Programmation et modélisation orientées objet
(USAL5A & USAL2H)

Dictionnaire distant

1 Implantation d'un dictionnaire distant

Implanter un dictionnaire (interface `Map<K,V>`) consultable et modifiable à distance. Un requête d'insertion prend en paramètre une clé et une valeur qui doivent être ajouté au dictionnaire par le serveur. Le protocole, qui est de haut niveau, est représenté par les types de données suivants :

```
sealed interface Request<K, V>
record Put<K, V>(K key, V value) implements Request<K, V> {}
record Get<K, V>(K key) implements Request<K, V> {}
record Stop<K, V>() implements Request<K, V> {}


sealed interface Answer<V>
record OldValue<V>(V oldValue) implements Answer<V> {}
record Value<V>(V value) implements Answer<V> {}
```

Questions

1. Implanter les classes `Server<K,V>` et `RemoteMap<K, V>` (qui étendra la classe `AbstractMap<K,V>` fournie) avec les méthodes suivantes :

   ```
   public V get(Object key)
   public V put(K key, V value)
   ```

2. Implanter la classe `Client` (la méthode `run` fera quelques opérations simple sur un dictionnaire distant de type `RemoteMap<Integer, String>`).

3. Implanter la classe `Main` et tester votre code.

4. Etendre le protocole pour ajouter les méthodes suivantes :

   ```
   public int size()
   public boolean isEmpty()
   public boolean containsKey(Object key)
   public boolean containsValue(Object value)
   public V remove(Object key)
   ```

2 Implantation des canaux locaux

L'utilisation directe des `BlockingQueue` de la JDK pour implanter les canaux locaux ne permet pas de distinguer les extrémités (en lecture ou en écriture) des canaux. Ceci est une source d'erreur, que le compilateur ne peut pas détecter. Pour y remédier, nous introduisons les interfaces Java pour modéliser l'extrémité d'un canal en lecture (resp. en écriture) permettant d'échanger des types de donnée génériques :

```
interface ObjectReader<A> {
    A readObject() throws IOException;
}


interface ObjectWriter<B> {
    void writeObject(B arg) throws IOException;
}
```

Questions

1. Ecrire deux classes `LocalObjectReader<A>` et `LocalObjectWriter<B>` qui implantent respectivement les interfaces `ObjectReader<A>` et `ObjectWriter<B>` ci-dessus en utilisant l'interface `BlockingQueue<E>` de la JDK. L'exception `InterruptedException` sera propagée, mais convertie auparavant en `IOException`.

2. Modifier l'implantation de `RemoteList<E>` et de `RemoteMap<K,V>` pour utiliser ces interfaces (et leurs implantations locales).