PILE E CODE

Angelo Di Iorio
Università di Bologna

Esercizio 1

- Implementare in Java una pila di oggetti generici di tipo `T`
- Realizzare la pila tramite puntatori
- Implementare le classi `Stack` per la pila e `StackItem` per gli oggetti nella pila
- `Stack` implementa il metodo `toString()` e le seguenti specifiche:

STACK
% Restituisce **true** se la pila è vuota **boolean** isEmpty()
% Inserisce v in cima alla pila push(ITEM v)
% Estrae l'elemento in cima alla pila e lo restituisce al chiamante ITEM pop()
% Legge l'elemento in cima alla pila ITEM top()

Esercizio 1: interfaccia Java

```
public interface IQueue<T> {

        public boolean isEmpty();

        public void push(T item);

        public T pop();

        public T top();

}
```

Esercizio 2

- Implementare una classe Java per gestire una coda di oggetti generici
- Realizzare la coda tramite un *vettore circolare*
- La classe `Queue` implementa il metodo `toString()` e le seguenti specifiche:

QUEUE
% Restituisce **true** se la coda è vuota **boolean** isEmpty()
% Inserisce v in fondo alla coda enqueue(ITEM v)
% Estrae l'elemento in testa alla coda e lo restituisce al chiamante ITEM dequeue()
% Legge l'elemento in testa alla coda ITEM top()

Esercizio 2: interfaccia Java

```
public interface IQueue<T> {

        public boolean isEmpty();

        public void enqueue(T item);

        public T dequeue();

        public T top();

}
```

Esercizio 3

- Fornire un'implementazione alternativa per la coda di oggetti generici (stessa interfaccia) ma realizzata tramite puntatori

- Discussione: differenze e aspetti simili?

Esercizio 4 (es. 4.9 sul libro)

- Il tipo di dato DEQUEUE (double ended queue) è una sequenza modificabile ad entrambi gli estremi, in cui è possibile inserire un elemento in testa o inserirlo in fondo, e cancellare un elemento dalla testa o cancellarlo dal fondo.
- Si fornisca in Java un'interfaccia per DEQUEUE ed una implementazione basata su puntatori.
- Discussione: qual è il costo computazionale delle operazioni?

Java Collection Framework

Interface	Hash Table	Resizable Array	Balanced Tree	Linked List	Hash Table + Linked List
Set	HashSet		TreeSet		LinkedHashSet
List		ArrayList		LinkedList	
Deque		ArrayDeque		LinkedList	
Map	HashMap		TreeMap		LinkedHashMap

Esercizio 5

- Usare `ArrayList`, `LinkedList` e `ArrayDeque` per eseguire in ordine le seguenti operazioni su una sequenza (lista/coda) di interi inizialmente vuota:
- Caso 1: *`enqueue(5), enqueue(3), dequeue(), enqueue(2), enqueue(8), dequeue(), dequeue(), enqueue(4)`*
- Caso 2: *`addFirst(3), addLast(8), addFirst(2), removeLast( ), addLast(7) addLast(4), removeFirst( ), removeFirst( )`*

- Quali operazioni sono possibili sulle tre strutture dati?
- Qual'è il risultato finale nei due casi precedenti?