

ESAME DI FONDAMENTI DI INFORMATICA T-2 del 2/07/2012

Proff. E. Denti – G. Zannoni

Tempo a disposizione: 4 ore MAX

NB: il candidato troverà nell'archivio ZIP scaricato da Esamix anche il software "Start Kit"

NOME PROGETTO ECLIPSE: CognomeNome-matricola (es. RossiMario-0000123456)

L'operatore turistico *Il Gatto e la Volpe Holidays* ha richiesto lo sviluppo di un'applicazione per guidare i potenziali clienti alla scelta della loro vacanza ideale.

DESCRIZIONE DEL DOMINIO DEL PROBLEMA.

I pacchetti turistici sono di tre tipi: *solo volo*, *volo più hotel*, *pacchetto completo*. Ogni pacchetto è caratterizzato da identificativo univoco, destinazione, data di inizio, durata, costo e descrizione. Più precisamente:

- l'identificativo univoco è una stringa alfanumerica senza spazi
- la destinazione è espressa dalla coppia Stato, Località (entrambe possono contenere spazi)
- la data di inizio ha la forma *giorno/mese/anno*
- la durata è espressa in giorni (zero nel caso di pacchetti *solo volo*)
- la descrizione è una stringa alfanumerica senza particolari vincoli.

L'applicazione deve permettere all'utente di specificare alcuni criteri (tipo di vacanza, costo massimo, destinazione, periodo) e mostrare i pacchetti vacanza che soddisfano i requisiti secondo un ordinamento selezionabile fra:

- costo crescente
- vicinanza alla durata richiesta dall'utente
- vicinanza alla data iniziale della vacanza richiesta dall'utente

Per "vicinanza" si intende la differenza in valore assoluto fra il valore richiesto dall'utente e quello della vacanza: ad esempio, se la durata richiesta è 14 giorni, una vacanza da 15 giorni è più vicina alla durata richiesta di una vacanza da 10 giorni, mentre è uguale a una vacanza da 13 giorni. Analogamente per la data di inizio vacanza.

Il file di testo [pacchetti.txt](#) contiene la descrizione dei diversi pacchetti offerti, nel formato più oltre specificato.

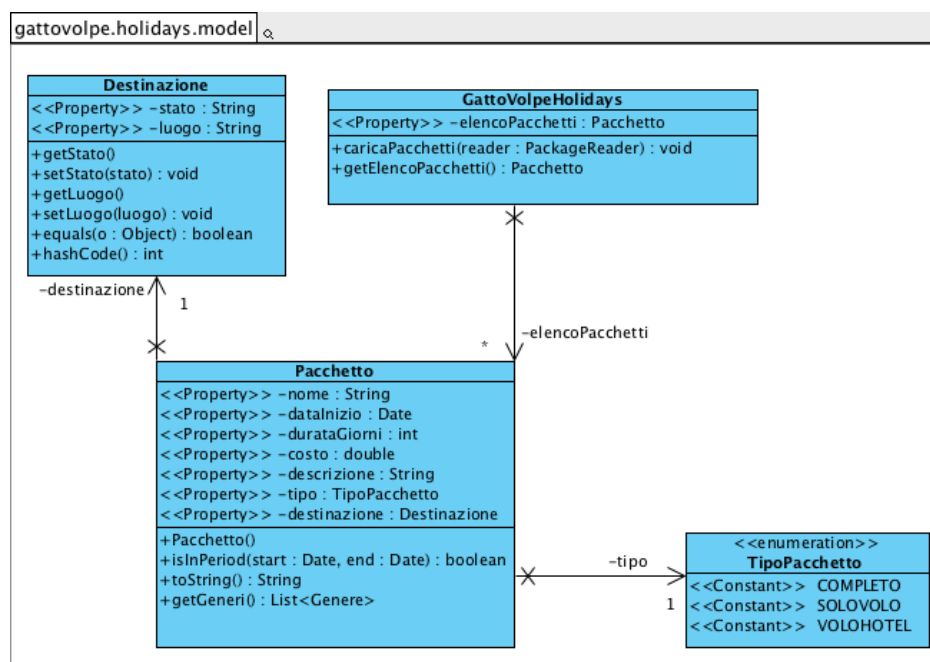
Parte 1

(punti: 16)

Dati (package gattovolpe.holidays.model)

(punti: 8)

Il modello dei dati deve essere organizzato secondo il diagramma UML di seguito riportato.



