere dopo carattere partendo dal primo, sino a quando o la stringa finisce o si incontra un carattere il cui codice Unicode è maggiore di 100. Il programma deve stampare su di una riga il motivo per cui è terminato. Ci sono due possibili motivi:
   - stringa consumata senza trovare il carattere. In questo caso il programma deve stampare "stringa consumata e carattere non trovato"
   - trovato carattere con codice Unicode maggiore di 100. In questo caso il programma deve stampare "Il primo carattere con codice Unicode maggiore di 100 è" seguito dalla stampa del carattere.

   Esempio:
   - inserendo la stringa "CONTE", stampa "stringa consumata e carattere non trovato"
   - inserendo la stringa "abaco", stampa "Il primo carattere con codice Unicode maggiore di 100 è o"
   - inserendo la stringa "adamo", stampa "Il primo carattere con codice Unicode maggiore di 100 è m"
7) Scrivere un programma python che chiede in input all'utente un carattere **c** e una sequenza di stringhe, una per una in maniera iterativa, terminando la richiesta di inserimento in input quando il carattere **c** compare **più di 2 volte** nella stringa inserita in input. Prima di terminare il programma stampa il numero di occorrenze di **c** nell'ultima stringa inserita. *Esempio:*
   - Inserendo in questo ordine il carattere "t" e le seguenti stringhe, "atto", "arto" e "architettata" il programma termina stampando "4"
   - Inserendo in questo ordine il carattere "b" e le seguenti stringhe, "bingo" e "obbrobrio", il programma termina stampando "3"
8) Scrivere un programma che chiede in input all'utente di inserire una stringa palindroma, e nel caso in cui la stringa inserita non sia palindroma continua a chiedere l'inserimento di una stringa palindroma. Il programma termina all'inserimento della stringa palindroma e stampa la il messaggio "stringa palindroma di lunghezza ", seguito dalla lunghezza della stringa. *Esempio*
   - Inserendo la stringa "arcobaleno" il programma stampa "non palindroma, inserire una stringa palindroma: "; inserendo successivamente "mamma"" il programma stampa "non palindroma, inserire una stringa palindroma"; inserendo 'itopinonavevanonipoti' il programma termina e stampa "stringa palindroma di lunghezza 21"
9) Scrivere un programma python che chiede in input all'utente un intero **n** e controlla se **n** è un numero primo. In caso affermativo stampa "numero primo", altrimenti stampa "numero non primo". *Esempio:*
   - Inserendo l'intero "7", il programma stampa "numero primo"
   - Inserendo l'intero "4096", il programma stampa "numero non primo"
10) Scrivere un programma python che chiede in input all'utente un numero intero positivo **n** maggiore di 1 e stampa uno per riga tutti i numeri primi compresi tra 2 e **n,** includendo 2 ed **n (**qualora fosse primo). *Esempio:*

- Inserendo il numero “9”, il programma stampa “2”, a capo “3”, a capo “5”, a capo “7”