Progetto di Programmazione Java (Michele Tarabella)

Gestione locale per compleanni di bambini

Il progetto consiste nella realizzazione di un programma Java per la gestione di un locale per compleanni di bambini.
I file *.java* che costituiscono il programma sono cinque. Tre di questi file contengono la definizione delle classi che costituiscono la struttura dati (*Prenotazione.java, PrenotazioneC.java, PrenotazioneCA.java*); il file *GestioneLocale.java* contiene i metodi principali per la gestione dei dati Prenotazione e infine *GestisciMain.java* contiene il main per la gestione dell'interfaccia testuale.

Struttura delle classi

Classi Prenotazione, PrenotazioneC, PrenotazioneCA

Le classi "Prenotazione" contengono informazioni come: la data della prenotazione, il nome del cliente che la effettua, il tipo di prenotazione, il numero degli invitati (nel caso sia previsto il catering), l'animazione scelta (nel caso sia prevista la formula con catering e animazione).
Le tre classi sono state implementate seguendo uno schema di eredità singola: `Prenotazione` **superclasse** di `PrenotazioneC` **superclasse** di `PrenotazioneCA`.
La classe Prenotazione viene quindi estesa da `PrenotazioneC` (prenotazione con catering) che aggiunge l'informazione riguardante al numero degli invitati. Quest'ultima classe viene estesa a sua volta da `PrenotazioneCA` (prenotazione con catering e animazione) che aggiunge come unica informazione il tipo di animazione.

Il meccanismo di eredità fa in modo che istanziando ad esempio una `PrenotazioneC` non viene aggiunto la variabile del tipo di intrattenimento, mentre invece se si istanzia una `PrenotazioneCA` viene ereditato dalla superclasse il dato contenente il numero di invitati.

Ogni classe è costituita da un insieme di metodi `Get`, uno per ogni dato.
Vi è inoltre un metodo per restituire una data in formato stringa `getDataStringa()`.
Infine, un metodo `getInfo()` che stampa un riassunto di tutta la prenotazione effettuata.
Le tre classi che costituiscono la struttura dati sono state implementate in tre file distinti renderne chiara la definizione. Le variabili d'istanza contenute nel file Prenotazione.java sono state dichiarate con il modificatore di visibilità "protected" per renderle visibili all'interno dello stesso package.

Classe GestioneLocale

La classe GestioneLocale contiene un vettore `elencoPrenotazioni` atto a contenere oggetti "Prenotazione" istanziati secondo le definizioni di tutta la loro gerarchia. Questa classe contiene inoltre una variabile booleana `flagMOD` che assume valori specifici qualora siano state apportate modifiche al vettore.

Oltre che a un costruttore che inizializza il vettore `elencoPrenotazioni` come vuoto, la classe è composta dai seguenti metodi:

METODI GETTER

1. `boolean getSavedIO():`
   restituisce il valore della variabile `flagMOD`();

2. `Prenotazione gelElement(int idx):`
   restituisce un elemento del vettore prenotazione prendendo un indice come parametro;

3. `int getIndice(Calendar dateIDX):`
   restituisce l'indice di un oggetto Calendar nel vattore di prenotazioni;

METODI SETTER

1. `setFlag(boolean IO):`
   metodo che setta la variabile `Flagmod` ogniqualvolta viene effettuata una modifica al file;

METODI AGGIUNGI, ELIMINA, VISUALIZZA

1. `Prenota(Calendar date, String cliente, int invitati, String animazione):`
   permette di istanziare un nuovo oggetto "prenotazione". Mediante opportuni controlli, l'algoritmo istanzia una prenotazione specifica: "Affitto semplice", "Catering", "Catering e animazione";

2. `Visualizza():`
   permette di visualizzare tutte le prenotazioni di ogni tipo, effettuate da tutti i clienti;

3. `VisualizzaCA()`:
   permette di visualizzare tutte le prenotazioni “Catering e animazione” effettuate da tutti i clienti;

