Amusement park
CIRCUS
Candy Cotton

Data Structures

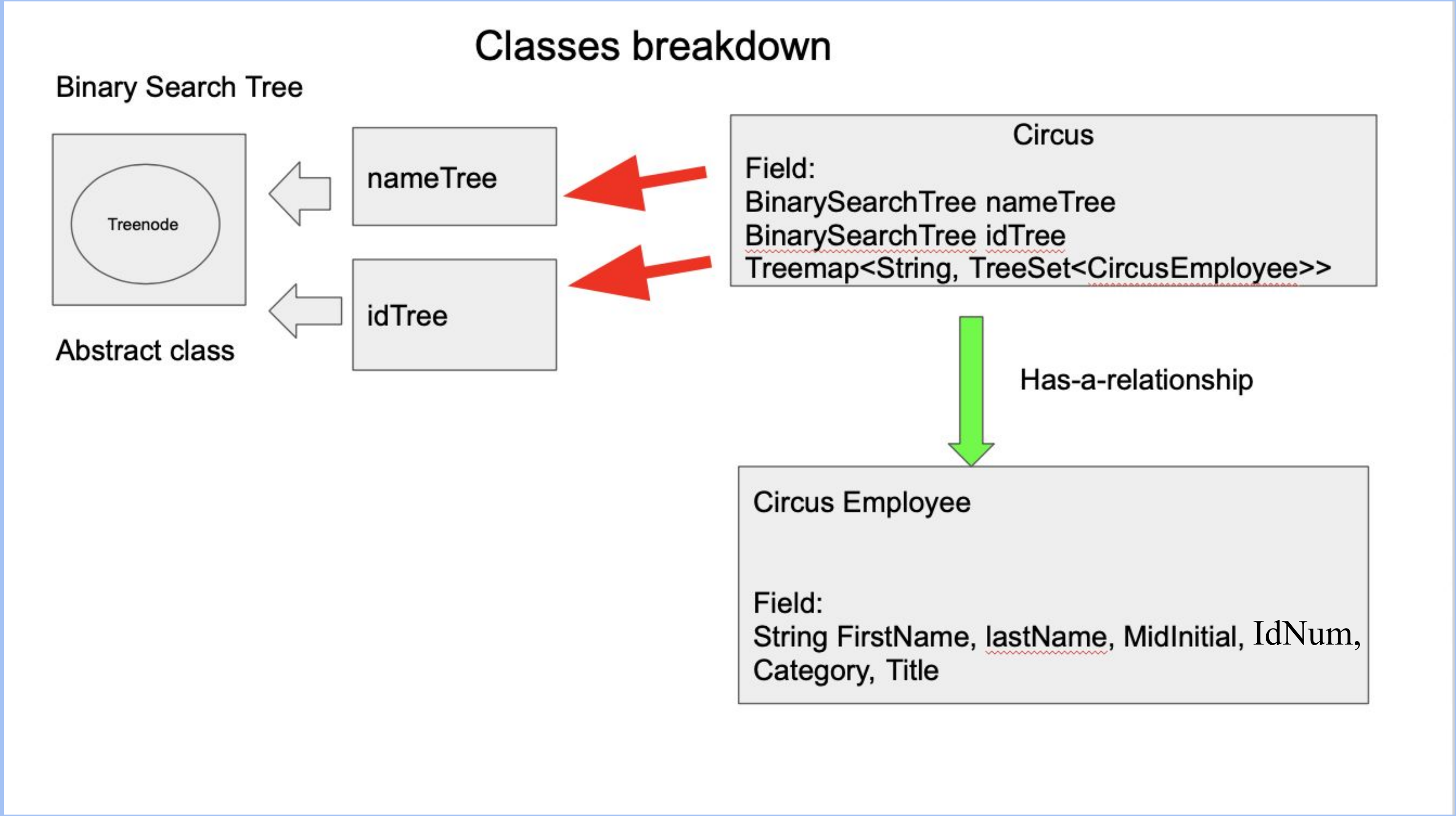
2 binary search trees of Circus Employees

- One tree is sorted by name, the other is sorted by idNum
- Left is less or equal than its parent, right is greater than the parent

