Dipartimento di Informatica Sistemistica e Comunicazione
Corso di Laurea in Informatica

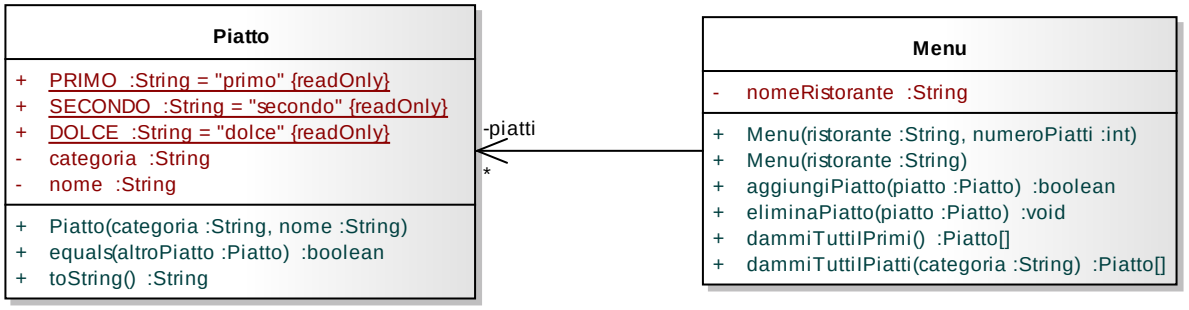
Programmazione 2

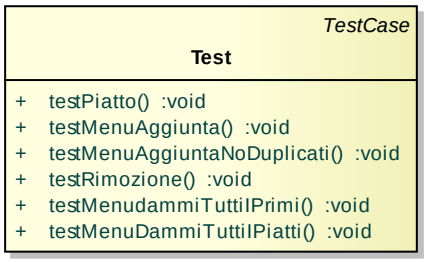
30 Giugno 2014 – Recupero Primo compitino

Testo parte di pratica

Un Menu contiene un insieme di piatti. Ogni piatto ha un nome e una categoria di appartenenza (primo, secondo, e così via). Si implementino le classi descritte nel seguente diagramma UML che modella i concetti sopra introdotti.

Nota: si eseguano i test man mano che si procede con l'implementazione, per verificare incrementalmente il lavoro via via fatto. Se i test o parte di essi non vengono superati, l'implementazione è con buona probabilità sbagliata, almeno parzialmente. Si eviti quindi di eseguire i test solo alla fine del lavoro, quando ormai sarebbe tardi per apportare correzioni.





Classe `Piatto`:

- ✓ rappresenta un piatto di un ristorante caratterizzato da un nome e da una categoria di appartenenza che può ricadere solo nei 3 valori specificati come costanti all'interno della classe (PRIMO, SECONDO, DOLCE)
- ✓ la categoria è immutabile, il nome è mutabile
- ✓ definisce un costruttore che inizializza entrambi gli attributi
- ✓ il metodo `toString` restituisce una stringa con l'informazione relativa ad un piatto (nome e categoria)
- ✓ il metodo `equals` permette di testare l'uguaglianza di due piatti secondo la seguente definizione: due piatti sono uguali se hanno lo stesso nome e la stessa categoria.

Classe `Menu`:

- ✓ rappresenta un menu di un ristorante e come tale è una collezione di piatti (da realizzare attraverso array di reference). È caratterizzata dal nome del ristorante a cui il menu appartiene (`nomeRistorante`)
- ✓ definisce un costruttore per inizializzare il menu. Il parametro `nomeRistorante` specifica il nome del ristorante, mentre `numeroPiatti` specifica il numero massimo di piatti che potranno essere contenuti nel menu; se tale valore è minore di 1, verrà comunque costruito un menu con capienza massima di 30 piatti
- ✓ definisce un ulteriore costruttore (overloading del costruttore) che inizializza il nome del ristorante a cui il menu si riferisce e imposta la capienza massima di piatti al valore di default 30

- ✓ L'attributo `nomeRistorante` è mutabile, ma i piatti non sono direttamente accessibili dall'esterno
- ✓ Il metodo `aggiungiPiatto(piatto): boolean` aggiunge un nuovo piatto al menu. L'inserimento ha esito negativo se il menu già contiene un numero di piatti pari alla sua capacità, o se un piatto uguale (`equals`) a quello specificato in ingresso è già presente. Quando l'inserimento ha esito negativo, lo stato del menu rimane immutato. Il metodo restituisce `true` se l'inserimento ha avuto esito positivo, `false` in caso contrario
- ✓ Il metodo `eliminaPiatto(piatto): Piatto` rimuove il piatto uguale al parametro passato se presente. Restituisce il piatto rimosso, oppure `null` se non presente
- ✓ Il metodo `dammiTuttiIPiatti(categoria): Piatto[]`restituisce in un array tutti i piatti che hanno la categoria uguale a quella specificata. Se non esistono piatti di quella categoria, il metodo restituisce `null`.
- ✓ Il metodo `dammiTuttiIPrimi(): Piatto[]` restituisce in un array tutti i piatti che hanno la categoria uguale al valore della costante `PRIMO` definita in `Piatto`. Se non esistono piatti di quella categoria, il metodo restituisce `null`.

Si identifichino gli eventuali metodi di incapsulamento non specificati nel testo.

Infine, si implementi il metodo `testMenudammiTuttiIPrimi` nella classe `Test` in modo tale da verificare il corretto funzionamento del metodo `dammiTuttiIPrimi` di `Menu`. In particolare:

- si istanzi un menu il cui nome ristorante è "il mio ristorante"
- si aggiungano i seguenti piatti:
  - categoria: primo, nome: Norma
  - categoria: secondo, nome: Filetto
  - categoria: primo, nome: Ravioli di magro
- si verifichi che l'array restituito dall'invocazione del metodo `dammiTuttiIPrimi` sia uguale a 2