**Esercizio 1**

L’oggetto sharedMap e’ una collection non thread safe. I TestWorker modificano il contenuto della mappa in maniera concorrente, aggiungendo, rimuovendo e cambiando il valore delle entry.

**Soluzione 1**: La modifica della mappa avviene in un blocco synchronized sull’oggeto condiviso sharedMap

private void updateMapSynchronized(Integer int1, Integer int5, Integer int10, String key) {  
 synchronized (*sharedMap*) {  
 if (counter == 0) {  
 // se counter=0, rimuovi dalla mappa se valore corrente=1  
 if (*sharedMap*.containsKey(key) && *sharedMap*.get(key).equals(int1)) {  
// if (!sharedMap.containsKey(key)) {  
// logErr("0: chiave non presente");  
// }  
// if (!sharedMap.get(key).equals(int1)) {  
// logErr("0: valore cambiato");  
// }  
  
 *sharedMap*.remove(key);  
 log("{" + key + "} remove 1");  
 }  
 } else if (counter == 1) {  
 // se counter=1, inserisci 1 se chiave non presente  
 if (!*sharedMap*.containsKey(key)) {  
// if (sharedMap.containsKey(key))  
// logErr("1: chiave presente");  
  
 *sharedMap*.put(key, int1);  
 log("{" + key + "} put 1");  
 }  
...

**Soluzione 2**: uso di ConcurrentMap

private final static Map<String, Integer> *sharedMap* = new HashMap<String, Integer>();  
private final static ConcurrentMap<String, Integer> *sharedMapConcurrent* = new ConcurrentHashMap<>();

private void updateMapConcurrentColl(Integer int1, Integer int5, Integer int10, String key) {  
 if (counter == 0) {  
 // se counter=0, rimuovi dalla mappa se valore corrente=1  
 if(*sharedMapConcurrent*.remove(key, int1))  
 log("{" + key + "} remove 1");  
 } else if (counter == 1) {  
 // se counter=1, inserisci 1 se chiave non presente  
 if(*sharedMapConcurrent*.putIfAbsent(key, int1) == null)  
 log("{" + key + "} put 1");  
 } else if (counter == 5) {  
 // se counter=5, aggiorna mappa con 5 se valore attuale=10  
 if(*sharedMapConcurrent*.replace(key, 10, int5))  
 log("{" + key + "} replace " + 10 + " with 5");  
 } else if (counter == 10) {  
 // se counter=10, aggiorna mappa con 10 indipendentemente dal valore attuale  
 *sharedMapConcurrent*.computeIfPresent(key, (k,v) -> {  
 log("{" + key + "} replace " + v.intValue() + " with 10");  
 return *sharedMapConcurrent*.put(k, int10);  
 });  
 }  
}

**Esercizio 2**

L’oggetto **sharedPhraes** e’ una collection non thread safe. I reader cercano di iterare sulla collezione mentre il writer ne cambia il contenuto aggiungendo nuovi elementi.

Questo causa il lancio di: **ConcurrentModificationException**

**Soluzione 1:** Sincronizzazione tramite **ReadWriteLock**.

public class S7Esercizio2 {  
 private final static ReadWriteLock *rwLock* = new ReentrantReadWriteLock();  
 private final static Lock *writeLock* = *rwLock*.writeLock();  
 final static Lock *readLock* = *rwLock*.readLock();

**Main Thread**: single writer

*writeLock*.lock();  
try {  
 S7Esercizio2.*sharedPhrase*.add(*getWord*());  
} finally {  
 *writeLock*.unlock();  
}

**ReadWorker**: many reader

S7Esercizio2.*readLock*.lock();  
try {  
 iterator = S7Esercizio2.*sharedPhrase*.iterator();  
 while (iterator.hasNext()) {  
 sb.append(iterator.next());  
 sb.append(" ");  
 }  
} finally {  
 S7Esercizio2.*readLock*.unlock();  
}

