Relazione progetto PR2

Implementazione 1

STRUTTURE UTILIZZATE:

- **Vector di elementi “E”:**
  Utilizzato per la collezione che memorizza tutti i dati degli utenti.
- **Vector di elementi “User” (Classe creata appositamente):**
  Utilizzato per la memorizzazione di tutti gli utenti.
- **Vector di elementi “sharedData” (Classe creata appositamente):**
  Utilizzato per creare un’associazione tra i dati condivisi e gli utenti con i quali, i dati stessi, sono condivisi.

Ogni utente, ha i suoi dati in un range specifico nella collezione dati, delimitato dagli indici Start ed End, inizializzati, nel momento della creazione dell’utente, rispettivamente con Start = dimensione della collezione ed End = Dimensione della collezione - 1.
Quando un utente aggiunge un dato, quest’ultimo verrà inserito nel range specificato sopra, relativo al proprietario del dato. (Grazie al metodo put(x, pos) della collezione Vector in Java)
Siccome ogni operazione, richiede l’accesso dell’utente che chiede l’esecuzione del metodo stesso, è stato inserito un metodo apposito, chiamato “login” che restituisce l’indirizzo (o indice) dell’utente che ha effettuato l’accesso, solo se il login è avvenuto correttamente. Altrimenti, restituisce -1.
Il numero degli elementi di un utente, è calcolato semplicemente con una sottrazione tra gli indici che delimitano il range dati di ogni utente.
Il metodo get, controlla tra tutti i dati dell’utente, una corrispondenza con il dato passato.
Solo se la trova, restituisce una copia del dato. Se l’utente che ha richiesto l’operazione, non è il proprietario del dato, allora il software cercherà una corrispondenza col dato passato, tra i dati che gli sono stati condivisi.
Il metodo remove, rimuove il dato dalla collezione solo se chi ha richiesto l’operazione è il proprietario. Altrimenti, se è un utente con il quale il dato è semplicemente stato condiviso, rimuove il permesso di accedere al dato in lettura, ma il dato rimane nella collezione.
Ogni utente, può creare una copia dei propri dati nella collezione, solo se proprietario.
La condivisione, in questo software, consiste nel permettere a un utente con il quale qualcuno voglia condividere un dato specifico, un accesso, solo in lettura. Questo utente, non potrà rimuoverlo dalla collezione e non potrà creare copie del dato.
Se un utente, condivide un proprio dato con un altro utente, allora, in sd verrà creata un’associazione tra l’indice nella collezione dati del dato condiviso e, gli eventuali indici degli utenti con i quali il dato è stato condiviso.

Implementazione 2

STRUTTURE UTILIZZATE:

- **Vector di elementi “E”:**
  Utilizzato per la collezione che memorizza tutti i dati degli utenti.
- **Vector di elementi “User” (Classe creata appositamente):**
  Utilizzato per la memorizzazione di tutti gli utenti.

Differentemente dalla prima implementazione, nella seconda, abbiamo solo due strutture dati. In realtà, nel vector degli utenti, a ogni utente, è associata una tabella hash così costituita:
<Key = E, Value = Vector<Integer>>
Ovvero, a ogni dato, è associato un vector contenente gli indici degli utenti con i quali il dato è stato condiviso.
Nel momento della creazione di un utente, viene anche inizializzata una tabella hash.
Anche per questa implementazione è stato usato un metodo “login”, identico a quello usato nella implementazione 1.
Il metodo per l’immissione di un dato, da parte di un utente, prevede l’aggiunta del dato nella collezione e la creazione di un’associazione nella tabella hash, in cui, la chiave è il dato stesso e il valore è un vettore di interi, inizialmente vuoto.
Per ottenere un copia di un dato, se l’utente è il proprietario, si controllerà la presenza del dato nella propria tabella hash, altrimenti, si controllerà, per ogni utente, la presenza dell’indice dell’utente che ha richiesto l’operazione, nei vector che corrispondono al valore associato alla chiave, ovvero, il dato stesso.
Per la rimozione del dato, se l’utente è il proprietario, cancella il dato nella collezione e l’associazione in tabella hash, eliminando così, anche gli indici degli utenti con i quali il dato era stato condiviso.
Altrimenti, se la rimozione è richiesta da un utente che non è proprietario del dato che si vuole rimuovere, verrà eliminato il suo indice, nel vector degli indici associato al dato, nella tabella hash del proprietario.
Può essere creata una copia di un dato nella collezione solo dal proprietario.
Anche in questa implementazione, il concetto di condivisione, corrisponde al permesso di un semplice accesso in lettura che non prevede la possibilità, per i non proprietari, di copiare e/o rimuovere il dato.
Condividere il dato con un utente, comporterà l’aggiunta dell’indice corrispondente all’utente con il quale voglio condividerlo, nel vector di interi associato al dato stesso, nella tabella hash del proprietario.