Generics, wildcard, iterable, iterateur

Exercice 1 - Générification

1 Générifier le code ci-dessous :

```
public static List listLength(List list) {
  ArrayList length=new ArrayList();
  for(int i=0;i<list.size();i++) {
    CharSequence seq=(CharSequence)list.get(i);
    length.add(seq.length());
  }
  return length;
}
public static void main(String[] args) {
  List l=Arrays.asList(args);
  System.out.println(listLength(l));
}
```

   1 En utilisant une variable de type `T`
   2 En utilisant la notation wildcard.

2 À quoi sert la constante `Collections.EMPTY_LIST` ?
   Comment peut-on l'utiliser dans l'implantation de la méthode `listLength()` ?

3 Changer l'implantation de la méthode `listLength()` pour utiliser la méthode `emptyList` de la classe `java.util.Collections`.

Exercice 2 - C'est loin la merge

On souhaite écrire une méthode permettant de fusionner deux listes `List` pour obtenir une liste contenant alternativement un élement de chaque liste.
La méthode devra s'assurer que les deux listes ont la même taille.

Quel est le profil de la méthode `merge` (le plus générique possible) sachant que le code suivant est valide.

```
List<String> list1=...
List<StringBuilder> list2=...
List<? extends CharSequence> result1=merge(list1,list2);
List<?> result2=merge(list1,list2);
```

Noter qu'il y a un bug dans eclipse :)
Implanter la méthode `merge` créant une nouvelle liste. La signature de la méthode devra être la plus générique possible.
Que se passe-t'il si une des liste est une`LinkedList` ?

Exercice 3 - Interval

On souhaite pouvoir écrire le code suivant :

```
for(int i:range(1,5))
  System.out.println(i); // affiche 1 2 3 4 5
```

1 Quel est le profil de la méthode `range` ?
2 Proposer une implantation évitant de créer un tableau correspondant à l'interval.

3 Combien le code donné en exemple effectue-t-il d'allocations ?
Et le code ci-dessous ?

```
for(int i:range(190,200))
  System.out.println(i);
```

4 Écrire l'implantation de cette méthode en utilisant des classes anonymes.