4. `VisualizzaFiltro(Vector<Prenotazione> ind)`:
   permette di visualizzare tutte le prenotazioni effettuate da un certo cliente sulla base del suo nome o una porzione di esso(questo metodo sfrutta il meccanismo della funzione `filtraSelezione()` e applica ad ogni selezione il metodo `getInfo()` su ogni elemento `Prenotazione` del vettore ricevuto come parametro);

5. `Elimina(int idx)` e `Elimina(Prenotazione idx)`:
   due versioni ottenute applicando la regola del polimorfismo:

   5.1. la prima versione viene utilizzata nel momento in cui viene passato un indice (rappresentato da un intero) come parametro. Viene utilizzato nel caso si voglia eliminare una singola data;
   5.2. la seconda versione prende in input un oggetto `Prenotazione` che rappresenta l'indice della prenotazioni che vogliamo eliminare.Viene utilizzato nel caso in cui si vogliano eliminare tutte le prenotazioni effettuate da un cliente;

METODI di CARICAMENTO e SALVATAGGIO

6. `caricaFile(String nuovoFile)`
   Consente di aprire un file binario in input contenente un oggetto `GestioneLocale`. Restituisce `true` se il file viene caricato correttamente, restituisce false altrimenti e stampa il relativo messaggio di errore se:
   6.1. il file non è presente nella cartella;
   6.2. il file non contiene un oggetto Events;
   6.3. in caso di altri errori di input/output;

7. `salvaFile(String nuovoFile)`
   salva le modifiche al vettore `elencoPrenotazioni`, qualora il valore di `flagMOD` sia uguale a `true`.
   7.1. Restituisce True se il salvataggio è stato eseguito correttamente.
   7.2. Restituisce FALSE se ci sono stati problemi di Input/Output
   7.3. Altrimenti, se non ci sono state modifiche restituisce True;

METODI per RICERCA della PRIMA DATA DISPONIBILE

8. `Calendar firstDate(Calendar dateF)`
   Metodo che verifica la prima data disponibile rispetto a un'altra data ricevuta come parametro(si suppone che sia il giorno corrente).
   Viene utilizzato il metodo `roll(Calendar.DAY_OF_YEAR, 1)` della classe `Calendar` che “aumenta” di un giorno la data corrente finché verifica, mediante il metodo `binSearch(Calendar dateBS)`, che una data non è stata prenotata;

9. `dateAvailable(Calendar dateAV)`
   metodo che verifica se una data ha una prenotazione;

METODI di RICERCA

10. `binSearchCalendar dateBS)`
    Ricerca binaria di una data: prende in input un oggetto data `Calendar`, restituisce un intero che rappresenta l'indice nel vettore dell'oggetto `Prenotazione` in relazione alla data `dateBS`.
    Se l'intero restituito è -1, allora non è stata trovata quella data.

11. `filtraSelezione(String nomeCliente, char mode)`
    Ricerca lineare del vettore `elencoPrenotazioni`:
    Parametri: il primo è la stringa che rappresenta il nome del cliente che intende effettuare la prenotazione, il secondo parametro è il modo in cui si vuole effettuare la ricerca:
    11.1. `t` : inserendo il nome per intero;
    11.2. `p` : una porzione di esso;
    Le due tipologie di ricerca avranno come metodi utilizzati per le stringhe rispettivamente `equals()` e `contains()`.
    La funzione restituisce un vettore di oggetti Prenotazioni che hanno soddisfatto i vincoli del filtro nella selezione.

Classe GestisciMain

L'interfaccia testuale è implementata dal file *GestisciMain.java*.
In primo luogo si richiede di inserire il nome di un file nuovo o di caricarne uno già esistente.
Una volta aperto o creato un file, compare il menù che consente di effettuare le seguenti operazioni:

1. `[A]ggiungi prenotazione (richiede tutte le info):`
2. `[V]isualizza tutte le prenotazioni:`
3. `[P]renotazioni con Catering e Animazione:`
4. `[E]limina prenotazione:`
   4.1. `Per data`
   4.2. `Per nome cliente`
5. `[C]ontrolla data disponibile:`
6. `[L]ista prenotazioni per utente:`
7. `[S]alva:`
8. `[U]scita:`