# Caching Manager exercise

### **Step 1**

a) Implement a simple cache for storing objects of any type. The cache should have 4 methods:

1. put an object into cache using string value as the key
2. get object from cache by its key or get NULL if key does not exist
3. remove object from cache by its key
4. invalidate all objects in the cache

Notes:

* cache should not contain NULL values in keys and stored objects

b) Propose and implement test cases for cache's put / get methods.

c) Limit maximum size of the cache to 100 elements. Cache should not store the 101st element. Modify method signature(s) to fit this requirement.

### **Step 2**

a) Extend cache manager with ability to set priority for each stored object and invalidate objects based on this priority.

There are three priorities: low, medium, high. Objects with low priority will be kept in cache for 15 seconds, medium – 10 seconds, high – 5 seconds.

1. Provide a sample implementation that will invalidate objects based on insertion time and time to expire.
2. Isolate invalidation logic for cached objects into a separate interface. Interface should offer ability to register/unregister and check object validity
3. Solution should be extendable for other ways to validate / invalidate objects basing on priority.

Notes:

* by default stored object has high priority

### **Step 3**

* Adjust cache manager to work in a concurrent environment.