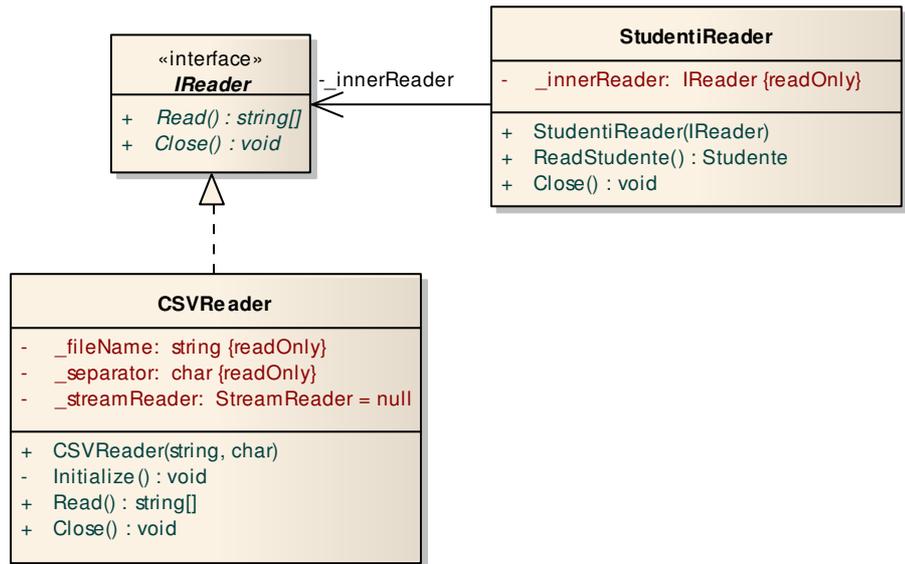
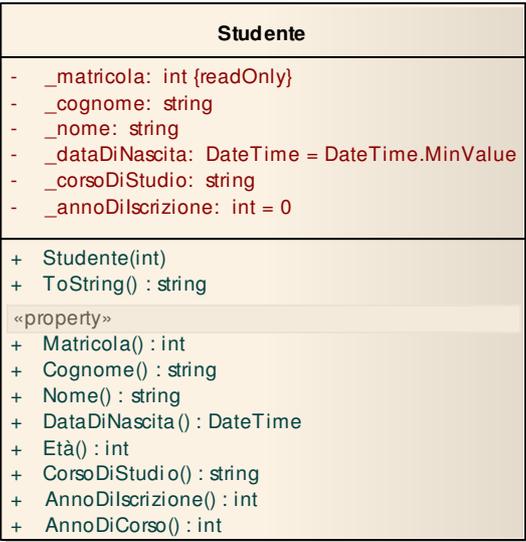


Laboratorio 1

Passo 0 – Creare una nuova *solution* di nome **Lab1** e un nuovo progetto di tipo *ConsoleApplication* di nome **Lab1**.

Passo 1 – Definire la classe `Studente` (ved. Il corrispondente diagramma UML). La proprietà **Età** deve restituire l'età dello studente in base alla data attuale e alla sua data di nascita. La proprietà **AnnoDiCorso** deve restituire l'anno di corso dello studente in base alla data attuale e al suo anno di immatricolazione.

Passo 2 – Definire un'interfaccia di nome `IReader` e una classe di nome `CSVReader` che la implementi (ved. Il corrispondente diagramma UML). `IReader` deve modellare un cursore *forward only* su un insieme di tuple (tutti dati di tipo stringa) – a ogni lettura (`Read()`) deve restituire la tupla successiva sotto forma di un array di stringhe e quando non ci sono più tuple disponibili deve restituire `null`. Il costruttore della classe `CSVReader` deve accettare come argomenti il nome di un file di testo e un carattere di separazione.



Un'istanza di `CSVReader` deve permettere la lettura riga per riga del file di testo; per effettuare tale operazione si utilizzi uno `StreamReader`. Il metodo `Read` di `CSVReader` deve leggere una riga da `_streamReader`, tramite il metodo `ReadLine` di `StreamReader`, e deve restituire un array di stringhe se è stata letta correttamente una riga dal file, `null` in caso contrario. Ogni linea letta contiene una serie di sotto-stringhe divise dal carattere di separazione; tali sotto-stringhe formano il risultato del metodo `Read` di `CSVReader`. Per effettuare tale suddivisione si utilizzi il metodo `Split` della classe `String`.