1 TreeMap<String, TreeSet<Employee>>

- the keys are categories, the TreeSet is sorted by name

Data Structures



Read from a file

1) Throw FileNotFoundException
2) Make a scanner that gets one line from the file
   - Scanner input = new Scanner(new File ("CircusEmployees.txt”));
   - String line = input.nextLine();
3) Have another scanner that reads each token in that line
   - Scanner input2 = new Scanner(line);
4) Store each token in a variable and make an instance of Circus Employee with those variables
   - firstName = input2.next(), lastName = input2.next(), ….
5) Add the CircusEmployee to name tree, id tree, and TreeMap
6) Repeat 2-5 until there is no more lines to read

n is how many employees there are

T(Add) = T(add to name tree) 0+ T(add to id tree) + T(add to TreeMap)

$= O(n\log(n)) + O(n\log(n)) + O(n\log(n))$

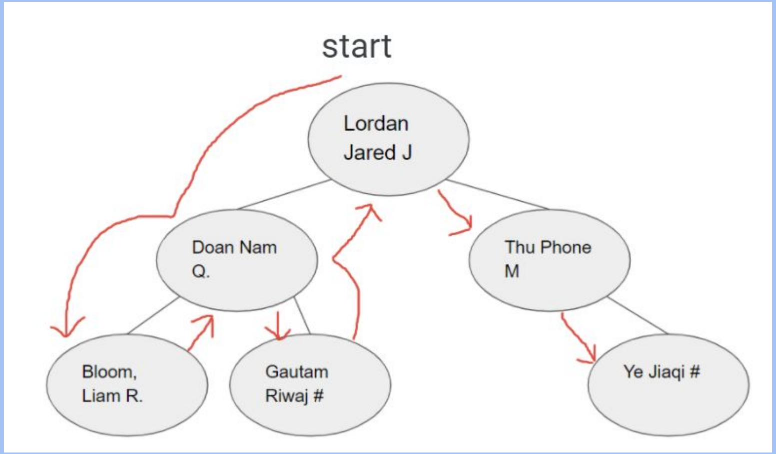
$= O(n\log(n))$

1) Print Alphabetically

1. Use the binary search tree that is sorted by name alphabetically
2. Print the tree with inorder traversal

T(print alphabetically) = T (print the name Tree)

= O(n)



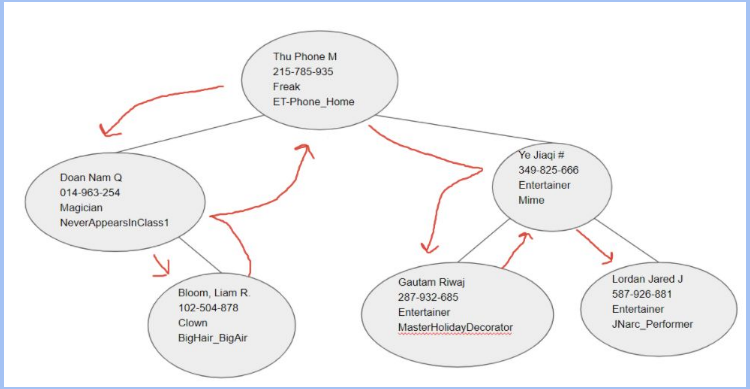
Output:
Bloom Liam R, Doan Nam Q, Gautam Riwaj #, Lordan Jared j, Thu Phone M, Ye Jiaqi #

2) Print IdNum

1. Use a binary search tree that is sorted by IdNum
2. Print the tree with inorder traversal

T(print by id) = T (print the id Tree)

= O(n)



Output:
014-963-254, 102-504-878, 215-785-935, 287-932-685, 349-825-666, 587-926-881

