AltriEsercizi8 - Esercizio1

Esercizio

Si consideri lo ADT Relazione nel file Relazione.ml (distribuito mercoed 11/5/2016). Si completi la definizione data con:
(a) AF, I
(b) Additionals
(c) Un caso di uso per verificare se un ('a,'b)relazione sia una relazione simmetrica.
(d) Un caso di uso per verificare se una relazione è contenuta in un'altra data.

Soluzione
In file RelazioneCompleta.ml in listing AltriEsercizi8lisngs, allegato.

AltriEsercizi8 - Esercizio2

Esercizio

Si definisca in Java una tipo astratto per relazioni binarie, su tipi generici, come quelle in esercizio1 ma per valori MODIFICABILI e con le sole operazioni addPair, removePair, isIn. In particolare, si fornisca:
(a) Interfaccia API
(b) Intestazione della classe;
(c) Stato Concreto;
(d) AF, I;
(e) Definizione dei metodi addPair, removePair, isIn;
(f) Trattamento degli Additionals (clone, equals, toString, elements);
(g) Una versione adatta dei casi d'uso discussi in esercizio1.

Soluzione
(a) In file MuRelazioneJAPI.java in AltriEsercizi8lisngs, allegati
(b-h) In file MuRelazioneADTX.java in AltriEsercizi8lisngs, allegati
(g) In file MainMuRelazioneADTX.java in AltriEsercizi8lisngs, allegati

AltriEsercizi8 - Esercizio3

Esercizio

Si assuma di operare su un sistema OCaml contenente il tipo astratto ('a,'b)relazione, definito come in esercizio1. Si chiede di specializzare la relazione binaria per operare con relazioni simmetriche. In particolare si deve fornire un nuovo tipo astratto specifico per relazioni simmetriche ed equipaggiato delle operazioni in ('a,'b)relazione, ovvero:
(a) Modulo API
(b) Intestazione del modulo ADT
(c) Stato Concreto
(d) AF, I
(e) Additionals
(f) Definizione delle operazioni
(g) Si dica se sia possibile usare i casi di uso di esercizio1 per i nuovi valori introdotti.

Soluzione
Due soluzioni che si differenziano nella rappresentazione usata.
1) discussa a lezione il 18/5/2016: (contiene banali errori sintattici da rimuovere) soluzione diretta, compatta ma non efficiente.
   In file RelazioneS1.ml in AltriEsercizi8lisngs, allegati
2) Soluzione più efficiente ma meno compatta (richiede una maggiore stesura di codice)
   In file RelazioneS.ml in AltriEsercizi8lisngs, allegati

Notare le differenti funzioni AF e I, utilizzato nelle 2 soluzioni.
Verificare il punto (g).

AltriEsercizi8 - Esercizio4

Esercizio

Lo stesso di esercizio3 ma in Java, per relazioni MODIFICABILI che specializzano quelle date in esercizio2. In particolare, si fornisca:
(a) Interfaccia API;
(b) Intestazione della classe ADT
(c) Stato Concreto;
(d) AF, I
(e) Additionals
(f) Definizione delle operazioni
(g) Si dica se sia possibile usare i casi di uso di esercizio2 per i nuovi valori introdotti.

Soluzione
(a) In file MuRelazioneJAPI.java in AltriEsercizi8lisngs, allegati
(b) eredita da classe in file MuRelazioneADTX.java
(c-h) In file MuRelazioneSADTX.java in AltriEsercizi8lisngs, allegati
(g) In file MainMuRelazioneSADTX.java in AltriEsercizi8lisngs, allegati

AltriEsercizi8 - Esercizio5

Esercizio

Si definisca in Java una tipo astratto per relazioni binarie, su tipi generici, come quelle in esercizio1 per valori IMMUTABLE e con le sole operazioni addPair, removePair, isIn. In particolare, si utilizzi l'interfaccia RelazioneJAPI.java nell'omonimo file allegato e si fornisca:
(a) Intestazione della classe;
(b) Stato Concreto;
(c) AF, I;
(d) Definizione dei metodi addPair, removePair, isIn, getUno;
(e) Trattamento degli Additionals (clone, equals, toString, elements);
(f) Una versione adatta dei casi d'uso discussi in esercizio1.

Soluzione
In file RelazioneADT.java in AltriEsercizi8lisngs, allegati