# VIEW

PER SEPARAZIONE MODELLO VISTA:

-tolto tutti i system.out.print da Fruitore e Fruitori e spostati in FruitoriView: in questo modo in Fruitore/i restano solo i dati (model) e i metodi che operano su di essi, mentre FruitoriView fa da view (li stampa) e anche da controller (?), fornendo al model l’input dell’utente. (fatto con tutte le classi)

Fare RisorseView con metodi uguali di FilmsView e LibriView per non avere codice duplicato?  
alcuni metodi hanno come parametro la classe sui quali vengono invocati, in modo che il metodo sia uguale ma stampi “libro” o “film” a seconda dell’invocazione.

Fruitore e Fruitori, Libro Libri Film Films, Prestito Prestiti, dovrebbero essere a posto

CONTROLLER:

Si potrà creare una specie di controller grasp per creare e gestire tutti i controller?

sposto nel controller i metodi del model che richiedono interazione con la view: model non deve dipendere da VIEW.

Per esempio addFilm (in films) richiedeva interazioni con la view. Ora delle interazioni si occupa il controller, Films si limita ad aggiungere il film creato.

Va fatto con tutto :C

filmController creato nel main, passato tra i vari menu fino ad arrivare a MenuAggiungiRisorsa, dove serve il controller per interagire con la view e creare il film.



Il controller (tramite la view) aggiorna i dati, e poi aggiorna quelli del model.

L’oggetto filmController viene passato tra i metodi e il suo model resta aggiornato, basta salvare l’archivio.

I dati restano salvati nel model del controller, che le salva sui file. Poi a inizio programma il main carica tali dati e li ridà al controller. Quindi poi al posto di passare il parametro Films passerò filmController?

filmController lavora su archivio e poi salva le sue modifiche (solo ai film) su file.

Quando filmArchivio modifica qualcosa, salva su file (che poi verrà caricato a inizio programma)

Se io creo archivio e lo passo come parametro al costruttore di FilmController, da filmController modifico l’archivio originale!

# MENU

Creata una classe per ogni menu, tolte tutte le parti di grafica e spostate nel package *view*.

Sono tutti Extract Class (del refactoring) ?

Anche per la grafica sono tutti extract method: i syso diventano metodi nella view (quindi anche move method)

Sono classi statiche, cioè non vengono istanziate ma hanno metodi statici.

Potrebbero diventare singleton? Così si possono istanziare ma solo una volta.