3) Insert a new Employee

1. Add Employee to the name tree and Id tree, which will automatically be sorted into the trees

2. Add it to the TreeMap

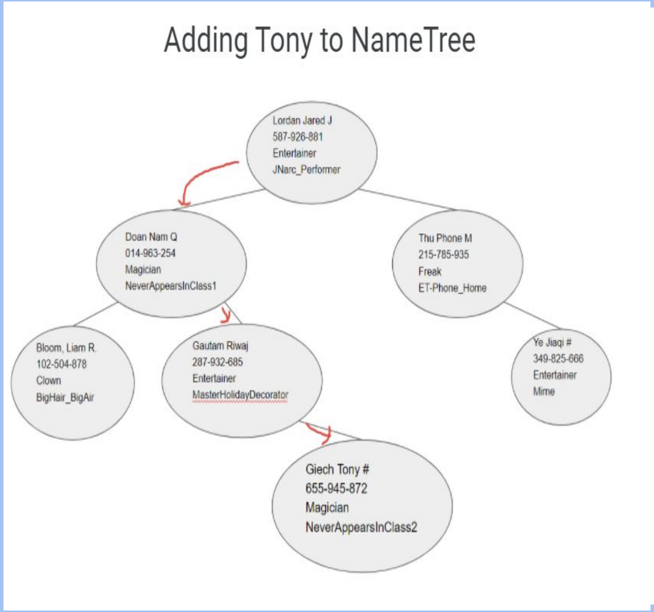
- Find the key that matches the category of the employee
- Then we add the employee to the value of that key

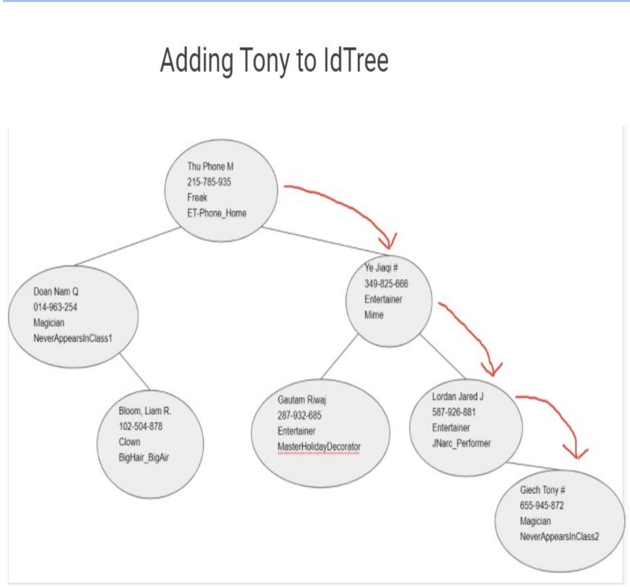
T(insert) = T (add to name tree) + T(add to Id tree) + T (find key) + T(add employee)

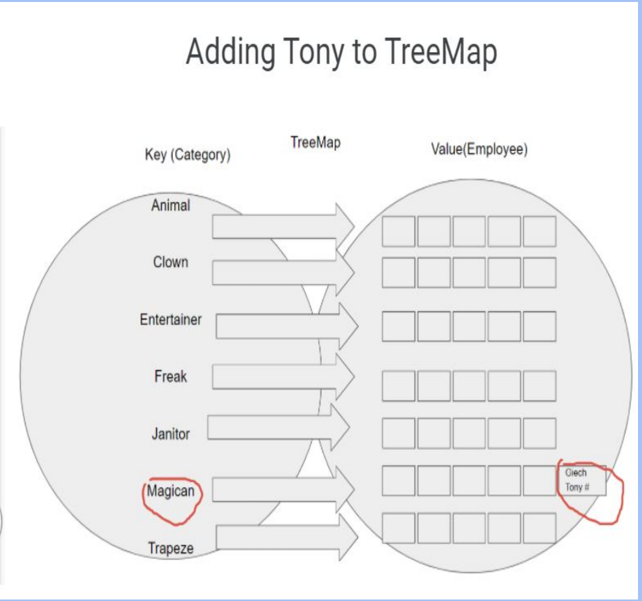
$$= O(\log(n)) + O(\log(n)) + O(\log(n)) + O(\log(n))$$

$$= O(\log(n))$$

3) Insert a new Employee


Adding Tony to NameTree
Lordan Jared J 587-926-881 Entertainer JNarc_Performer
Doan Nam Q 014-963-254 Magician NeverAppearsInClass1
Thu Phone M 215-785-935 Freak ET-Phone_Home
Bloom, Liam R. 102-504-878 Clown BigHair_BigAir
Gautam Riwaj 287-932-685 Entertainer MasterHolidayDecorator
Ye Jiaqi # 349-825-666 Entertainer Mime
Giech Tony # 655-945-872 Magician NeverAppearsInClass2


