EEM 480 Homework-2 Report

The homework was developed in Apache NetBeans IDE 12.0 using JDK8-301. The total facebook package contains 7 Java classes with main class and 1 java interface declaration.

In the main class there is a "obj" object declaration from ReadandParse class. With the "obj" object I called a method get_func() in infinite loop. In the data structure, there are 2 different linked list. Friendlist is a linked list that holds person's friends. The second one is a linked list that holds persons.

`public class ReadandParse()`

In the constructor block myfile, reader and database objects created respectively from File, Scanner and Database classes. Myfile is generated with the path given "src/facebook/Input.txt"

1. `public void get_func():`
   Gets the data from get_line method then parses the data with split. Used Switch – case for decision part. Param1 symbolizes the name1 after the command value. Name2 is other variable for methods that process with 2 names. Case 'I' uses database object's method called SearchName to get the person value. If it is not null, then that means the object is already created just increase the hitcount. Else it generates a new object and pushes the object to the database linkedlist with AddPerson method. Case 'F' uses two parsed variables called param1 and param2 then calls the method called AddFriend. Case 'D' checks the person if it is valid. If it is not, then put an error statement. If valid then use database method to call DeletePerson. Case 'P' if else statements used to check the person if it is in the database, if its friendlist is empty finally, uses person object's method called GetFriendNames to get friends it has. Case 'W' uses database method to FindMostPopular. Case 'O' uses database method called OutputList to print all the database list with the hitcount . Case 'X' closes the reader object and uses System.exit to stop java runtime environment. There is no loop in this method so that complexity is $\Theta(1)$.
2. `private String get_line()`
   Uses the reader object to use nextLine method. It gets the lines one by one and return the data everytime the method called. There is no loop in this method so that complexity is $\Theta(1)$.

`public class Person extends Friend`

In the constructor block Name, HitCount and FriendList objects created respectively from String, Integer and Friend classes. Friendlist is a linked list which holds the names of person's friends.

There are pre-defined methods used in the Homework-2.pdf.

1. `public void Append(Person new_data)`
   I prefered this method to put the nodes to the last. Friend node created and the data which is Person type put inside the node. Basic getter& setter methods used during this process. If the Linked List is empty, then the new node set as head. Return statemens used so that the code will exit after if. If the previous statement is not verified then Friendlist node's pointer will set as null then last node will be created to find last object in the linkedlist. Finally the last will point to the new node. In worst case the method goes to the nth node. There is a single loop in this method so that complexity is $\Theta(n)$.

2. `public boolean SearchInFriendList(String x)`
   Searches the friendlist with the key called x. Checks all the nodes and gets the names of these nodes. Compares the node.name with the given key. If the key matches, then returns true. In worst case the method goes to the n and returns false. There is a single loop in this method so that complexity is $\Theta(n)$.

3. `public int GetFriendListCount()`
   The method iterates in the linked friendlist and increases the counter with 1 for each friend found. In worst case the method goes to the n and returns count. There is a single loop in this method so that complexity is $\Theta(n)$.

4. `public String GetFriendNames()`
   The method iterates in the linked friendlist and concatenate with string for each friend found. In worst case the method goes to the n and returns friend_names variable as String. There is a single loop in this method so that complexity is $\Theta(n)$.

`public class CustomLinkedList`

In the constructor block head object created from Node class, a linked list which holds the names of person's friends. Node class is a class that holds data and link respectively from Person and Node classes. An object that can be created from CustomLinkedList class is a linked list that holds all the persons in the database.

1. `public void Append(Person new_data)`
   This method is the same method used in person class. All the process and the complexity are same. In worst case the method goes to the nth node. There is a single loop in this method so that complexity is $\Theta(n)$.

2. `public void printList()`

Iterates in the linked list. Calls the method called ContentOut which is pre-defined method in person class. The method always goes to the nth node. There is a single loop in this method so that complexity is Θ(n).

3. `public Person search(String x)`

   This method searches the key x in all the nodes in linked list. If it is found, then return a Person object called person dummy. If it cannot be found, then return null. In worst case the method goes to the nth node. There is a single loop in this method so that complexity is Θ(n).

4. `public Person SearchPopular()`

   It iterates in the database's linked list and pulls the hit_count. Multiplies the hit_count with the person's friends count. Person's friends count found by using GetFriendListCount method. Maximum number of popularities is chosen with the multiplication of that two number. Returns the most popular person in the database. In worst case the method goes to the nth node. There is a single loop in this method so that complexity is Θ(n).

5. `public String GetPersonNamesFromFriendName(String param1)`

   This method checks the param1 if it is in the some person's friendlist or not. If it is in someone's friendlist then takes that person's name in a string. For each person that it is in their friendlist it concetanetes its string. Returns the string. The method always goes to the nth node. There is a single loop in this method so that complexity is Θ(n).

6. `void deleteNode(Person key)`

   Deletes the nodes with the key. Links the previous node to the next node. So that the deleted node is taken out from the linked list. In worst case the method goes to the nth node. There is a single loop in this method so that complexity is Θ(n).

7. `public String DatabasetoString()`

   Iterates in the linked list. Concatenates all the nodes in the linked list as a string then returns its string value. The method always goes to the nth node. There is a single loop in this method so that complexity is Θ(n).

`public class DataBase implements DataBaseInterface`

Linker object created from CustomLinkedList class. DataBase class implements its methods from DataBaseInterface.java.

1. `public Person SearchName(String tName)`

   Using linker object search method used to check if the person is valid. There is no loop in this method so that complexity is Θ(1).

2. `public void OutputList()`
   Using linker object printList method used to print all the nodes with their corresponding hit count values. There is no loop in this method so that complexity is $\Theta(1)$.

3. `public boolean AddPerson(Person tNewPerson)`
   Using linker object Append method used to add the person object into the linked list. Main data structure generates here. There is no loop in this method so that complexity is $\Theta(1)$.

4. `public boolean DeletePerson(String pName)`
   Using linker object GetPersonNamesFromFriendName method used to pull all the nodes names into the "person_names" string. Then search the if the person with its given String pName and puts the person object into the variable. If the person_names string is empty, then this means the person is not in someones friendlist. So, it deletes the node by using deleteNode method with linker object. If the person is in someone's friendlist, then it prints all the person that the person is in it.

5. `public String toString()`
   Initializes an empty string than set this this "persons" string with the return value of DatabasetoString method. Returns its String "persons" value to the where it is called. There is no loop in this method so that complexity is $\Theta(1)$.

6. `public boolean AddFriend(String Name1, String Name2)`
   Searches the if the person objects with its given String pName and puts the person object into the variable. If person1 and person2 are not null, then searches person1's friendlist with the key of person2's name2 variable. If the flag is true, then they are already friends print no need to add. Otherwise add the person2 to person1's friend list. There is no loop in this method so that complexity is $\Theta(1)$.

7. `public void FindMostPopular()`
   Using linker object SearchPopular method used to check if the person is valid. Then prints the person object called most_popular's name as an output. There is no loop in this method so that complexity is $\Theta(1)$.