**Public class OpenHash{**

**Public OpenHash(int size)**

Initialize arraySize to size

Initialize hashArray as a new DataItem array of size arraySize

Initialize probeArray as new int array of size 9

Initialize probeElem to 0

Initialize foundArray as new int array of size 6

Initialize foundElem to 0

//end constructor

**Public void displayTable(int table)**

For (int i =0, while i is less than arraySize, increment i)

If (value at hashArray index i is not null)

If(key value at hashArray index i is null)

Print index number, deletion mark, and last name at index i

Else

Print index number, and last name at index i

//end if

Else

Print index number and whitespace

//end if

//end for

//end displayTable()

**Public int hashFunc(int key)**

Return key mod arraySize

//end hashFunc()

**Public void insert(DataItem item)**

Initialize local key to item’s key

Initialize hashVal to int returned by hashFunc and key

While(value in hashArray at hashVal index is not null and key at same index is not null)

Increment hashVal

Wrap around table by modding hashVal by arraySize

//end while

Set value in hashArray at hashVal index to item

//end insert()

**Public void insertUpdate(DataItem item)**

Initialize local key to item’s key

Initialize hashVal to int returned by hashFunc and key

Initialize probes to 1

While(value in hashArray at hashVal index is not null and key at same index is not null)

Increment hashVal

Wrap around table by modding hashVal by arraySize

Increment probes

//end while

Set value in probeArray at probeElem index to probes, increment probeElem

Set value in hashArray at hashVal index to item

//end insert()

**Public DataItem delete(DataItem item)**

Initialize nonItem as new DataItem with key value -1 and president object of item

Initialize local key to item’s key

Initialize hashVal to int returned by hashFunc and key

Initialize probes to 1

While (value in hashArray at hashVal index is not null)

If (key of value in hashArray at hashVal index matches local key)

Initialize temp as DataItem with value of item in hashArray at hashVal index

Set value in hashArray at hashVal to nonItem

Set value in probeArray at probeElem index to probes, increment probeElem

Set value in foundArray at foundElem index to 1, increment foundElem

Return temp

//end if

Increment probes

Increment hashVal

Wrap around table by modding hashVal by arraySize

//end while

Set value in probeArray at probeElem index to probes, increment probeElem

Set value in foundArray at foundElem index to 0, increment foundElem

Return null

//end delete()

**Public int[] getProbes()**

Return probeArray

//end getProbes()

**Public int[] getFoundArray()**

Return foundArray

//end getFoundArray

//end OpenHash Class

**Public class ChainHash{**

**Public ChainHash(int size)**

Initialize arraySize to size

Initialize hashArrayas new SortedList array of size arraySize

For (int i =0; while i is less than arraySize; increment i)

Initialize value at index i of hashArray to a new SortedList

//end for

//end constructor

**Public void displayTable()**

Print an empty line

For (int i = 0; while i is less than arraySize; increment i)

Print index number i

Display the list at hashArray index i

//end for

//end displayTable

**Public int hashFunc(int key)**

Return key mod arraySize

//end hashFunc

**Public void insert (Link link)**

Initialize local key to key of link

Initialize hashVal to int returned by hashFunc and key

Insert link into hashArray at hashVal index

//end insert

**Public void delete (Link link)**

Initialize local key to key of link

Initialize hashVal to int returned by hashFunc and key

Delete link from hashArray at hashVal index

//end delete

//end ChainHash class

**Public class SortedList{**

**Public SortedList()**

Initialize first to null

//end constructor

**Public void insert(Link link)**

Initialize local key to link’s key

Initialize Link previous to null

Initialize Link current to first

While(current is not null and key is less than current’s key)

Set previous to current

Set current to current’s next link

//end while

If (previous is null)

Set first to link

Else

Set previous’s next link to link

//end if

Set link’s next link to current

//end insert

**Public Link Delete(Link link)**

Initialize local key to link’s key

Initialize Link current to first

While(current is not null and current’s key is less than or equal to key)

If (current’s key is equal to key)

Set current’s key to null

Return current

//end if

Set current to current’s next link

//end while

Return null

//end delete()

**Public void displayList()**

Initialize Link current to first

While (current is not null)

Display current link

Set current to current’s next link

//end while

Print a blank line

//end displayList()

//end SortedList Class