SEMANTICA:

- a) l'enumerativo **TipoPacchetto** (fornito nello start kit) definisce i tre possibili tipi di pacchetto;
- b) la classe **DateUtil** (fornita nello start kit) contiene vari metodi statici di utilità sulle date, in particolare per
 - i) normalizzare un Date alla mezzanotte, ii) creare una Date da una stringa nel formato gg/mm/aaaa, iii) calcolare la differenza in giorni fra due istanze di Date.
- c) la classe **Destinazione (da realizzare)** rappresenta la destinazione di un pacchetto turistico, caratterizzata dalle due proprietà *Stato* e *Località*, recuperabili tramite opportuni metodi accessor. È anche necessario ridefinire:
 - il metodo **equals** in modo che sia insensibile a maiuscole/minuscole;
 - il metodo **hashCode**, in modo coerente con il metodo **equals**.
- d) la classe **Pacchetto (da realizzare)** rappresenta il pacchetto turistico, caratterizzato dalle proprietà di cui sopra, tutte recuperabili tramite opportuni metodi accessor. Dev'essere inoltre definito il metodo **isInPeriod**, che verifica se il pacchetto rientra nell'intervallo di date passate come argomenti [estremi inclusi]. In particolare, le date che vengono passate come argomenti vanno normalizzate alla mezzanotte per non tenere di conto dell'ora del giorno, ed i pacchetti "solo volo" si intendono validi se la data del volo coincide con quella passata come primo argomento, indipendentemente dall'eventuale presenza di una data di termine passata come secondo argomento e che viene ignorata nel calcolo per questo tipo di pacchetto.
- e) la classe **GattoVolpeHolidays** (fornita nello start kit) definisce e mantiene le strutture dati fondamentali del sistema, tra cui in particolare l'elenco dei pacchetti turistici. Pertanto, essa definisce:
 - un metodo **caricaPacchetti** che, sfruttando il **PackageReader** (v. dopo) ricevuto come argomento, carica i dati dal file **pacchetti.txt**;
 - un metodo accessor **getElencoPacchetti** che permette di recuperare l'elenco dei pacchetti.

Lo Start Kit contiene anche i test (da includere nel progetto) per verificare il funzionamento di queste classi.

Persistenza (package **gattovolpe.holidays.persistence**)

(punti 8)

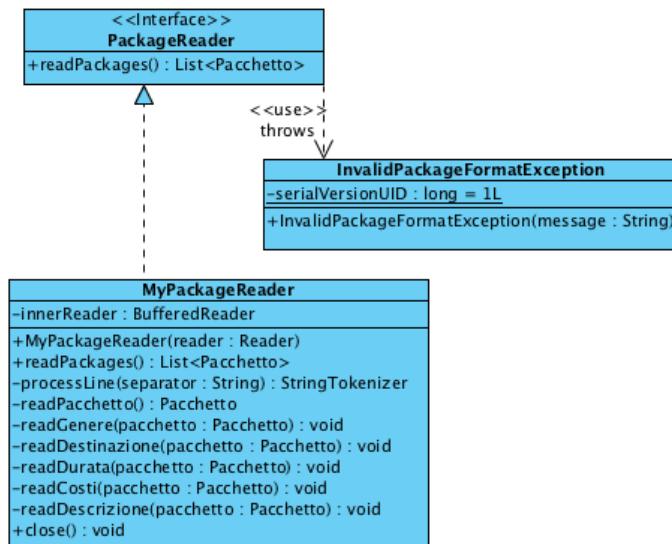
Il file di testo **pacchetti.txt** contiene la descrizione dei diversi pacchetti vacanza, ognuna su più righe: ogni pacchetto inizia con una riga "STARTPACKAGE" (che contiene anche l'identificativo e il tipo del pacchetto, scritto esattamente come il corrispondente valore nell'enumerativo **TipoPacchetto**) e una riga "ENDPACKAGE". Nelle altre righe troviamo:

