

# Code kata

## Stringhe

## **Indice generale**

<b>1</b>	<b>Filtrare un elenco di stringhe (liste/vettori).....</b>	<b>3</b>
<b>2</b>	<b>Conversione di un vettore in stringa (vettori).....</b>	<b>4</b>
<b>3</b>	<b>Contare n° sotto stringhe in un testo.....</b>	<b>5</b>
<b>4</b>	<b>Settare a maiuscola la prima lettera delle parole di inizio periodo.....</b>	<b>6</b>
<b>5</b>	<b>Parsing di “stringhe numeriche” intere.....</b>	<b>7</b>
<b>6</b>	<b>“Splitting” di una stringa.....</b>	<b>8</b>

# 1 Filtrare un elenco di stringhe (liste/vettori)

## Problema

Data una lista di stringhe restituire una nuova lista contenente solo quelle che contengono almeno un carattere diverso da spazio (stringhe non vuote).

Esempio

```
List<string> lista = new List<string>() { "Roma", "", "Napoli", " ", "Firenze" };  
// -> "Roma", "Napoli", "Firenze"
```

## Variazioni

Dato un vettore di stringhe, restituire un vettore di stringhe.

Considerare la possibilità che il vettore/lista contenga stringhe nulle:

```
string[] lista = { "Roma", "", "Napoli", null, "Firenze" };  
// -> "Roma", "Napoli", "Firenze"
```

## Requisiti (necessari / utili) per la soluzione

Togliere gli spazi iniziali e finali di una stringa; metodo `Trim()`:

```
string s = " hello! ";  
string s2 = s.Trim(); // -> "hello!";
```

Conoscere la lunghezza di una stringa; proprietà `Length`:

```
string s = " hello! ";  
int lung = s.Length; // -> 8
```

Trasformare una lista in un vettore; metodo `ToArray()`:

```
List<string> lista = new List<string>() { "qui", "quo", "qua" };  
string[] vettore = lista.ToArray();
```

## 2 Conversione di un vettore in stringa (vettori)

### Problema

Dato un vettore di interi restituire una stringa che contenga i suoi elementi separati da ",".

Esempio:

```
int[] dati = { 1, 34, 102 };  
// -> "1;34,102"
```

### Variazioni

Generalizzare il procedimento per l'uso di qualsiasi separatore:

```
int[] dati = { 1, 34, 102 }; // usa " , " -> "1 , 34 , 102"  
int[] dati = { 1, 34, 102 }; // usa " | " -> "1 | 34 | 102"
```

Generalizzare il procedimento per l'elaborazione di un vettore vuoto (lunghezza zero):

```
int[] dati = { }; // -> ""
```

Fornire versioni del procedimento sia usando il ciclo **for** che il **foreach**.

### Requisiti (necessari / utili) per la soluzione

Convertire un numero in stringa; metodo `ToString()`:

```
int a = 10;  
string s = a.ToString(); // -> "10"
```

Concatenare due stringhe: operatore `+`:

```
string s = "hello";  
s = s + ", darling!"; // -> "hello, darling!"
```

Usare uno oggetto `StringBuilder`:

```
StringBuilder sb = new StringBuilder();  
sb.Append("hello");  
sb.Append(", darling!");  
string s2 = sb.ToString(); // -> "hello, darling!"
```

Eliminare determinati caratteri in coda (e in testa) di una stringa; metodo `Trim()`:

```
string s = "hello!!!";  
string s2 = s.Trim('!'); // -> "hello";
```

## 3 Contare n° sotto stringhe in un testo

### Problema

Dato un testo (stringa) e una sotto stringa, contare quante volte questa compare nel testo.

Esempio

```
string testo = "La donzelletta vien dalla campagna";
string s = "la";
// -> 1 (Il primo "La" non viene contato perché la "L" è maiuscola)
```

### Variazioni

Non fare distinzione tra maiuscole e minuscole (*case insensitive*).

```
string testo = "La donzelletta vien dalla campagna";
string s = "la";
// -> 2
```

### Requisiti (necessari / utili) per la soluzione

Lunghezza di una stringa; proprietà `Length`:

```
string testo = "Hello!";
int lung = testo.Length; // -> 6
```

Trovare la posizione di una sotto stringa in una stringa; metodo `IndexOf()`:

```
string testo = "Ciao Gianni, come ti vanno le cose?";
int indice = s.IndexOf("Gianni"); // -> 5
```

Esiste una versione di `IndexOf()` che consente di specificare l'indice del carattere dal quale iniziare la ricerca della sotto stringa:

```
string testo = "amore, amore, amore...";
int indice = s.IndexOf("amore", 6); // -> 7 ("amore" iniziale viene saltata)
```

Trasformare una stringa in minuscolo o maiuscolo; metodi `ToLower()` e `ToUpper()`:

```
string s = "Hello!";
string s1 = s.ToLower(); // -> "hello!"
string su = s.ToUpper(); // -> "HELLO!"
```