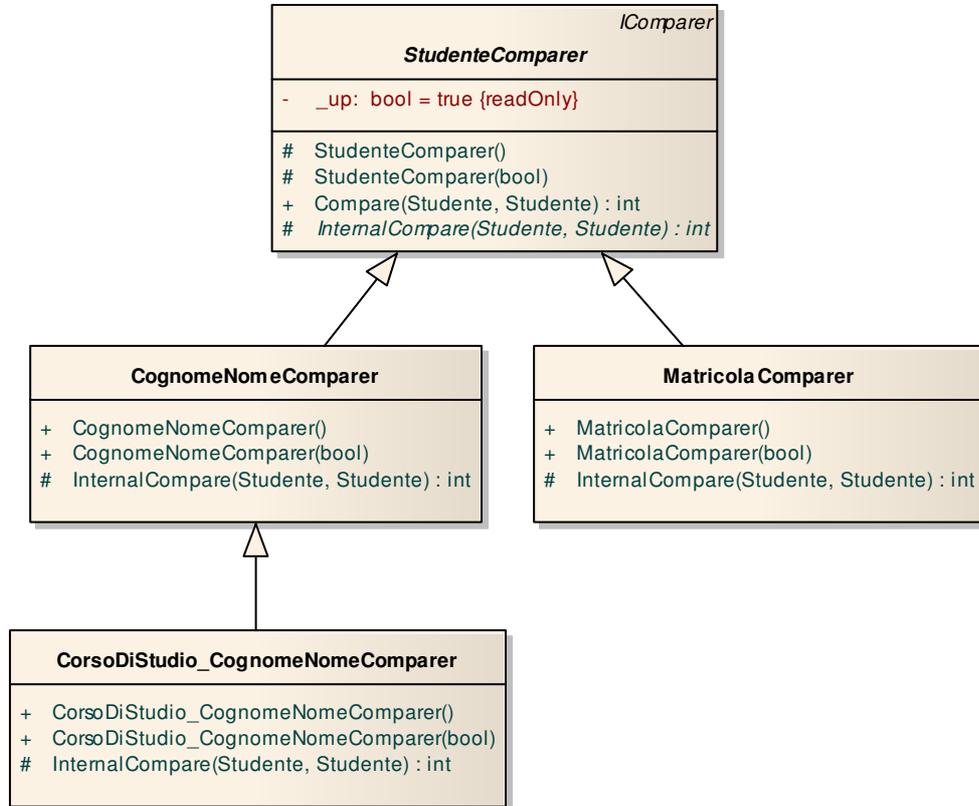
Passo 3 – Definire una classe di nome `StudentiReader` (ved. Il corrispondente diagramma UML) che incapsula un `IReader` in modo da consentire la lettura di un insieme di tuple, ciascuna delle quali contiene i dati di uno studente nel seguente ordine: `matricola`; `cognome`; `nome`; `corso di studio`; `data di nascita`; `anno di iscrizione`. Il metodo `ReadStudente`, utilizzando i servizi dell'`IReader`, deve creare un nuovo studente per ogni tupla letta e inizializzarlo con i dati contenuti nella tupla stessa; una volta raggiunta la fine dell'insieme di tuple il metodo deve restituire `null`.

Passo 4 – Per verificare il corretto funzionamento di tutte le classi realizzate, nel `Main` leggere il file CSV (scaricabile dal sito del corso) tramite uno `StudentiReader` e un `CSVReader` e

Laboratorio 1

inserire tutte le istanze di `Studente` così create in una lista di studenti; quindi, visualizzare a *console* tutte le proprietà di tutti gli studenti in lista.

Passo 5 – Definire la classe, astratta `StudenteComparer` e le sottoclassi concrete `CognomeNomeComparer`, `MatricolaComparer` e `CorsoDiStudio_CognomeNomeComparer` (ved. Il corrispondente diagramma UML).



La classe `StudenteComparer` deve implementare l'interfaccia `IComparer<Studente>`. Il campo `_up`, indica il senso dell'ordinamento. Il metodo `InternalCompare` è astratto e nelle sottoclassi deve effettuare un confronto sempre di tipo ascendente. Il metodo `Compare` invoca `InternalCompare` e ne aggiusta il risultato in base al valore di `_up`. Si noti l'utilizzo del *pattern method template*.

La classe `CognomeNomeComparer` deve permettere di confrontare due istanze di `Studente` per cognome e nome.

La classe `MatricolaComparer` deve permettere di confrontare due istanze di `Studente` per matricola.

La classe `CorsoDiStudio_CognomeNomeComparer` deve permettere di confrontare due istanze di `Studente` prima per corso di studio e quindi per cognome e nome.

Passo 6 – Per verificare il corretto funzionamento di tutte le classi realizzate, nel `Main`, dopo aver letto gli studenti dal file CSV:

1. ordinare gli studenti tramite un `MatricolaComparer` e visualizzare a *console* il risultato;
2. ordinare con ordinamento discendente gli studenti tramite un `CognomeNomeComparer` e visualizzare a *console* il risultato;
3. ordinare gli studenti tramite un `CorsoDiStudio_CognomeNomeComparer` e visualizzare a *console* il risultato.