Adding Tony to IdTree
Thu Phone M 215-785-935 Freak ET-Phone_Home
Doan Nam Q 014-963-254 Magician NeverAppearsInClass1
Ye Jiaqi # 349-825-666 Entertainer Mime
Bloom, Liam R. 102-504-878 Clown BigHair_BigAir
Gautam Riwaj 287-932-685 Entertainer MasterHolidayDecorator
Lordan Jared J 587-926-881 Entertainer JNarc_Performer
Giech Tony # 655-945-872 Magician NeverAppearsInClass2



4) Delete an Employee

1. delete Employee from the name tree and Id tree

2. Delete Employee from the TreeMap

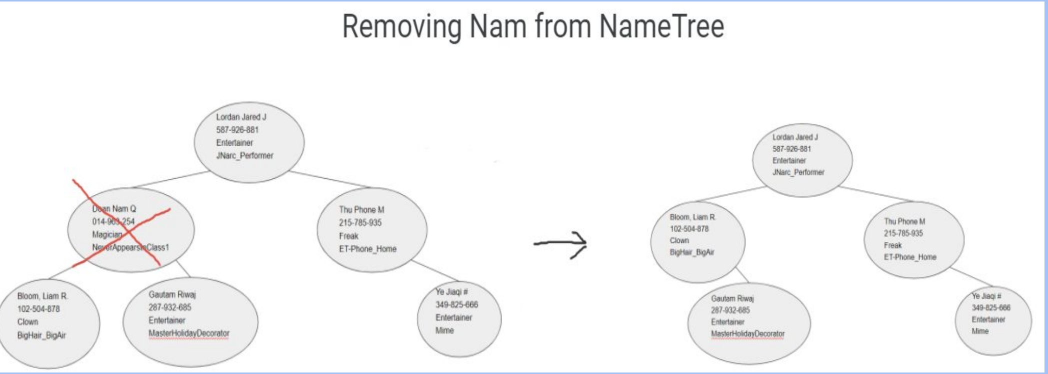
- Find the key that matches the category of the employee
- Then we remove the employee from the value of that key using Iterator

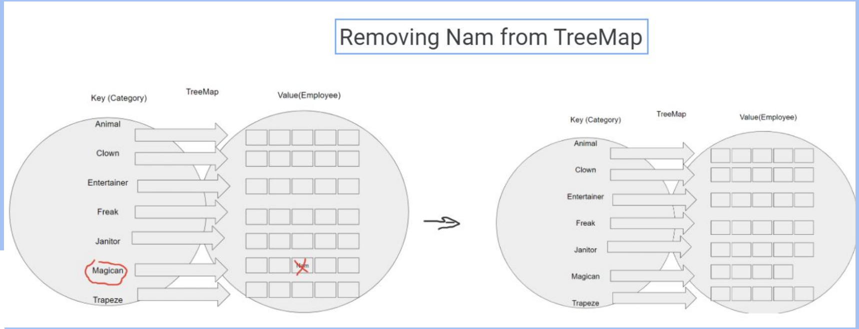
T(delete) = T (delete from name tree) + T(delete from Id tree) + T (find key) + T(delete employee)

