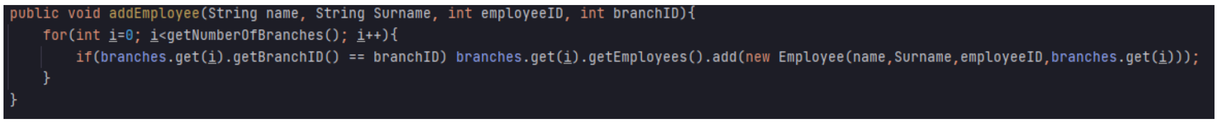
# GTU Department of Computer Engineering CSE 222/505 - Spring 2021

## Homework 3 Method Analyzes

BAHADIR ETKA
KILINÇ 1901042701

1. Adding an employee



```java
public void addEmployee(String name, String Surname, int employeeID, int branchID){
    for(int i=0; i<getNumberOfBranches(); i++){
        if(branches.get(i).getBranchID() == branchID) branches.get(i).getEmployees().add(new Employee(name,Surname,employeeID,branches.get(i)));
    }
}
```

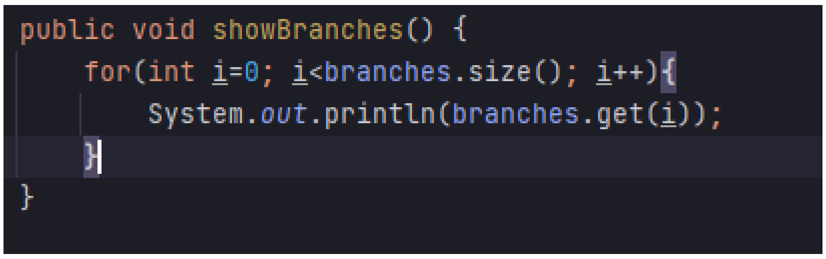
Branches are kept on Linked List so branches.get(i) takes O(N).

Employees are kept on ArrayList and its add method takes amortized constant time O(1)

For loop takes O(N) time

Multiplying O(N) and O(N) The total running time is O(N$^2$)

2. List all Branches



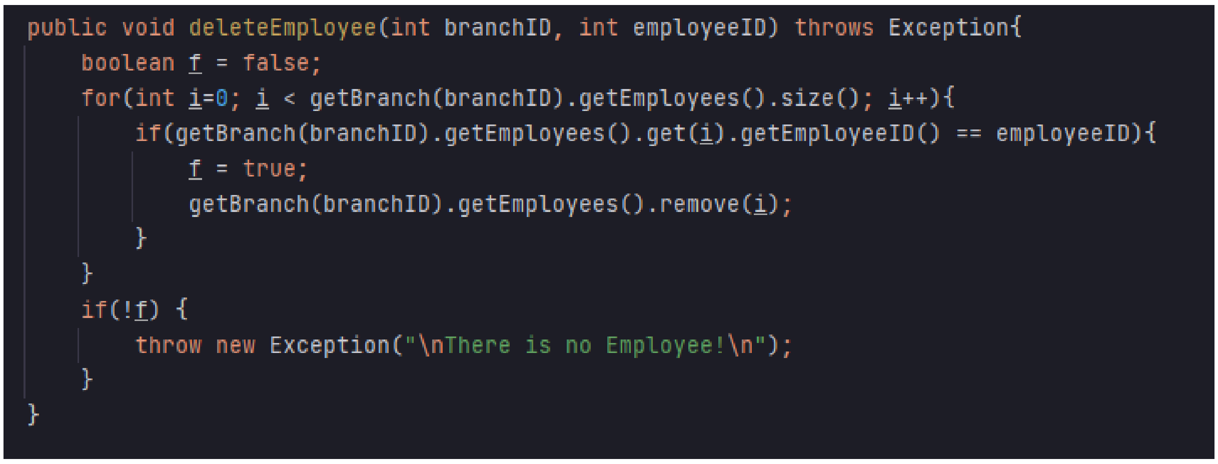
```java
public void showBranches() {
    for(int i=0; i<branches.size(); i++){
        System.out.println(branches.get(i));
    }
}
```

Branches are kept on Linked List so branches.get(i) takes O(N).

