Strutture Dati / Laboratorio di Algoritmi e Strutture Dati
Anno Accademico 2009/2010
18/6/10

Cognome e Nome:
Numero di Matricola:

Spazio riservato alla correzione

1	2	3	totale
/15	20	/25	/60

1. [15 punti] Scrivere la funzione

```
public static <E> PositionList<E>  createList(Stack<E>S)
```

nella classe **ExStack_18_6** fornita dalla docente.

Istruzioni per lo svolgimento dell'esercizio:

- La funzione deve restituire una lista palindroma la cui prima metà contiene gli elementi di **S** nello stesso ordine in cui appaiono in **S** (se letti dal bottom verso il top) e la cui seconda metà contiene gli elementi di **S** disposti in ordine inverso. Ad esempio se **S**=<a b c d>, dove il bottom è l'elemento più a sinistra, allora la funzione restituisce la lista <a b c d d c b a>.
- Se lo stack è vuoto la funzione deve lanciare l'eccezione StackEmptyException.
- La funzione deve lasciare inalterato il contenuto dello stack **S**.
- La funzione **NON** deve usare alcuna struttura dati al di fuori di **S** e della lista output. Nel caso in cui non venga soddisfatto questo requisito la funzione sarà valutatala massimo **7 punti**.

La classe di test **ExStack_18_6** deve essere inserita nel pacchetto in cui si trova l'interfaccia **Stack**.

2.[20 punti] Scrivere la funzione

```
public static <K,V> Iterable<K> maxElts(PriorityQueue<K,V> Q1,
        PriorityQueue<K,V> Q2, Comparator <K> C, int k)
```

nella classe **ExPQ_18_6** fornita dalla docente.

Istruzioni per lo svolgimento dell'esercizio:

- La funzione deve restituire una collezione iterabile delle **k** chiavi più grandi tra tutte quelle contenute in **Q1** e **Q2**. Si assuma che k è maggiore di 0 e che la relazione di ordine totale definita sull'insieme delle chiavi di **Q1** e **Q2** è la stessa specificata dal comparatore **C**.
- Se **k>|Q1|+|Q2|** allora la funzione deve restituire una collezione vuota.
- La funzione deve lasciare inalterate le due code a priorità.

La classe di test **ExPQ_18** deve essere inserita nel pacchetto in cui si trova l'interfaccia **PriorityQueue**.

3. [25 punti] Scrivere la funzione

```
public static <E> Iterable<Position<E>>collectElts(BinaryTree<E> T, int k, E x)
```

nella classe **ExBinaryTree_18_6** fornita dalla docente.

Istruzioni per lo svolgimento dell'esercizio:

- La funzione deve effettuare una visita ricorsiva **inorder** dell'albero **T** e restituire una collezione iterabile contenente i primi **k** nodi visitati contenenti **x**.
- Se nell'albero ci sono meno di **k** nodi contenenti **x** allora la funzione restituisce una collezione contenente tutti i nodi contenenti **x** (se l'albero è vuoto la collezione restituita è vuota).
- La funzione **NON** deve invocare funzioni che restituiscono o utilizzano collezioni/iteratori di **tutti** i nodi dell'albero. Nel caso in cui non venga soddisfatto questo requisito la funzione sarà valutata al massimo **12 punti**.

La classe di test **ExBinaryTree_18_6** deve essere inserita nel pacchetto in cui si trova l'interfaccia **BinaryTree**.