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Recognized Changes:** | **totCompares:** | **Avg compares per change** |
| ReadWorker00 | 11 | 334321 | 30392.818 |
| ReadWorker01 | 11 | 331467 | 30133.363 |
| ReadWorker02 | 11 | 328281 | 29843.727 |
| ReadWorker03 | 11 | 334271 | 30388.273 |
| ReadWorker04 | 11 | 326360 | 29669.092 |
| ReadWorker05 | 11 | 335869 | 30533.545 |
| ReadWorker06 | 11 | 336765 | 30615.000 |
| ReadWorker07 | 11 | 331347 | 30122.455 |
| ReadWorker08 | 11 | 334927 | 30447.908 |
| ReadWorker09 | 11 | 330305 | 30027.727 |
| ReadWorker10 | 11 | 341544 | 31049.455 |
| ReadWorker11 | 11 | 326611 | 29691.908 |
| ReadWorker12 | 11 | 325886 | 29626.000 |
| ReadWorker13 | 11 | 331046 | 30095.092 |
| ReadWorker14 | 11 | 335364 | 30487.637 |
| Simulation took: | 10245 ms |  |  |

**Soluzione 2:** uso di **Collections.synchronizedList**

S7Esercizio2.*sharedPhrase* = Collections.*synchronizedList*(list);

synchronized (S7Esercizio2.*sharedPhrase*) {  
 final Iterator<String> iterator = S7Esercizio2.*sharedPhrase*.iterator();  
 while (iterator.hasNext()) {  
 sb.append(iterator.next());  
 sb.append(" ");  
 }  
}

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Recognized Changes:** | **totCompares:** | **Avg compares per change** |
| ReadWorker00 | 11 | 161374 | 14670.363 |
| ReadWorker01 | 11 | 168075 | 15279.546 |
| ReadWorker02 | 11 | 172591 | 15690.091 |
| ReadWorker03 | 11 | 198426 | 18038.727 |
| ReadWorker04 | 11 | 158537 | 14412.454 |
| ReadWorker05 | 10 | 112990 | 11299.000 |
| ReadWorker06 | 11 | 137315 | 12483.182 |
| ReadWorker07 | 11 | 131556 | 11959.637 |
| ReadWorker08 | 11 | 167791 | 15253.728 |
| ReadWorker09 | 11 | 163288 | 14844.363 |
| ReadWorker10 | 11 | 144027 | 13093.363 |
| ReadWorker11 | 11 | 203971 | 18542.818 |
| ReadWorker12 | 10 | 128347 | 12834.700 |
| ReadWorker13 | 11 | 138026 | 12547.818 |
| ReadWorker14 | 10 | 102051 | 10205.100 |
| Simulation took: | 10328ms |  |  |

**Soluzione 3:** uso di **CopyOnWriteArrayList**

S7Esercizio2.*sharedPhrase* = new CopyOnWriteArrayList<>(list);

In questo caso il reader non ha bisogno di usare synchronized block perche’ l’iterator ci restituisce gia’ uno snapshot della collection che non subira’ modifiche.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Recognized Changes:** | **totCompares:** | **Avg compares per change** |
| ReadWorker00 | 11 | 374261 | 34023.727 |
| ReadWorker01 | 11 | 379099 | 34463.547 |
| ReadWorker02 | 11 | 375356 | 34123.273 |
| ReadWorker03 | 11 | 378100 | 34372.727 |
| ReadWorker04 | 11 | 373216 | 33928.727 |
| ReadWorker05 | 11 | 372253 | 33841.184 |
| ReadWorker06 | 11 | 384742 | 34976.547 |
| ReadWorker07 | 11 | 373451 | 33950.090 |
| ReadWorker08 | 11 | 374911 | 34082.816 |
| ReadWorker09 | 11 | 378785 | 34435.000 |
| ReadWorker10 | 11 | 372323 | 33847.547 |
| ReadWorker11 | 11 | 374587 | 34053.363 |
| ReadWorker12 | 10 | 373170 | 37317.000 |
| ReadWorker13 | 11 | 367798 | 33436.184 |
| ReadWorker14 | 10 | 377555 | 37755.500 |
| Simulation took: | 10176ms |  |  |