Ryan Goldstone

[ryangold@bu.edu](mailto:ryangold@bu.edu)

CS 342 – Assignment #3 Pseudo Code

\*All classes using MusickLib package\*

**MusickLib Class**

public static void main (String[] args)

* create new OpenAddressHashtable object
* create new ChainedHashtable object
* give both a size of 35

public static void populateDate(MusicCollection collection)

* create array of strings with songs and artist for the collection
* create array of strings with songs and artist to delete
* add songs and then show the hash table
* delete songs and then show the hash table
* add more songs and show the hash table
* perform trace search for specific song and show if found or not found for song that exists in collection
* perform trace search and show if found or not found for song that does not exist in collection

**MusicCollection Class**

* create MusicCollection as an interface
* add, delete, search, showHash, and traceSearch functions

**Composition Class**

public CompositionName()

* create ways to set, get, equals value names

public ing getKey()

* find a way to generate a unique key for the items

public Boolean equals(Compositiion c)

* true/false to check for equals

public String toString()

* return key and name

**OpenAddressHashtable Class**

public class OpenAddressHashtable

* implement MusicCollection

public void showHash

* iterate through all elements in hashtable
* print out empty or deleted if needed

public void delete(String composition)

* search the index
* if element is found, delete it
* mark it as deleted

public Boolean traceSearch(String composition)

* create coefficient for quadratic function
* create hash function value for this element
* search through the hash table
  + check if empty or deleted
  + compare if found
  + if equals adjust coefficient
  + if not found, add to table and compute new hash
* return false if not found

public int search(String composition, Boolean searchDeleted)

* confirm not an empty cell
* compare with current entry if not empty
* if equals, return index
* adjust coefficient
* return -1 if not found

public boolean isEmptyCell(int index, boolean searchDeleted)

* check for empty cells

private int calculateNextHash(composition c, int coef)

* create unique value for next hash

public void add(String composition)

* search the index
* create a new entry
* place in empty cell and adjust coefficient
* if no available cells found, run quadratic probing to compute new hash
* print out collision if trying to add to occupied space

**ChainedHashtable Class**

public class ChainedHashtable

* implement Music Collection

public void showHash

* iterate through all the elements
* print out the table

public void delete(String composition)

* search for the element
* if found, mark it as deleted

private int calculateHash(String composition)

* calculate the index

public boolean traceSearch(String composition)

* create new element and hash function value
* compare items in list with current search
* return -1 if not found

public int search(String composition, boolean searchDeleted)

* search through hash table to find element
* return -1 if not found

public void add(String composition)

* search for item in hash table
* if not found, create new linked list
* print out collision if trying to add to occupied space