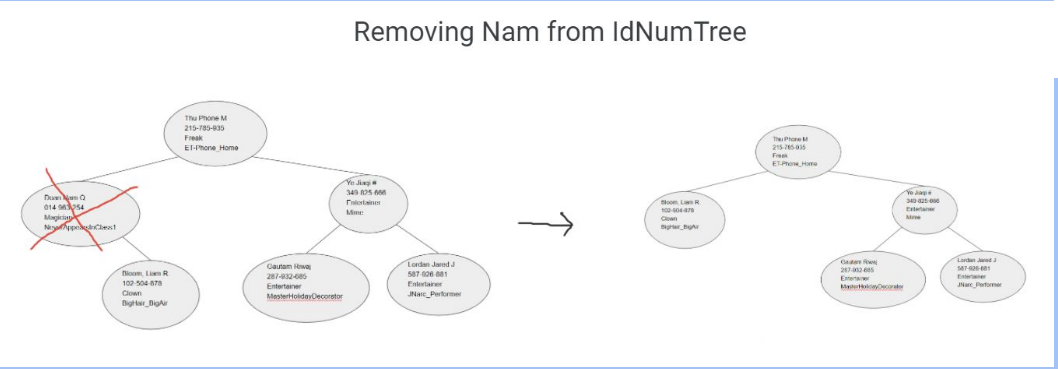
$$= O(\log(n)) + O(\log(n)) + O(\log(n)) + O(\log(n))$$

$$= O(\log(n))$$

4) Delete an Employee

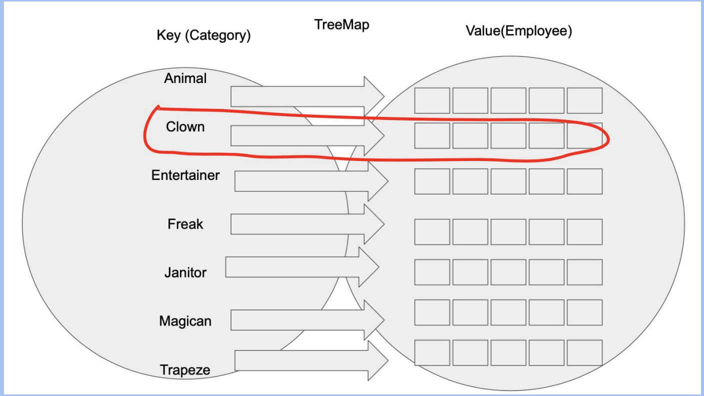






5) Print a Particular Category Alphabetically

1. Find the key with that particular category
   - map.get(category)
2. Get the TreeSet
3. Print the value of the TreeSet using Iterator
   - Iterator value_itr = set.iterator();



Category:
Clown

T(print a particular category) = T(find the key) + T (value of the TreeSet)

= O (log(n)) + O (n)

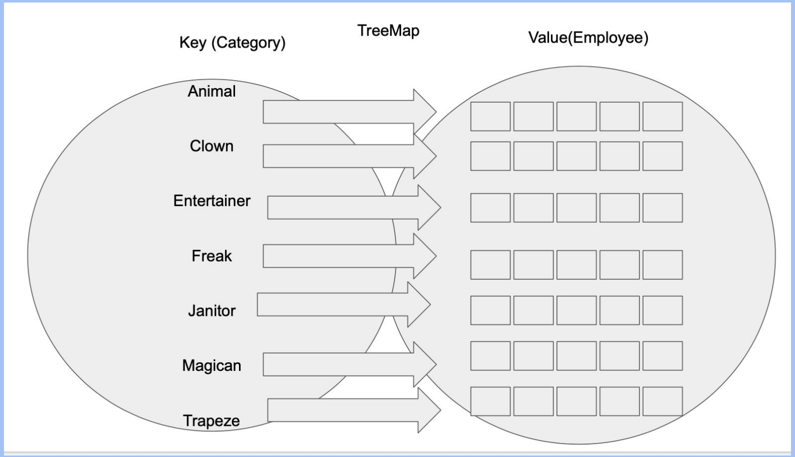
= O(n)

6) Print Employees and Categories Alphabetically

1) Examine each key of the TreeMap, using one iterator
   - Iterator key_itr = TreeMap.iterator();
2) Print the current key's value, TreeSet, using another Iterator
   - Iterator value_itr = TreeSet.iterator();

T(print all categories) = # of categories x T (print values)

= O (n)