- destinazione (sequenza *stato*, *località*)
- una data di inizio nel formato gg/MM/aaaa e una durata in giorni (separate una dall'altra da una virgola)
- il costo del pacchetto preceduto dalla indicazione della valuta "EUR", separati da spazio
- eventualmente, un testo descrittivo (su più righe), che però può anche mancare.

```
ESEMPIO DEL FILE pacchetti.txt
STARTPACKAGE p4z746 COMPLETO
Grecia, Stegna
9/6/2012, 7 giorni
EUR 395
Trattamento di pensione completa, 33km da Rodi centro
spiaggia e negozi nelle vicinanze
ENDPACKAGE
...
```

L'interfaccia **PackageReader** (fornita nello start kit) dichiara il metodo **readPackages** che legge una lista di **Pacchetto**; tale metodo dichiara di sollevare una **InvalidPackageFormatException** (fornita nello start kit) destinata a incapsulare le specifiche eccezioni eventualmente riscontrate nella lettura da file.

La classe **MyPackageReader (da realizzare)** implementa l'interfaccia **PackageReader**, permette la lettura da file di testo utilizzando un opportuno oggetto **Reader** ricevuto nel costruttore, e solleva l'eccezione richiesta quando opportuno.



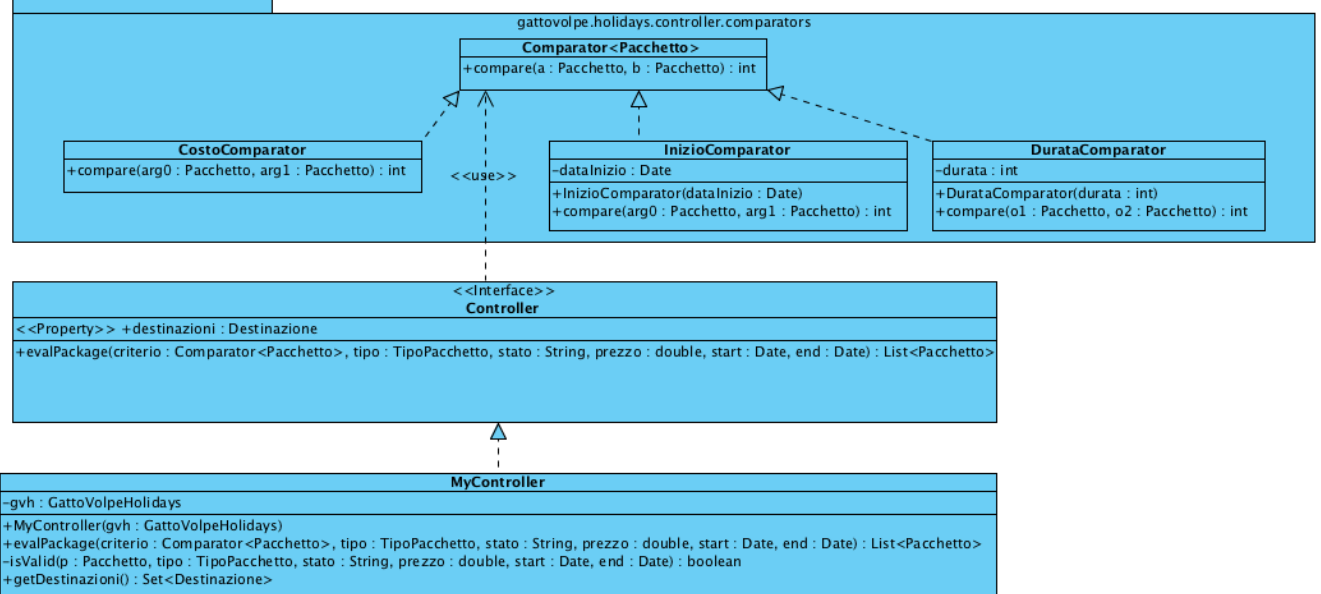
Lo Start Kit contiene anche i test (da includere nel progetto) per verificare il funzionamento di questa classe.

