



## Problema 10

7 Aprile 2021

### Descrizione – Parte I

Definisci in Java una classe `StringSList` per rappresentare liste di *stringhe* nello stile di Scheme. Analogamente alla classe introdotta a lezione per le liste di interi, il protocollo deve prevedere i seguenti costruttori e metodi:

```
public StringSList()                // null
public StringSList( String e, StringSList sl ) // cons
public boolean isNull()             // null?
public String car()                  // car
public StringSList cdr()             // cdr
public StringSList cons( String e ) // cons (modalità alternativa)
public int length()                  // length
public String listRef( int k )       // list-ref
public boolean equals( StringSList sl ) // equal?
public StringSList append( StringSList sl ) // append
public StringSList reverse()         // reverse
public String toString()              // visualizzazione testuale
```

### Descrizione – Parte II

Al fine di verificare le funzionalità della classe `StringSList`, in un file separato definisci una procedura (metodo statico) che, data una stringa *btr* (oggetto di tipo `String`) che rappresenta un intero non negativo nella notazione BTR (Balanced Ternary Representation) e dato un intero non negativo *n* (valore di tipo `int`), restituisce la lista di *n* interi consecutivi in notazione BTR, a partire da *btr*. Per esempio, se *btr* = "+-" e *n* = 5, la lista risultante deve contenere le cinque stringhe "+-", "+.", "++", "+--" e "+-.", nell'ordine. Per generare le successive rappresentazioni in forma BTR applica la procedura `btrSucc` realizzata in Java nell'esercitazione precedente.

Infine, operando su liste di diversa lunghezza costruite in questo modo, verifica sperimentalmente che il risultato dell'applicazione di ciascuno dei metodi del protocollo pubblico di `StringSList` sia coerente con quello delle corrispondenti procedure predefinite di Scheme.