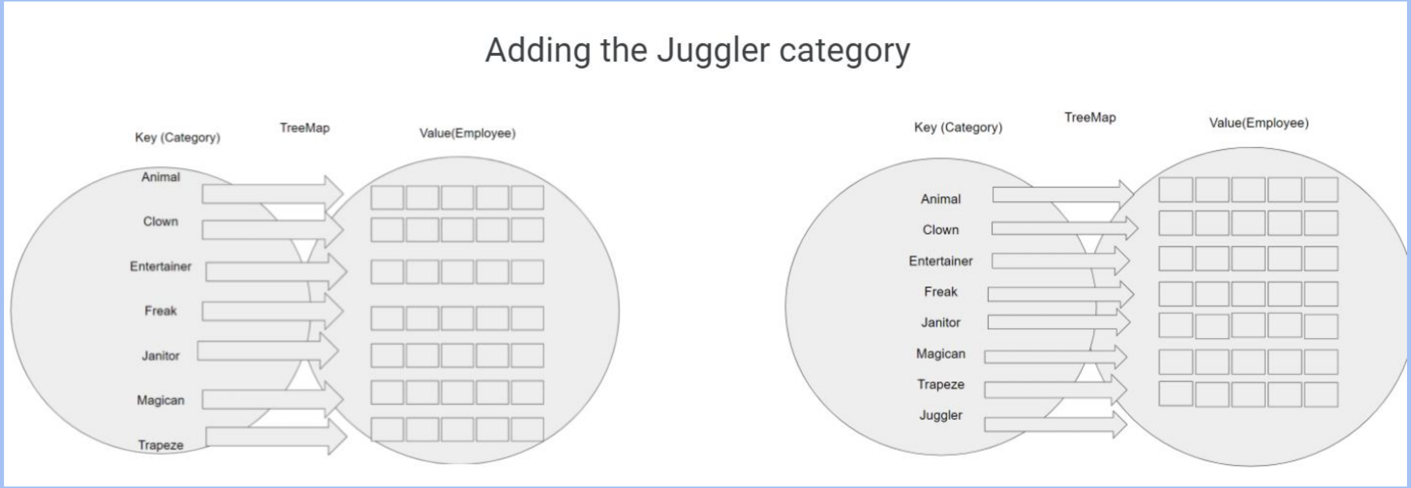


7) Add a Category

1) Add a new key to the TreeMap
- map.put(category, new TreeSet());

T(add a category) = T(add new key)

= O(log(n))



8) Delete a Category

1) Delete the key of the category

- map.remove(key)
- Since we remove the key, we automatically delete all the corresponding value

