

Nome e cognome: _____ Matricola: _____

Programmazione ad oggetti mod.1, esame del 01/06/2023

PROVA DI PROGRAMMAZIONE

Durante questa prova si possono utilizzare il libro di testo, appunti o altro materiale (come, ad esempio, la stampa delle slide del corso). È invece proibito l'utilizzo di dispositivi elettronici quali, ad esempio, computer portatili, smartphone ed e-reader. Le risposte devono essere date scrivendo il codice sotto il testo dell'esercizio e, se necessario, sul retro del foglio. La durata della prova è di 60 minuti.

Nello svolgimento degli esercizi:

- È necessario massimizzare sempre l'information hiding,
- Le classi possono implementare metodi aggiuntivi se ritenuto utile o necessario, ma questi non devono essere visibili al di fuori della classe stessa.

Esercizio 1 (5 punti): Flight

Si implementi una classe `Flight` che rappresenta un volo aereo. In particolare, tale classe deve memorizzare la città di partenza e di arrivo (rappresentate da stringhe), l'orario di partenza e di arrivo (rappresentati come oggetti istanza di `java.util.Date`), e il numero del volo (che è un valore numerico intero compreso tra 0 e 9999). Una volta inizializzate, queste informazioni non possono più essere modificate, ma deve essere possibile leggerle esternamente.

Il costruttore controlla che l'orario di partenza sia prima di quello di arrivo. Per fare ciò, si possono usare i seguenti metodi `after` e `before` della classe `java.util.Date`

<code>boolean</code>	<code>after(Date when)</code> Tests if this date is after the specified date.
<code>boolean</code>	<code>before(Date when)</code> Tests if this date is before the specified date.

Nel caso in cui l'orario di partenza sia dopo quello dell'arrivo, il costruttore deve lanciare un'istanza della seguente eccezione.

```
public class FlightException extends Exception {}
```

Esercizio 2 (5 punti): Airplane

Si implementi una classe astratta `Airplane` che contenga un elenco di voli che un aeromobile percorre. Ciascuna tratta è un'istanza della classe `Flight` definita nell'esercizio 1. L'elenco dei voli deve essere inizialmente vuoto, e deve essere visibile solamente dallo stesso package e dalle sottoclassi.

La classe `Airplane` deve implementare poi un metodo per aggiungere un volo all'elenco. Nel caso in cui il volo aggiunto abbia un orario di partenza precedente all'orario di arrivo dell'ultimo volo in elenco, tale metodo deve lanciare una `FlightException` (definita nel testo dell'esercizio 1).

Esercizio 3 (5 punti): Boeing747

Si implementi una classe Boeing747 che estende Airplane.

La classe Boeing747 deve implementare un metodo che esegue il primo volo dell'elenco. Tale metodo deve controllare che la data attuale (ottenibile istanziando la classe `java.util.Date` senza passare alcun parametro al costruttore) sia successiva alla data di partenza del volo (usando i metodi `after` e `before` specificati nell'esercizio 1), e levare il primo volo dall'elenco. Nel caso in cui una delle condizioni elencate non sia validata, il metodo deve lanciare una `FlightException` (definita nell'esercizio 1). In caso di successo, tale metodo deve ritornare l'orario di arrivo previsto del volo.