**Class Design for N-way set associative Cache**



**Design Considerations**:

* Simplistic design, but extendable/scalable
* Performance optimization is not one of the main consideration to make the design simple and as realistic as possible for the time limitation

**Implementation details**:

* Software is distributed to the users/clients as a (dynamic) library ***SetAssociativeCache.dll*** and as two header files ***SetAssociativeCache.h*** and ***Cache.h***
* ***SetAssociativeCache.h*** is defined as a thin template. So all the implementation details including internal data structures are totally hidden from the clients
* ***Cache.h*** has the class declaration for the above mentioned template based header to include
* Simple Array based Hash mechanism is used for maintaining and for easy look up for the Sets
* Simple Linked list (STL *list*) is used for maintaining and look up of the slots
* I avoided using boost library, that would give me heterogeneous containers
* For error handling exceptions can be used, which I have avoided because of the time constraints

**API specification**:

***SetAssociativeCache.h*** is documented to cover up how to use the APIs.

It exports 4 APIs:

This function should be the first one to use to set up the Cache that takes arguments like:

* Capacity of the Cache (though no check is done, preferably this is power of 2)
* Number that represents N-way Set (though no check is done, preferably this is power of 2)
* Replacement policy (LRU or MRU)

This function should be the first one to use to set up the Cache that takes arguments like:

void DestroyCache()

This function should be called in the end to free-up the Cache

template <class Key, class Value> void AddToCache(const Key& aKey, const Value& aValue)

Template based function that Adds key-value pair to the Cache

template <class Key, class Value> void RetrieveFromCache(const Key& aKey, Value& aValue)

Template based function that retrieves the value based on the key