Homework 2.1 (Implement a LRU cache)

```
class LRUCache {

  class DoubleLinkedNode {
    int key;
    int value;
    DoubleLinkedNode prev;
    DoubleLinkedNode next;
  }

  private void addNode(DoubleLinkedNode node) {
    node.prev = head;
    node.next = head.next;

    head.next.prev = node;
    head.next = node;
  }

  private void removeNode(DoubleLinkedNode node){
    DoubleLinkedNode prev = node.prev;
    DoubleLinkedNode next = node.next;

    prev.next = next;
    next.prev = prev;
  }

  private void moveToHead(DoubleLinkedNode node){
    removeNode(node);
    addNode(node);
```

```
}

private DoubleLinkedNode popTail() {
  DoubleLinkedNode res = tail.prev;
  removeNode(res);
  return res;
}

private Map<Integer, DoubleLinkedNode> cache = new HashMap<>();
private int size;
private int capacity;
private DoubleLinkedNode head, tail;

public LRUCache(int capacity) {
  this.size = 0;
  this.capacity = capacity;

  head = new DoubleLinkedNode();
  tail = new DoubleLinkedNode();
  head.next = tail;
  tail.prev = head;
}

public int get(int key) {
  DoubleLinkedNode node = cache.get(key);
  if (node == null) return -1;

  moveToHead(node);
```

```
    return node.value;
  }

  public void put(int key, int value) {
    DoubleLinkedNode node = cache.get(key);

    if(node == null) {
      DoubleLinkedNode newNode = new DoubleLinkedNode();
      newNode.key = key;
      newNode.value = value;

      cache.put(key, newNode);
      addNode(newNode);

      ++size;

      if(size > capacity) {
        DoubleLinkedNode tail = popTail();
        cache.remove(tail.key);
        --size;
      }
    } else {
      node.value = value;
      moveToHead(node);
    }
  }
}
```

Homework 2.2

View and stored procedure

A view represents a virtual table. You can join multiple tables in a view and use the view to present the data as if the data were coming from a single table.

A stored procedure uses parameters to do a function... whether it is updating and inserting data, or returning single values or data sets.

A Stored Procedure:

- Accepts parameters
- Can NOT be used as building block in a larger query
- Can contain several statements, loops, IF ELSE, etc.
- Can perform modifications to one or several tables
- Can NOT be used as the target of an INSERT, UPDATE or DELETE statement.

A View:

- Does NOT accept parameters
- Can be used as building block in a larger query
- Can contain only one single SELECT query
- Can NOT perform modifications to any table
- But can (sometimes) be used as the target of an INSERT, UPDATE or DELETE statement.

Materialized view and view

Sr. No.	Key	Views	Materialized Views
1	Definition	Technically View of a table is a logical virtual copy of the table created by “select query” but the result is not stored anywhere in the disk and every time we need to fire the query when we need data, so always we get updated or latest data from original tables.	On other hand Materialized views are also the logical virtual copy of data-driven by the select query but the result of the query will get stored in the table or disk.
2	Storage	In Views the resulting tuples of the query expression is not get storing on the disk only the	On other hand in case of Materialized views both query expression and resulting tuples of the query get stored on the disk.

Sr. No.	Key	Views	Materialized Views
		query expression is stored on the disk.	
3	Query Execution	As mentioned above in case of Views the query expression is stored on the disk and not its result so query expression get executed every time when user try to fetch data from it so that user will get the latest updated value every time.	While on other hand in case of Materialized Views the result of query is get stored on the disk and hence the query expression did not get executed every time when user try to fetch the data so that user will not get the latest updated value if it get changed in database.
4	Cost Effective	As Views does not have any storage cost associated with it so they also does not have any update cost associated with it.	On other hand Materialized Views does have a storage cost associated with it so also have update cost associated with it.
5	Design	Views in SQL are designed with a fixed architecture approach due to which there is an SQL standard of defining a view.	On other hand in case of Materialized Views in SQL are designed with a generic architecture approach so there is no SQL standard for defining it,and its functionality is provided by some databases systems as an extension.
6	Usage	Views are generally used when data is to be accessed infrequently and data in table get updated on frequent basis.	On other hand Materialized Views are used when data is to be accessed frequently and data in table not get updated on frequent basis.