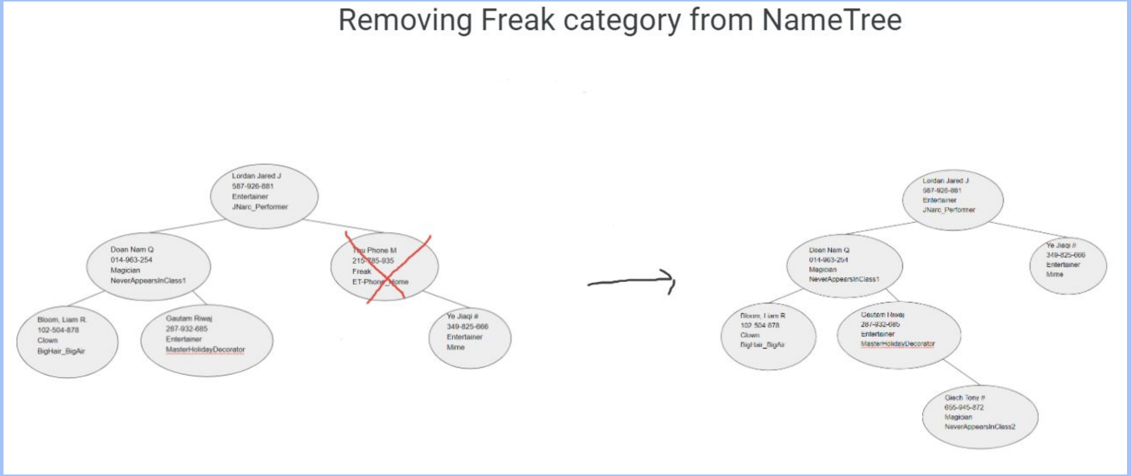
2) Delete from the tree

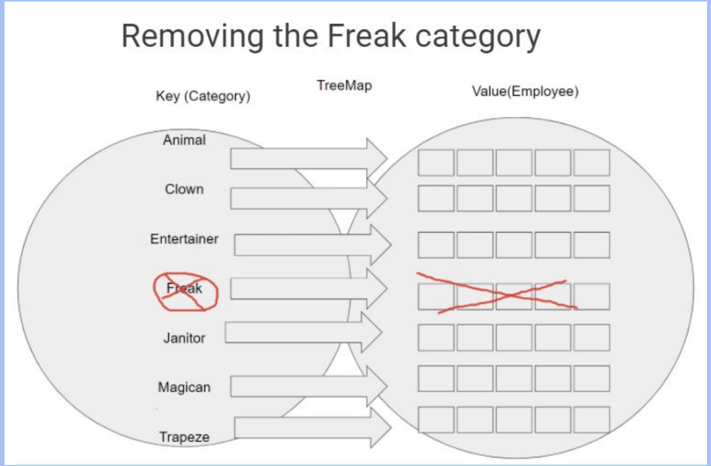
T(delete a category) = T(remove key) + T (remove nodes from nameTree) + T (remove nodes from idTree)

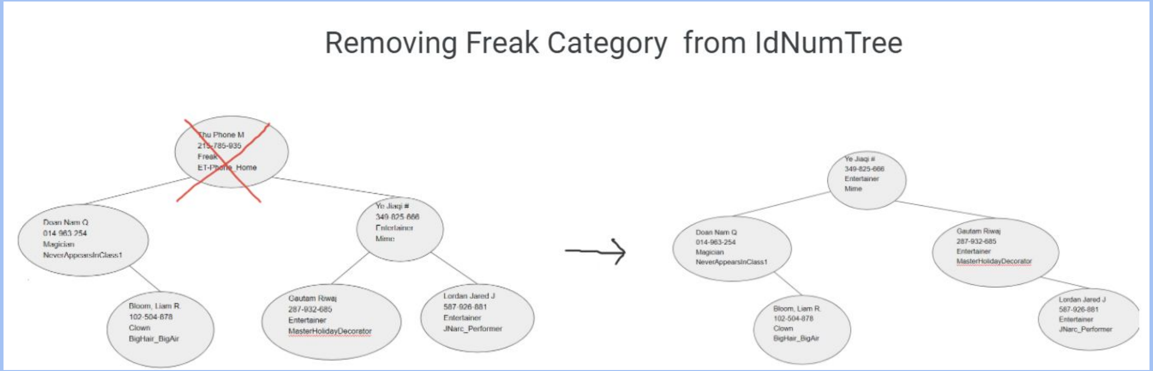
= O(log n) + O(nlog(n)) + O(nlog(n))

= O(nlog(n))

8) Delete a Category


Removing Freak category from NameTree
Lordan Jared J 587-926-881 Entertainer JNarc_Performer
Doan Nam Q 014-963-254 Magician NeverAppearsInClass1
Thu Phone M 215-785-935 Freak ET-Phone_Home
Bloom, Liam R. 102-504-878 Clown BigHair_BigAir
Gautam Riwaj 287-932-685 Entertainer MasterHolidayDecorator
Ye Jiaqi # 349-825-666 Entertainer Mime
Giech Tony # 655-945-872 Magician NeverAppearsInClass2




Removing Freak Category from IdNumTree
Thu Phone M 215-785-935 Freak ET-Phone_Home
Doan Nam Q 014-963-254 Magician NeverAppearsInClass1
Ye Jiaqi # 349-825-666 Entertainer Mime
Bloom, Liam R. 102-504-878 Clown BigHair_BigAir
Gautam Riwaj 287-932-685 Entertainer MasterHolidayDecorator
Lordan Jared J 587-926-881 Entertainer JNarc_Performer

9) Quit

- JOptionPane.showMessageDialog(null,"Good Bye!");
  System.exit(0);
- Clear the two trees and the treemap
- Close the file

Thank you!