Metodi iterativi con stringhe ed array

1. Scrivere un metodo che, dati un carattere `c` ed una stringa `s`, restituisce il numero delle occorrenze di `c` in `s`.

```
public static int occorrenzeCar (char c, String s) {
  int cont = 0;
  for (int i=0; i<s.length(); i++) {
    if (s.charAt(i) == c)
      cont++;
  }
  return cont;
}
```

2. Scrivere un metodo che, dati un carattere `c` ed una stringa `s`, restituisce `true` se `c` occorre in `s`, `false` altrimenti.

```
public static boolean occorreCar (char c, String s) {
  int i = 0;
  boolean trovato = false;
  while (i<s.length() && !trovato) {
    if (s.charAt(i) == c)
      trovato = true;
    i++;
  }
  return trovato;
}
```

3. Scrivere un metodo che, dati un carattere `c` ed una stringa `s`, restituisce la posizione della prima occorrenza di `c` in `s`, e `-1` se `c` non occorre in `s`.

```
public static int posizioneCar (char c, String s) {
  int i = 0;
  boolean trovato = false;
  while (i<s.length() && !trovato) {
    if (s.charAt(i) == c)
      trovato = true;
    else i++;
  }
  if (trovato)
    return i;
  else return -1;
}
```

4. (*Esame scritto del 10/12/2001, una versione dell'Esercizio 3.*)
Scrivere un metodo in linguaggio Java che data una stringa `s` e due caratteri `c1` e `c2` determini se il numero di occorrenze di `c1` in `s` sia uguale o meno al numero di occorrenze di `c2`.

```
public static boolean occorrenze (String s, char c1, char c2) {
  int count1 = 0, count2 = 0;
  for (int i=0; i<s.length(); i++) {
    if (s.charAt(i) == c1)
      count1++;
    if (s.charAt(i) == c2)
      count2++;
  }
  return (count1 == count2);
}
```

5. (*Esame scritto del 4/11/2002, soluzione iterativa per una versione dell'Esercizio 4*.)
Scrivere un metodo *iterativo* che, dato un array `a` di stringhe, restituisce `true` se `a` è ordinato in modo crescente rispetto alla lunghezza dei suoi elementi, e `false` altrimenti. Ad esempio, dato `a = {"ad", "cbah", "wqzxtr"}`, il metodo restituisce `true`.

```
public static boolean crescente (String[] a) {
  for (int i=0; i<a.length-1; i++) {
     if (a[i].length() >= a[i+1].length())
       return false;
  }
  return true;
}
```

6. (*Esame scritto del 4/11/2002, una versione dell'Esercizio 5*.)
Scrivere un metodo che, dato un array bidimensionale quadrato `a` di caratteri, restituisce la stringa ottenuta concatenando i caratteri presenti nel triangolo in alto a sinistra inclusa la diagonale. Ad esempio, data la matrice `a`

```
{{'a','b','c'},
 {'d','e','f'},
 {'g','h','i'}}
```

il metodo restituisce la stringa `"abcdeg"`.

```
public static String triangolo (char[][] a) {
  int m = a.length;
  String s = "";
  for (int i=0; i<m; i++)
    for (int j=0; j<m-i; j++)
      s = s+a[i][j];
  return s;
}
```

7. (*Esame scritto del 15/9/2004, Esercizio 3*.)
Scrivere un metodo che, dati un array a di stringhe ed un intero k ($k \geq 2$), restituisce *true* se in a esistono almeno k stringhe di uguale lunghezza, altrimenti il metodo restituisce *false*. Ad esempio, dato `a = {"leone","cane","gatto","pesce","elefante"}` e `k=3`, il metodo restituisce *true*.

```
public static boolean m (String[] a, int k) {
  int i = 0;
  int cont = 1;
  while (i < a.length) {
    int j = i+1;
    while (j < a.length) {
      if (a[i].length() == a[j].length()) {
        cont++;
      }
      if (cont >= k) {
        return true;
      }
      j++;
    }
    i++;
    cont = 1;
  }
  return false;
}
```