Confrontare due stringhe senza distinzione tra maiuscole e minuscole; metodi: `string.Compare()`, `Equals()`:

```
string s1 = "Casa";
string s2 = "casa";
bool uguali = s1 == s2; //-> false ("Casa" != "casa")
uguali = s1.Equals(s2, StringComparison.CurrentCultureIgnoreCase); //-> true (s1 == s2)
int r = string.Compare(s1, s2, true); //-> 0 (s1 == s2)
```

## 4 Settare a maiuscola la prima lettera delle parole di inizio periodo

### Problema

Dato un testo correggere la lettera iniziale delle parole di inizio periodo, impostandola a maiuscola se necessario. Restituire il testo corretto. Si suppone che un periodo sia terminato da:

. , ! o ?

Esempio:

```
string s1 = "non hai ancora finito? dobbiamo muoverci!";  
// -> "Non hai ancora finito? Dobbiamo muoverci!"
```

### Requisiti (necessari / utili) per la soluzione

Trasformare in maiuscolo una stringa; metodo `ToUpper()`:

```
string s = "hello!";  
s = s.ToUpper(); // -> "HELLO!"
```

Trasformare in maiuscolo un carattere; metodi `char.ToUpper()`:

```
char ch = "a";  
ch = char.ToUpper(ch); // -> 'A'
```

## 5 Parsing di “stringhe numeriche” intere

### Problema

Data una stringa contenente un numero intero, restituire l'equivalente valore numerico. (Realizzare un metodo simile a `int.Parse()`). Considera solo numeri positivi. Realizza il metodo in un modulo di nome `Intero`.

Esempio:

```
int a = Intero.Parse("245"); // a -> 245
```

### Variazioni ed estensioni

1. Considera la possibilità che la stringa non contenga un numero valido. In questo caso, occorre sollevare l'eccezione `FormatException`.
2. Estendi il processo ai numeri negativi.
3. Realizza un metodo analogo, che però non sollevi eccezioni ma funzioni in modo analogo a `int.TryParse()`.
4. Aggiungi la possibilità di processare valori espressi in formato binario, ottale, esadecimale. Il tipo di formato sarà indicato da un prefisso: `b` → binario, `o` → ottale, `x` → esadecimale:

```
int a = Intero.Parse("b1010"); // a -> 10
a = Intero.Parse("1010"); // a -> 1010
a = Intero.Parse("o1010"); // a -> 520
a = Intero.Parse("x1010"); // a -> 4112
```

### Requisiti (necessari / utili) per la soluzione

Indicizzare una stringa.

Ottenere il codice ASCII di un carattere. (È sufficiente utilizzarlo come un valore intero)

```
char c = '0';
int codiceAscii = c; // -> 48
```

Stabilire se un carattere è una cifra. Verificare se cade nell'intervallo '0' ↔ '9'. Alternativamente si può usare il metodo `char.IsDigit()`.

```
bool ècifra = char.IsDigit('0'); // -> true
```

Sollevare un'eccezione; nello specifico, l'eccezione `FormatException`:

```
throw new FormatException("Formato non valido");
```

## 6 “Splitting” di una stringa

### Problema

Data una stringa estrarre le parole che la compongono e restituirle in un vettore. (Realizzare un omologo del metodo `Split()`). Considera come carattere separatore il singolo spazio. Realizza il metodo in un modulo di nome `Stringa`.

Esempio:

```
string s = "Non hai ancora finito? Dobbiamo muoverci!";
string[] parole = Stringa.Split(s);
//-> "Non", "hai", "ancora", "finito?", "Dobbiamo", "muoverci!"
```

### Variazioni ed estensioni

1. Supponi che due parole possano essere separate da più di uno spazio.
2. Aggiungi la possibilità di specificare più caratteri separatori.

```
string s = "Non hai ancora finito? Avanti, dobbiamo muoverci!";
string[] parole = Stringa.Split(s, ' ', ',', '?', '!');
//-> "Non", "hai", "ancora", "finito", "Avanti", "dobbiamo", "muoverci"
```

3. Supportare che due parole possano essere separate da più di uno spazio (in generale da più di un carattere separatore).

### Requisiti (necessari / utili) per la soluzione

Ottenere una sotto stringa di una determinata lunghezza; metodo `SubString()`:

```
string testo = "Hello, darling!";
string s = testo.Substring(7); // -> darling!
s = testo.Substring(7, 3);    // -> dar
```

Ottenere l'indice di una sottostringa:

```
string testo = "Hello, darling!";
int indice = testo.IndexOf("dar"); // -> 7
```

Ottenere l'indice di una sottostringa a partire da un certo indice:

```
string testo = "Ok, mamma, ma non fare tardi!";
int indice = testo.IndexOf("ma"); // -> 4
indice = testo.IndexOf("ma", 5); // -> 11
```