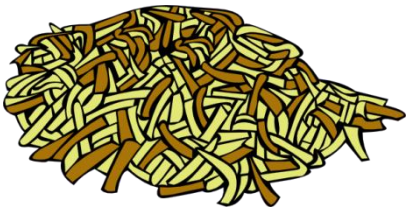
Hash Table Lab

"What I cannot create, I do not understand."



After finishing each part of the lab, copy your entire project and work on the copy for the next part!

Part 1: Implement a simple *HashTable* class.

- All methods take & return *Object* types, but for this lab, you will store < *Integer*, *String*> objects.
- Write a driver routine (*main* method) to:
  - Create a *HashTable* object
  - Read a text file containing < *Integer*, *String*> word pairs
  - Save them to the table.
  - Implement a *toString* method returning the saved objects, ordered by bucket index
  - Print the resulting table.
- Implement a simple *Node* class:

```
public class Node
--------------------------------------------------------------------------
          Node()                  // set key & value to null
          Node(Object key,
               Object value)

  String  toString()              // return a formatted string for the key & value
```

  - Make your fields public
- Implement a simple *HashTable* class:

```
public class HashTable
--------------------------------------------------------------------------
          HashTable()             // set default table size to 101
          HashTable(int initCap)
  Object  put(Object key,         // the previous value associated with key,
              Object value)       //     or null if there was no mapping for key

  Object  get(Object key)         // Returns the value to which the specified key is mapped,
                                  //     or null if this map contains no mapping for the key

  String  toString()              // return a formatted string, ordered by bucket index
```

  - Assume the *initCap* parameter is prime
  - For *put* & *get*, assume there are no collisions.
  - For the *put* method, use the input parameters to build a *Node* object
  - For the *get* method, unwrap the *Node* object & return the value
  - When determining the hash index, call the *hashCode* method on the key (external call), then mod with the table size to find the array index
  - For *toString*, make sure to order <*key*, *value*> pairs by array index.
- Test your program by running the *main* method on a small table.
  - Use only non-colliding keys & valid search keys
  - Calculate by hand to validate.