Lab 6

CSE 274

A programmer has developed a `HashTableChain` class which implements a Hash Table data structure with chaining mechanism. Create a new project in Eclipse and import the `Application.java` file.

I. The HashTableChain class

The `HashTableChain` class in `Application.java` has the following methods:

```
public HashTableChain(int size)
```

The above constructor creates an `InternalArray` of size `arraySize`. Each entry of the `InternalArray` stores an object of a `DoublyLinkedList` class.

```
public void displayChains()
```

The above method iterates over all entries of the `InternalArray` and for each entry, calls the `displayForward` method of the object that is stored in the entry.

```
 public int hashFunc(String key)
```

The above method is a hash function.

```
public void insertInaChain(String key)
```

The above method receives a `key`, calls the `hashFunc` to obtain the `hashIndex` corresponding to the `key`, and calls the `insertNoDuplicate` method of the object that is stored at `hashIndex` of the `InternalArray`.

```
public void deletefromaChain(String key)
```

The above method receives a `key`, calls the `hashFunc` to obtain the `hashIndex` corresponding to the `key`, and calls the `delete` method of the object that is stored at `hashIndex` of the `InternalArray`.

```
public boolean findinChains(String key)
```

The above method receives a `key`, calls the `hashFunc` to obtain the `hashIndex` corresponding to the `key`, and calls the `find` method of the object that is stored at `hashIndex` of the `InternalArray`.

II. The DoublyLinkedList class

For the `HashTableChain` to work, there is a need for a `DoublyLinkedList` class with the following methods:

```
public boolean find(String key)
```

The above method receives a `key` and `return true` only if there is a link in the linked list that has the `key`.

```
public void insertNoDuplicate(String id)
```

The above method receives a `key` and checks if there is a link in the link list that has the `key`. If there exists such a link the method `return`. If there exists no such a link, the method creates a new link containing the `key` and inserts the link at the end of the linked list.

```
public void delete(String key)
```

The above method receives a `key`. If there is a link containing the `key`, the above method deletes the link from the linked list. If there are multiple links containing the `key`, the method deletes the first link that is closer to the beginning part of the linked list.

```
public void displayForward()
```

The above method displays the data inside the links of the linked list, starting from the first link.

III. The `Link` Class

The links in linked lists of `DoublyLinkedList` class of Section II are objects of a `Link` class. The `Link` class has the following variables:

```
public String SData;
public Link next;
public Link previous;
```

Also, the `Link` class has the following methods:

```
public Link(String id)
public void displayLink()
```

IV. Question

Develop the `DoublyLinkedList` and the `Link` classes with the specifications provided above. The `Link` class should have only the methods discussed above, but the `DoublyLinkedList` can have additional methods if needed. The developed `DoublyLinkedList` and `Link` classes should make the `HashTableChain` class fully functional. To test the developed classes, erase the body of the `main` method, copy the following lines of code into the body of the `main` method, run the application, and make sure that you see the expected output:

```
HashTableChain myhashtable = new HashTableChain(52);

myhashtable.insertInaChain("AK");
myhashtable.insertInaChain("FL");
myhashtable.insertInaChain("IA");
myhashtable.insertInaChain("MN");
myhashtable.insertInaChain("NV");
myhashtable.insertInaChain("SC");
myhashtable.insertInaChain("VT");
myhashtable.insertInaChain("ND");
myhashtable.insertInaChain("ME");

myhashtable.deletefromaChain("CO");
myhashtable.deletefromaChain("ME");

System.out.println(myhashtable.findinChains("ME"));

myhashtable.displayChains();
```

The expected output is printed below:

```
false
Chain 0:
Chain 1:
Chain 2:
Chain 3:
Chain 4:
Chain 5:
Chain 6:
Chain 7:
Chain 8: IA
Chain 9:
Chain 10: AK
Chain 11:
Chain 12:
Chain 13:
Chain 14:
Chain 15:
Chain 16: FL ND
Chain 17:
Chain 18:
Chain 19:
Chain 20: SC
Chain 21:
Chain 22:
Chain 23:
Chain 24:
Chain 25: MN
Chain 26:
Chain 27:
Chain 28:
Chain 29:
Chain 30:
Chain 31:
Chain 32:
Chain 33:
Chain 34: NV
Chain 35:
Chain 36:
Chain 37:
Chain 38:
Chain 39:
Chain 40: VT
Chain 41:
Chain 42:
Chain 43:
Chain 44:
Chain 45:
Chain 46:
Chain 47:
Chain 48:
Chain 49:
Chain 50:
Chain 51:
```

V. Submitting the Assignment

Keep the above lines of code in the body of the `main` method and submit the `Application.java` file.