For loop takes Θ(N)

If we multiply O(N) and Θ(N)  The total running time is O(N$^2$)

3. Deleting an employee from a branch



```java
public void deleteEmployee(int branchID, int employeeID) throws Exception{
    boolean f = false;
    for(int i=0; i < getBranch(branchID).getEmployees().size(); i++){
        if(getBranch(branchID).getEmployees().get(i).getEmployeeID() == employeeID){
            f = true;
            getBranch(branchID).getEmployees().remove(i);
        }
    }
    if(!f) {
        throw new Exception("\nThere is no Employee!\n");
    }
}
```

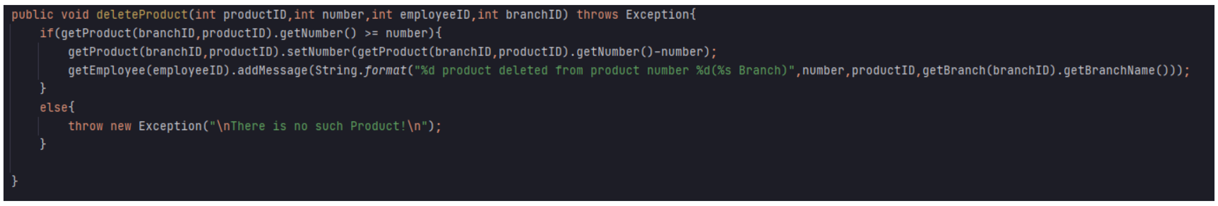
Branches are kept on Linked List so branches.get(i) takes O(N).

Employees are kept on ArrayList and its get method takes constant time O(1).

Employees are kept on ArrayList and its remove method takes constant time O(N).

For loops takes Θ (N) If we Multiply O(N) and Θ (N) The total running time is O($N^2$)

4. Deleting a product from a branch



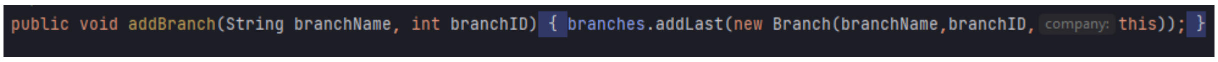
```java
public void deleteProduct(int productID,int number,int employeeID,int branchID) throws Exception{
    if(getProduct(branchID,productID).getNumber() >= number){
        getProduct(branchID,productID).setNumber(getProduct(branchID,productID).getNumber()-number);
        getEmployee(employeeID).addMessage(String.format("%d product deleted from product number %d(%s Branch)",number,productID,getBranch(branchID).getBranchName()));
    }
    else{
        throw new Exception("\nThere is no such Product!\n");
    }

}
```

Products are kept in Hybrid List and its at method takes O(N)

Branches are kept on Linked List so branches.get(i) takes O(N)

If we add O(N) and O(N) The total running time O($N^2$)

5. Adding a branch to company

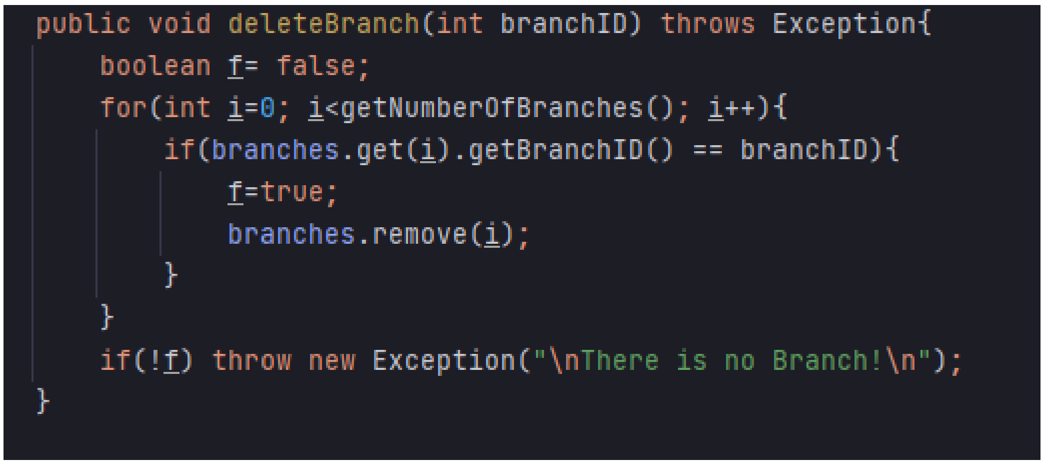


```java
public void addBranch(String branchName, int branchID) { branches.addLast(new Branch(branchName,branchID, company: this)); }
```

