Afifa Saeed 825367

a.saeed@campus.unimib.it

Progetto C++ - Classe Set

Il Set

La classe **Set** è una classe **templata** sui parametri **T** ed **Eql**. Il parametro **T** rappresenta il tipo di dato che il **Set** andrà a contenere, mentre **Eql** è il funtore per il confronto di uguaglianza tra i dati di tipo **T**.

La scelta di **templare** Set anche sul **funtore** di **uguaglianza** è un esigenza emersa per evitare di limitare l'utente (in generale colui che andrà ad utilizzare il Set) nella definizione di un proprio criterio di confronto, sia per tipi di dati primitivi sia per tipi di dati complessi e sia per tipi di dati custom.

Ad **esempio**, nel caso di un std::string l'utente potrebbe voler operare il confronto su una sottostringa specifica a partire dalla stringa iniziale.

Un Set non può contenere duplicati, ma **l'ordine dei dati è indifferente**. Quindi due Set con la stessa dimensione e gli stessi dati anche se in posizioni differenti sono considerati uguali (**operator==**).

Esempio: s1 = [2,3,4,5] e s2 = [3,4,5,2] sono uguali.

Alcune indicazioni sui metodi principali

Il Set può contenere n elementi. Per semplificare l'utente nel distinguere le posizioni del Set, si è scelto di far partire **l'indice del Set da 1**. In questo modo la ricerca con, per esempio **l'operatore []**, risulta più semplificata e intuitiva.

A livello di codice, invece, all'interno della classe Set, per **convenzione** l'indice viene fatto **partire da 0**. Infatti, ogni volta che viene passato un indice alla classe Set questa lo deve **reinterpretare** in base al proprio stato interno.

Ad esempio: abbiamo un Set di 5 elementi, allora è possibile accedere al Set fino alla posizione 5 (che nella classe Set risulterà essere la posizione 4).

Il Set **non** prevede elementi **duplicati**, quindi alla chiamata della funzione membro **add**, nel caso in cui l'elemento da aggiungere sia già presente nel Set, questo viene **ignorato** senza dare alcuna notifica all'utente.

Lo stesso principio descritto per la add viene applicato anche per la **remove**. Anche in questo caso non viene data alcuna notifica all'utente nel caso in cui l'elemento da eliminare non è presente nel Set.

Funzioni globale

Le funzioni globali come **filter_out**, **operator+**, **operator-** possono generare un eccezione di **bad_alloc** in quanto devono effettivamente allocare dei dati e restituirli come output.

In questo caso, l'elemento in **output viene ritornato per copia**, in quanto non c'è la possibilità di ritornarlo per **Reference**, visto che il dato uscito di scope non esisterà più. Il **ritorno per puntatore** non è stato valutato in quanto la gestione della memoria per il Set diventerebbe responsabilità dell'utente: cancellazione Set dalla memoria e reset il puntatore a tale dato.