Laboratorio 5

Elementi di Informatica e Programmazione

Esercizio 1: modulo `myinput`

Scrivere il modulo `myinput.py` che contenga le seguenti funzioni di utilità per l'acquisizione di dati dall'utente:

- `inputYesNo(message, yes, no)` restituisce `True` se l'utente ha scritto la stringa fornita come secondo argomento, restituisce `False` se l'utente ha scritto la stringa fornita come terzo argomento; ignora la differenza tra maiuscole e minuscole. Esempio di utilizzo:

```
if inputYesNo("Vuoi continuare? (S/N) ", "S", "N"):
    # vuole continuare
```

- `inputStringStartingWith(message, startingString)` restituisce una stringa che inizia con la stringa `startingString`; se `startingString` è la stringa vuota, viene restituita la prima stringa digitata dall'utente (anche se è la stringa vuota)
- `inputStringEndingWith(message, endingString)` restituisce una stringa che termina con la stringa `endingString`; se `endingString` è la stringa vuota, viene restituita la prima stringa digitata dall'utente (anche se è la stringa vuota)
- `inputStringContaining(message, substring)` restituisce una stringa che contiene la stringa substring; se substring è la stringa vuota, viene restituita la prima stringa digitata dall'utente (anche se è la stringa vuota)
- `isDecimalInteger(s)` restituisce True se e solo se la stringa s contiene un numero intero decimale, che ha questo formato: zero o più spazi iniziali, un eventuale segno meno, una o più cifre decimali (da 0 a 9), zero o più spazi finali (quindi, ad esempio, non ci può essere uno spazio tra il segno meno e la prima cifra del numero)
- `inputDecimalInteger(message)` restituisce un numero intero; la funzione deve utilizzare in modo opportuno la funzione isDecimalInteger

- `inputPositiveDecimalInteger(message)` restituisce un numero intero positivo; la funzione deve utilizzare in modo opportuno la funzione `inputDecimalInteger`
- `inputNegativeDecimalInteger(message)` restituisce un numero intero negativo; la funzione deve utilizzare in modo opportuno la funzione inputDecimalInteger
- `inputNonPositiveDecimalInteger(message)` restituisce un numero intero non positivo; la funzione deve utilizzare in modo opportuno la funzione inputDecimalInteger
- `inputNonNegativeDecimalInteger(message)` restituisce un numero intero non negativo; la funzione deve utilizzare in modo opportuno la funzione inputDecimalInteger
- `isFloating(s)` restituisce True se e solo se la stringa s contiene un numero decimale in virgola mobile, che ha questo formato: zero o più spazi iniziali, un eventuale segno meno, una o più cifre decimali (da 0 a 9), un eventuale separatore decimale (il carattere "punto") seguito da una o più cifre decimali, un'eventuale lettera "e" (maiuscola o minuscola) seguita da un eventuale segno meno e da una o più cifre decimali, zero o più spazi finali
- `inputFloating(message)` restituisce un numero in virgola mobile; la funzione deve utilizzare in modo opportuno la funzione isFloating

Tutte le funzioni di tipo `input...` devono chiedere ripetutamente il dato all'utente finché questo non rispetta le specifiche della funzione, riproponendo il messaggio message (senza visualizzare messaggi d'errore). Tutte le funzioni di tipo `is...`, invece, non devono avere alcuna interazione con l'utente (né in input né in output).

Esercizio 2: nomi dei numeri

Scrivere il programma `number_name.py` che acquisisca un numero intero positivo minore di un milione e ne visualizzi la descrizione in italiano, seguendo le regole riportate qui: https://it.wiktionary.org/wiki/Appendice:Tutti_i_numeri _in_lettere (attenzione ai molti casi particolari…). Definire funzioni opportune in modo da evitare, per quanto possibile, la duplicazione di codice.

Potete usare le funzioni del modulo `myinput` sviluppato nell'esercizio precedente per gestire l'interazione con l'utente.

Esercizio 3: gioco fantasy - inventario

In un videogioco a tema fantasy ciascun protagonista ha a disposizione un inventario di oggetti. Il videogioco rappresenta l'inventario come un dizionario dove le chiavi sono stringhe che descrivono l'oggetto e il valore è un intero rappresentante il numero di oggetti di quel tipo. Ad esempio, il dizionario

```
{'corda': 1, 'torcia': 6, "monete d'oro": 42, 'spada': 1, 'freccia': 12}
```

rappresenta un inventario con una corda, 6 torce, 42 monete, una spada e 12 frecce.

Sviluppate una funzione `display_inventory()` che dato un inventario lo stampi come segue:

```
Inventario:
12 freccia
42 monete d'oro
1 corda
6 torcia
1 spada
Numero totale di elementi: 62
```

L'ordine degli elementi non è importante.

Esercizio 4: anagrammi

Un anagramma è una permutazione delle lettere di una parola compiuta in modo tale da ottenere un'altra parola di senso compiuto. Quindi data una coppia di parole possiamo chiederci se una è l'anagramma dell'altra. Scrivere una funzione `anagrams` che riceve in input una lista di parole e ritorna una lista dei gruppi di parole che sono l'una l'anagramma dell'altra.

Ad esempio, se la lista di parole è:

```
[
  'canori', 'carino',
  'cranio', 'naso',
  'rancio', 'gelo',
  'gole', 'lego'
]
```

la funzione deve ritornare

```
[
  ['gelo', 'gole', 'lego'],
  ['canori', 'carino', 'cranio', 'rancio']
]
```

Attenzione: la parola `naso` non fa parte di nessuna lista di output, perchè non è l'anagramma di nessun'altra parola della lista.

Nella progettazione della funzione considerate i seguenti fatti:

- la relazione di anagramma è simmetrica, ovvero "lego" è anagramma di "gole" e "gole" è anagramma di "lego"
- se ordiniamo le lettere di una parola in ordine alfabetico (ovvero "lego" diventa "eglo"), abbiamo che tutte le parole che sono anagramma l'una dell'altra hanno la stessa sequenza di lettere ordinate.
- se una parola è l'anagramma dell'altra, hanno lo stesso numero di lettere

Per implementare la funzione ricordate che (tra le altre cose):

- per ordinare i caratteri di una stringa `s` potete usare `sorted(s)`. Qual è il risultato? Qual è il tipo di dato?
- potete trasformare una lista di caratteri `l` in una stringa usando `"".join(l)` (controllate la documentazione del metodo `join` delle stringhe)
- potete usare un dizionario per memorizzare i gruppi di parole che sono una l'anagramma dell'altra. Quali sono le chiavi? Quali sono i valori?
- dato un dizionario `d`, con `d.values()` potete iterare sui *valori* contenuti nel dizionario

Per testare la funzione potete usare la lista di parole contenuta nel file `parole.txt` scaricabile da moodle.