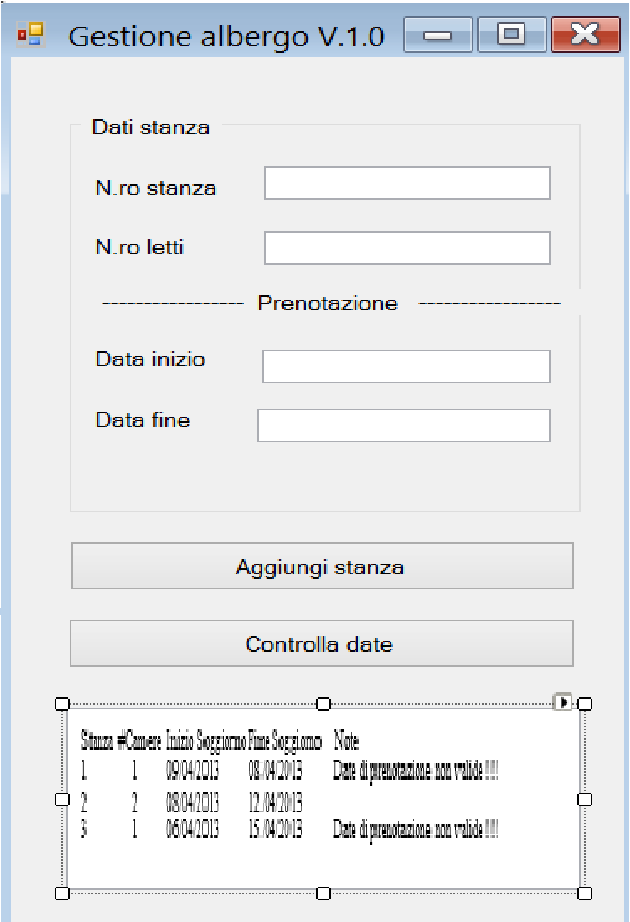
### Verifica di recupero debiti 1.mo trimestre

Cognome \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Nome \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Data \_\_\_\_\_\_\_\_\_\_

### Programmazione OOP in C# - Albergo

# classe 4.ta G– a.s. 2016-2017

Si vuole implementare un software che consenta la **gestione** della **prenotazione** delle stanze di un albergo. Le stanze sono classificabili in ***singole*** e ***doppie*** a seconda dei posti letto presenti. Le stanze possono essere *libere* oppure *occupate*; in quest’ultimo caso occorre poter specificare le date d’inizio e fine soggiorno/prenotazione.



Si consideri pertanto il seguente diagramma pseudo-UML.

classe astratta ***Stanza***

* Attributi
  + *Numero (es. 112)*
  + *NumPostiLetto (es.2)*
  + *Occupata (valore di verità vero/falso; lo stato iniziale è sempre falso)*
  + *DataInizioPrenotazione (stringa espressa nel formato gg/mm/aaaa)*
  + *DataFinePrenotazione (stringa espressa nel formato gg/mm/aaaa)*
* Metodi
  + *DataValida(): metodo astratto in grado di ritornare un valore booleano (es. true)*
* Proprietà (per ciascun attributo della classe)

classe **Singola**

* Metodi
  + *Singola( …..) : costruttore che assegna un valore a tutti gli attributi ( il numero di posti letto deve essere fissato uguale ad* ***1*** *e la stanza può essere libera o ocupata )*
  + *DataValida(): metodo “polimorfico” che ritorna un valore di verità opportuno in relazione a quanto segue:*
    - *la data d’ inizio prenotazione deve essere precedente e alla data di fine prenotazione*
    - ***non è possibile prenotare questo tipo di stanza per più di 5 giorni***

classe **Doppia**

* Metodi
  + *Doppia( …..) : costruttore che assegna un valore a tutti gli attributi ( il numero di posti letto deve essere fissato uguale ad* ***2*** *e la stanza può essere libera o ocupata )*
  + *DataValida(): metodo “polimorfico” che ritorna un valore di verità opportuno in relazione a quanto segue:*
    - *la data d’ inizio prenotazione deve essere precedente alla data di fine prenotazione*
    - *nessun vincolo sul numero di giorni prenotabili*

A tal proposito sviluppare un programma *C#/OOP* che, utilizzando le classi implementate*,* realizzi quanto segue.

* Consenta la *dichiarazione* di un array (***Stanze***) di almeno **tre** oggetti *Stanza*
* Consenta l’*istanza* nell’ array *Stanze*  di più stanze di tipo ***Singola***  e ***Doppia,*** inizializzando opportunamente tutti gli attributi associati
* Verificare, per ogni oggetto dell’array *Stanze*, che le date d’inizio prenotazione siano sempre “date valide” secondo i criteri imposti per l’implementazione del metodo *DataValida();* nel caso in cui si riscontri un’anomalia, visualizzare un opportuno messaggio d’errore (es. “ ***Date di prenotazione non valide!***”) secondo l’output seguente

**Note implementative**

* Implementare l’interfaccia rappresentata in figura rispettando le denominazioni dei controlli
* Creare un **file** separato per ogni classe del progetto. I file devono essere denominati come le classi in essi definite.
* Implementare, in tali file un ***namespace*** (*OggettiAlbergo*) in modo che raggruppi le definizioni di tutte le classi necessarie
* Le variabili globali devono essere definite in un file distinto (***Globals.cs***) da quello del programma principale
* Implementare le singole classi specificando in modo opportuno i **modificatori di accesso**
* Implementare in modo opportuno la **derivazione/polimorfismo** descritta dal diagramma soprastante ed i meccanismi che garantiscano il polimorfismo.