1 Gestore di una libreria

La libreria *Le Pagine di Andrea* ogni giorno ha un volume di clienti tale da causare al proprietario e ai suoi dipendenti situazioni di grande disagio. Dato che non è possibile assumere più personale o ridurre la clientela, Andrea vi chiede di creare un software che possa aiutarlo nel suo lavoro.

Andrea acquista i libri da un singolo fornitore ad un prezzo **fisso** di 10 euro ciascuno. Inizialmente sono state acquistate 2 copie di 1000 libri diversi. Mentre ogni settimana viene acquistato un quantitativo di libri pari a `quantity`, che è decisa da Andrea sulla base delle vendite della settimana precedente. Andrea decide di investire inizialmente cinquantamila euro.

Si consideri che i libri **usati in buono stato** siano messi in vendita a 12 euro, mentre il prezzo di ogni libro **nuovo** sia 20 euro. Partendo dal file `Esercitazione_Libreria.py`, implementare una classe `BookShop` che abbia i seguenti metodi.

1. Un metodo `buy_from_suppliers` che prenda in ingresso `quantity` e `copies_number` che permetta ad Andrea di comprare dai fornitori più copie di libri.
2. Un metodo `sell_new_book` con ingressi `book_title` e `book_author` che permetta di vendere al cliente un libro presente tra gli articoli in vendita. Comunicare all'utente se il libro desiderato non è in vendita. Restituire il libro acquistato dal cliente togliendolo dalla merce nuova disponibile.
3. Un metodo `sell_used_book` con ingressi `book_title` e `book_author` che permetta di vendere al cliente un libro usato in vendita. Comunicare all'utente se il libro desiderato non è in vendita. Restituire il libro acquistato dal cliente togliendolo dalla merce usata disponibile.
4. Un metodo `estimate_book_conditions` avente ingresso `book` di tipo `Book` che restituisca un intero. Se il libro non è in buono stato, il metodo restituisce 0 comunicando al cliente che non è possibile accettare il libro.
   Se il libro è in buono stato, è necessario stimare il prezzo. Se l'importo stimato è pari a 0, allora comunicare al cliente che il libro non ha mercato e restituire 0; altrimenti accettare il libro e restituire l'importo stimato al cliente.
5. Un metodo `_is_in_good_state`, avente in ingresso `book` di tipo `Book` con il seguente funzionamento. se le pagine lette sono minori del 10% delle pagine totali, allora sicuramente il libro è in ottime condizioni. Altrimenti è deciso randomicamente se lo stato è buono no. Per farlo, aprire l'interprete Python e digitare queste due righe di codice: provate ad utilizzare `random.choice`.
   ```
   import random
   help(random.choice)
   ```

6. Un metodo `_decide_price` che restituisca il prezzo generato randomicamente appartenente all'intervallo [0, `USED_BOOK_MAX_PRICE_POSSIBLE + 1`], dove quest'ultima è una costante fornita nel testo dell'esercizio.
7. Due metodi `on_sale_goods_summary` e `total_books_on_sale` per stampare a video il titolo e l'autore di tutti i libri usati e nuovi in vendita e il numero di articoli totali in vendita.

Porre attenzione all'aggiornamento della quantità di denaro posseduta da Andrea, contenuta nell'attributo `current_account` della classe.
I libri nuovi acquistati da Andrea mediante il metodo `buy_from_suppliers` devono essere inseriti nella lista `on_sale_new_books`. I libri usati, se accettati nel metodo `estimate_book_conditions`, devono essere inseriti nella lista `on_sale_used_books`.