# Concurrency

## Threading Problems

# Liveness

- the ability of an application to be able to execute in a timely manner
- liveness problem occurs when application becomes unresponsive ("stuck")
  - these problems are result of a thread entering BLOCKING or WATING state

    (forever or altering between entering/exiting these states)
- for the exam you need to know three types of liveness issues:
  - **deadlock**, **starvation**, **livelock**

# Deadlock

- two or more threads are blocked forever
- because each thread is waiting on the other to complete

# Starvation

- a single thread is perpetually denied access to a shared resource or a lock
- the thread is still active, but is unable to complete its work
  - because other thread(s) are constantly taking the resource it's trying to access

# Livelock

- two or more threads are conceptually blocked forever
  - even thought each of them are active and is trying to complete its task
- this is special case of resource starvation:
  - two or more threads actively try to acquire a set of locks
  - and since they are unable to do so, the process is restarted
- in practice, livelock is difficult issue to detect
  - because threads in livelock state appear active and resposnive
  - but actually they are just stuck in an endless state

# Race Condition

- two tasks that should be completed sequentially are completed at the same time
- most common example is creation of unique username:
  - either both users will create an account with the same username
  - or neither user will be able to create an account and will get an error
  - or one user will be allowed a username, and other one will get an error
- neither of these outcomes are desirable

# Concurrent Collection Classes

| Class Name | Java Collection interfaces |
|---|---|
| ConcurrentHashMap | Map, ConcurrentMap |
| ConcurrentLinkedQueue | Queue |
| ConcurrentSkipListMap | Map, SortedMap, NavigableMap, ConcurrentMap, ConcurrentNavigableMap |
| ConcurrentSkipListSet | Set, SortedSet, NavigableSet |
| CopyOnWriteArrayList | List |
| CopyOnWriteArraySet | Set |
| LinkedBlockingQueue | Queue, BlockingQueue |

# Synchronized Collections methods

| |
|---|
| `synchronizedCollection(Collection<T> c)` |
| `synchronizedList(List<T> list)` |
| `synchronizedMap(Map<K,V> m)` |
| `synchronizedNavigableMap(NavigableMap<K,V> m)` |
| `synchronizedNavigableSet(NavigableSet<T> s)` |
| `synchronizedSet(Set<T> s)` |
| `synchronizedSortedMap(SortedMap<K,V> m)` |
| `synchronizedSortedSet(SortedSet<T> s)` |