Parte 2

(punti: 14)

Controller (package gattovolpe.holidays.ui)

(punti 6)



La classe **MyController** (da realizzare) implementa l'interfaccia **Controller** (fornita nello start kit): il costruttore riceve in ingresso un'istanza di **GattoVolpeHolidays** pronta all'uso, ovvero con i pacchetti già caricati da file.

Il controller è responsabile del filtraggio dei pacchetti turistici in base ai requisiti impostati dall'utente e al loro ordinamento secondo il criterio richiesto. A tal fine, l'interfaccia **Controller** dichiara il metodo **getDestinzioni** che recupera tutte le destinazioni presenti nel file restituendo un **Set** di oggetti **Destinazione**, nonché il metodo **evalPackages** che restituisce la lista dei pacchetti che soddisfano i requisiti dell'utente. I requisiti sono quattro:

- il tipo di vacanza (*solo volo, volo+hotel, pacchetto completo*)
- la destinazione (*solo lo Stato*)
- il prezzo massimo accettabile
- la data iniziale e finale del periodo accettabile

La lista restituita da `evalPackages` deve essere ordinata in senso crescente in base al criterio espresso dal `Comparator<Pacchetto>` ricevuto come argomento; gli altri argomenti sono, ovviamente, i quattro valori necessari al filtraggio sopra specificato. I tre criteri di ordinamento sono contenuti nelle classi `CostoComparator` (da realizzare), `DurataComparator` (da realizzare) e `InizioComparator` (da realizzare): queste classi implementano tutte `Comparator<Pacchetto>` e sono contenute nel package `gattovolpe.holidays.controller.comparators`.

Lo Start Kit contiene anche i test (da includere nel progetto) per verificare il funzionamento di questa classe.

Interfaccia utente (package `gattovolpe.holidays.view`)

(punti 8)

L'interfaccia utente deve essere simile (non necessariamente identica) all'esempio mostrato in figura.

La classe `Program` (non mostrata nel diagramma UML, ma fornita nello start kit) contiene il main di partenza dell'intera applicazione.

La classe `MainFrame` (da realizzare) realizza la finestra principale, prevede un costruttore con parametro di tipo `Controller`, ed è così articolata (Fig. 1):

- in alto, opportune combobox e campi di testo permettono l'inserimento dei quattro requisiti di filtraggio (*suggerimento: per le date, invece dei campi di testo può essere furbo utilizzare un `JSpinner` con opportuno `SpinnerDateModel`...*);
- sul lato sinistro, una combobox permette di scegliere il tipo di pacchetto: subito sotto, tre radiobutton permettono di scegliere il criterio di ordinamento; nel caso della durata del soggiorno, un ulteriore campo di testo permette di inserire la durata del soggiorno desiderata;
- sul lato destro, un'area di testo mostra i tutti e soli i pacchetti che rispettano i criteri dati, nell'ordine specificato dai radiobutton.

Se viene variato uno dei quattro requisiti di filtraggio (Fig 2) o si cambia il criterio di ordinamento (Fig.3), i pacchetti vacanza mostrati nell'area di destra devono essere aggiornati in base alla nuova selezione.



Fig. 1

Il gatto e la volpe holidays

KENYA 910 02/07/2012 02/10/2012

KENYA2356; Dest. :Watamu; Costo: 900.0; Inizio: 03/07/2012; Durata: 9 giorni

Tipo COMPLETO

Basso costo ☐

Inizio vicino ☒

Durata simile ☐ 7

Fig. 2

Il gatto e la volpe holidays

KENYA 1000 02/07/2012 02/10/2012

KENYA2355; Dest. :Watamu; Costo: 966.0; Inizio: 03/07/2012; Durata: 9 giorni
 KENYA2356; Dest. :Watamu; Costo: 900.0; Inizio: 03/07/2012; Durata: 9 giorni
 KENYA2354; Dest. :Watamu; Costo: 936.0; Inizio: 28/07/2012; Durata: 9 giorni

Tipo COMPLETO

Basso costo ☐

Inizio vicino ☒

Durata simile ☐ 7

Fig. 3