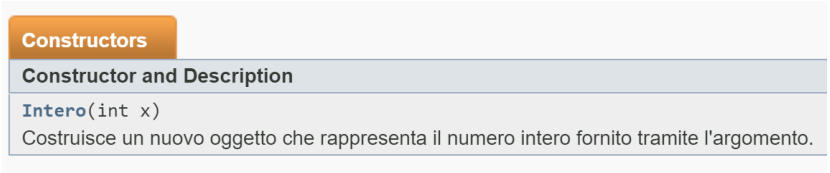
Esercizio 1/3

Il package `prog.util` fornisce una classe Intero le cui istanze rappresentano numeri interi. Tale classe mette a disposizione il seguente costruttore:



Constructors

Constructor and Description
`Intero(int x)` Costruisce un nuovo oggetto che rappresenta il numero intero fornito tramite l'argomento.

Fra gli altri, la classe mette a disposizione questi metodi:

`public int intValue()`

Restituisce il valore intero rappresentato dall'oggetto che esegue il metodo.

`public java.lang.String toString()`

Costruisce una stringa contenente il numero rappresentato dall'oggetto che esegue il metodo, espresso in lettere; ad esempio, se l'oggetto che esegue il metodo rappresenta il numero 34, il metodo restituirà la stringa `"trentaquattro"`.

Scrivete un'applicazione che legga due interi e ne calcoli la somma, L'applicazione dovrà scrivere l'operazione effettuata, esprimendo i numeri prima in cifre e poi in lettere (usando i metodi della classe `Intero`). Ad esempio un'esecuzione dell'applicazione potrebbe essere:

```
primo addendo? 4
secondo addendo? 5
4 + 5 = 9
Quattro + cinque = nove
```

Esercizio 2/3

La classe `String` mette a disposizione un metodo `compareTo` che riceve come argomento un riferimento a un oggetto di tipo `String` e restituisce un valore di tipo `int`. In particolare il metodo restituisce:

- zero se la stringa fornita come argomento è uguale a quella che esegue il metodo;
- un valore minore di zero se la stringa che esegue il metodo è lessicograficamente minore di quella fornita come argomento, cioè se la precede nell'ordine alfabetico
- un valore maggiore di zero se la stringa che esegue il metodo è lessicograficamente maggiore di quella fornita come argomento, cioè se la segue nell'ordine alfabetico

Utilizzando tale metodo scrivete un'applicazione che, lette due stringhe inserite dall'utente, indichi se sono uguali o diverse. Nel caso le stringhe siano diverse l'applicazione dovrà visualizzarle prima in ordine alfabetico e poi in ordine di lunghezza. Un esempio di esecuzione è:

```
Prima stringa? cane
Seconda stringa? bovino
Ordine alfabetico:
bovino
cane
Ordine di lunghezza:
cane
bovino
```

Esercizio 3/3

Scrivete un'applicazione che legga una sequenza di stringhe e visualizzi:

- la stringa lessicograficamente minima
- la stringa lessicograficamente massima
- la stringa più corta
- la stringa più lunga

tra quelle inserite dall'utente. Nel caso vi siano più stringhe della stessa lunghezza, l'applicazione può visualizzare una qualunque di queste. Si supponga che la sequenza venga terminata dalla stringa vuota (che non fa parte della sequenza).