Esercizi sui TDA Stack e Queue

1. Scrivere la funzione public static String inverti(String s) che prende in input una stringa s e restituisce in output la stringa ottenuta invertendo s. La funzione deve usare come unica struttura dati ausiliaria uno stack.

2. Scrivere la funzione public static boolean checkParentheses(String s). La funzione prende in input una stringa che rappresenta un'espressione contenente sia parentesi tonde che parentesi quadre e restituisce **true** se e solo se le parentesi sono correttamente annidate.

3. Scrivere la funzione public static boolean isPalindrome(Sring s). La funzione prende in input una stringa e restituisce **true** se e solo se la stringa è palindroma. Ad esempio, la funzione restituisce **true** se riceve in input la stringa "anna" o la stringa "abcecba" e restituisce **false** se riceve in input la stringa "abc" o la stringa "abcdba".

4. Scrivere la classe SQueue che implementa l'interfaccia Queue mediante due stack. La classe SQueue deve avere solo 2 variabili di istanza, entrambe di tipo Stack. Analizzare la complessità dei metodi enqueue e dequeue.

5. Scrivere la funzione public Integer extract(Queue<Integer>Q, int k) che restituisce in output l'elemento di Q che si trova in k-esima posizione a partire dal front. Il metodo non deve utilizzare strutture dati ausiliare e deve lasciare inalterato il contenuto di Q. Se Q contiene meno di k elementi, allora la funzione deve lanciare l'eccezione **NotEnoughElements**.

6. Scrivere la classe QStack che implementa l'interfaccia Stack mediante una coda. La classe QStack deve avere come unica variabile di istanza una variabile di tipo Queue. Analizzare la complessità dei metodi pop(), top() e push().

7. Scrivere la funzione public static <E> void reverse(Queue<E> Q) che inverte il contenuto della coda Q.

8. Scrivere la funzione **ricorsiva** public static <E> void recReverse(Queue<E> Q) che inverte il contenuto della coda Q. La funzione non deve utilizzare alcuna struttura dati ausiliaria.

9. Scrivere la funzione **ricorsiva** public static <E> boolean isContained(Queue <E>Q,E x) che restituisce true se x è presente in Q e false altrimenti.

10. Scrivere il metodo toString() della classe LinkedQueue. La funzione non deve invocare i metodi dell'interfaccia Queue.