Programmazione 2

Relazione primo progetto Intermedio

Gabriele Sergi

Corso A

Per questo progetto intermedio di Programmazione 2 dell'anno accademico 18-19, ho pensato a due implementazioni per l'interfaccia SecureDataContainer<E>.

SecureDataBase<E>:

Che contiene:

- HashMap<String, String> che contiene la coppia Utente, Password;
- List<Data<E>> implementata con un ArrayList<Data<E>>;
  Data<E> contiene:
  - un dato generico E data
  - una Stringa Owner per indicare chi e' il proprietario del dato
  - Vector<String> shered nel quale sono contenuti tutti gli utenti con cui e' condiviso il dato

SecureSillyDataBase<E>:

Che contiene:

- HashMap<String, Dati<E>> che contiene come chiave il nome dell'Owner

  Dati<E> che contiene:

  - List<E> dati implementata con un ArrayList<E> che contiene i dati dell'Owner;
  - Stringa passw;

Le principali funzione che hanno bisogno di chiarimenti sull'implementazione sono le seguenti:

- **removeUser:**
  Per entrambe le implementazioni, alla rimozione dell'utente vengono rimossi i dati di sua proprieta';

- **put:**
  Per entrambe le implementazioni ho deciso di inserire l'oggetto all'interno della collezione anche se gia' presente, questo perche' se esiste un metodo copy non ne vedo il motivo per non inserirne un altro uguale;

- **getSize**
  Per entrambe le implementazioni restituisce il numero dei dati appartenenti ad Owner e dei dati condivisi con Owner;

- **remove**
  Per entrambe le implementazioni elimina tutte le occorrenze del dato passatogli

- **share(SecureDataBase<E>)**
  Superati tutti i controlli di esistenza e proprieta', aggiunge alla lista shered relativa all'oggetto che contiene il dato, Other che poi avra' delle restrizioni su quel dato. Ovvero potra' lavorarci solo in lettura.

- **share(SecureSillyDataBase<E>)**
  Superati tutti i controlli di esistenza e proprieta', prende l'elemento data e lo inserisce all'interno della collezione di Other fornendoli di conseguenza tutti i permessi di accesso al dato, senza restrizioni.

TEST

I test sono all'interno di diverse MainClass che vanno ad esaminare il comportamento della struttura gradualmente. Infatti la MainClass0 effettua solo delle operazioni di inserimento e rimozione, mentre la MainClass4 richiama qualsiasi metodo sviluppato.
In particolare (la descrizione dell'ennesima MainClass si va a sommare alla precedente):

MainClass0: Inserimento e rimozione di utenti dalla collezione;

MainClass1: Inserimento di dati;

MainClass2: Condivisione dei dati;

MainClass3: Copia dei dati;

MainClass4: Rimozione di dati, rimozione di un utente possessore di dati e vista dei dati con Iterator;