Programmazione 2

Primo Progetto: *SecureDataContainer*

Matricola: *561281*
Studente: *Zegarelli Antonio*
Corso: *B*
Anno: *2018/2019*

1. Descrizione del problema

Sviluppare un componente software di supporto alla gestione di insiemi di dati SecureDataContainer che è un contenitore di oggetti di tipo E.
La collezione deve:

- Permettere la memorizzazione e condivisione di dati di tipo E.
- Garantire un meccanismo di sicurezza dei dati che gestisce le identità degli utenti.
- Fornire un meccanismo di controllo degli accessi che permette al proprietario del dato di eseguire una restrizione selettiva dell'accesso ai suoi dati inseriti nella collezione. Alcuni utenti possono essere autorizzati dal proprietario ad accedere ai dati, mentre altri non possono accedervi senza autorizzazione.

2. Descrizione dell' interfaccia

L' interfaccia SecureDataContainer<E> rappresenta una collezione modificabile per la memorizzazione e condivisione di dati di tipo E tramite autenticazione con id e password.
Ogni utente dopo essersi registrato può accedere con id e password al suo spazio utente formato dai dati di cui è proprietario e quelli che sono stati condivisi con lui.
I dati posseduti da un utente sono unici come anche i dati condivisi con lo stesso, che però hanno un contatore di condivisione che indica il numero di volte che un dato è stato condiviso con l'utente da utenti diversi in modo da rendere possibile la condivisione da parte di più utenti.

Metodi:

- **createUser (String Id, String passw):** Permette di creare un nuovo utente
- **getSize(String Owner, String passw):** Restituisce all' utente il numero dei dati posseduti
- **put(String Owner, String passw, E data):** Permette all' utente di inserire un nuovo dato tra quelli posseduti
- **get(String Owner, String passw, E data):** Restituisce all'utente un riferimento al dato cercandolo tra tutti gli oggetti nella collezione
- **remove(String Owner, String passw, E data):** Permette all'utente di rimuovere un dato posseduto
- **copy(String Owner, String passw, E data):** Permette all'utente di copiare tra i dati posseduti un oggetto condiviso con lui
- **share(String Owner, String passw, String Other, E data):** Permette all'utente di condividere un dato posseduto con un altro utente
- **removeShare(String Owner, String passw, String Other, E data):** Permette all'utente di rimuovere la condivisione di un dato posseduto con un altro utente
- **getIterator(String Owner, String passw):** Restituisce all'utente un iteratore che genera tutti i dati di cui è proprietario

3. Descrizione delle implementazioni

Costruttori e metodi privati delle implementazioni:

- **MySecureDataContainer():** Costruttore che permette di inizializzare la collezione senza utenti
- **MySecureDataContainer(String Id, String passw):** Costruttore che permette di inizializzare la collezione con un utente
- **auth(String Id, String passw):** Permette di verificare la correttezza delle credenziali inserite dall'utente per autenticarsi

Implementazioni:

- **MySecureDataContainer**
  Per la realizzazione di questa implementazione ho utilizzato 3 vector paralleli che rappresentano rispettivamente id, password e spazio dei dati di un utente.

- **MySecureDataContainerHT**
  Per la realizzazione di questa implementazione ho utilizzato 3 HashTable che rappresentano rispettivamente id, password e spazio dei dati di un utente utilizzando per tutte l'id dell'utente come chiavi

4. Descrizione classi supplementari

Per realizzare le implementazioni ho creato delle classi supplementari:

- **UserSpace:** rappresenta lo spazio di un utente con all'interno i dati posseduti e quelli condivisi con lui con un contatore che mi indica quante volte è stato condiviso da utenti diversi.
  Metodi:
  - **UserSpace():** Costruttore che inizializza lo spazio dei dati dell' utente
  - **addObj(E data):** Aggiunge un dato posseduto
  - **getObj(E data):** Restituisce un riferimento al dato posseduto dall'utente
  - **removeObj(E data):** Rimuove il dato posseduto dall'utente
  - **addShare(String id, E data):** Aggiunge l'id tra quelli con cui è condiviso il dato
  - **removeShare(String id, E data):** Rimuove l'id da quelli con cui è condiviso il dato
  - **addShared(E data):** Aggiunge un dato condiviso
  - **removeFromShared(E data):** Rimuove dato condiviso
  - **getSharedObj(E data):** Restituisce un riferimento al dato condiviso con l'utente
  - **getItOwned():** Restituisce all'utente un iteratore che genera tutti i dati di cui è proprietario l'utente
  - **getItShared(E data):** Restituisce all'utente un iteratore che genera tutti i dati condivisi con l'utente

- **UserData** rappresenta l'oggetto posseduto e gli id degli utenti con cui è stato condiviso.
  Metodi:
  - **UserData(E data):** Costruttore che inizializza la struttura di un dato di cui l'utente è proprietario con la lista degli id con i quali è condiviso vuota
  - **getObj():** Restituisce il dato
  - **addShare(String id):** Aggiunge l'id alla lista di quelli con cui è condiviso il dato
  - **removeShare(String id):** Rimuove l'id dalla lista di quelli con cui è condiviso il dato
  - **getSharedList():** Restituisce all'utente un iteratore che genera tutti gli id con cui è condiviso il dato