Branches are kept on Linked List and its addLast method takes O(1)

So total running time is O(1)

6. Deleting a branch from company



```java
public void deleteBranch(int branchID) throws Exception{
    boolean f= false;
    for(int i=0; i<getNumberOfBranches(); i++){
        if(branches.get(i).getBranchID() == branchID){
            f=true;
            branches.remove(i);
        }
    }
    if(!f) throw new Exception("\nThere is no Branch!\n");
}
```

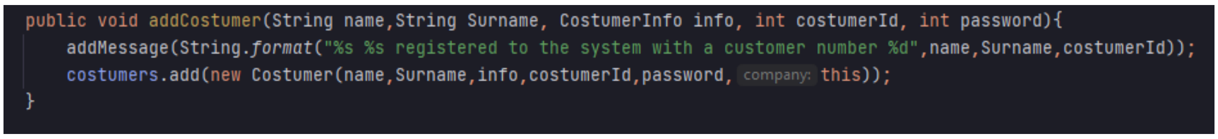
Branches are kept on Linked List so branches.get(i) takes O(N)

Branches are kept on Linked List and its remove method takes O(1)

Total of O(N) and O(1) is O(N)

For loops takes Θ (N) And If we multiply Θ (N) and O(N) the total running time will be O($N^2$)

7. Adding a costumer to system(company)

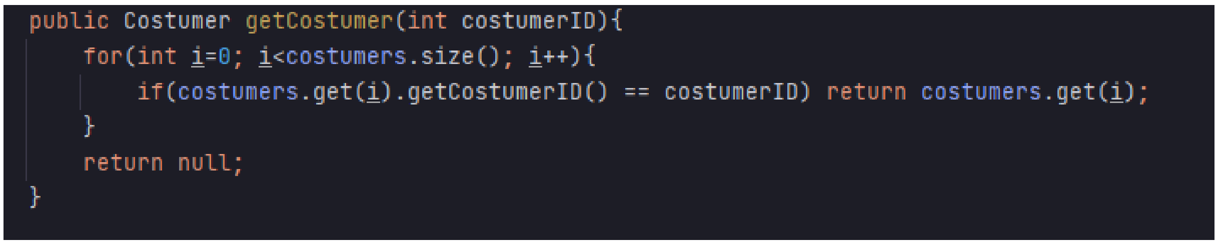


```java
public void addCostumer(String name,String Surname, CostumerInfo info, int costumerId, int password){
    addMessage(String.format("%s %s registered to the system with a customer number %d",name,Surname,costumerId));
    costumers.add(new Costumer(name,Surname,info,costumerId,password, company: this));
}
```

Costumers are kept in Array List and its add method takes O(1)

So the total running time is O(1)

8. Getting a costumer from company

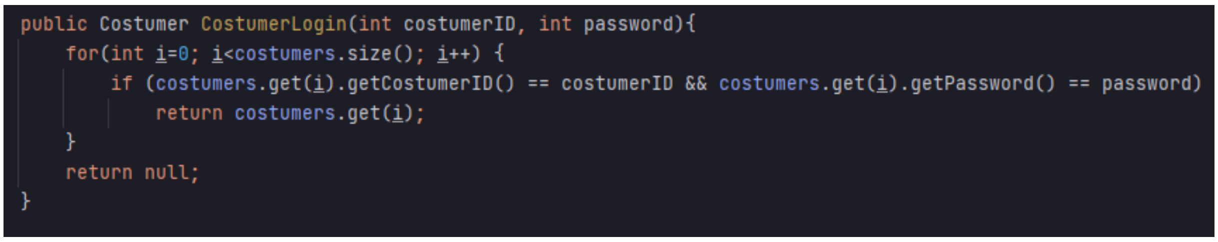


```java
public Costumer getCostumer(int costumerID){
    for(int i=0; i<costumers.size(); i++){
        if(costumers.get(i).getCostumerID() == costumerID) return costumers.get(i);
    }
    return null;
}
```

Costumers are kept in Array List and its get method takes O(1)

So the total running time is O(1)

9. Costumer Login

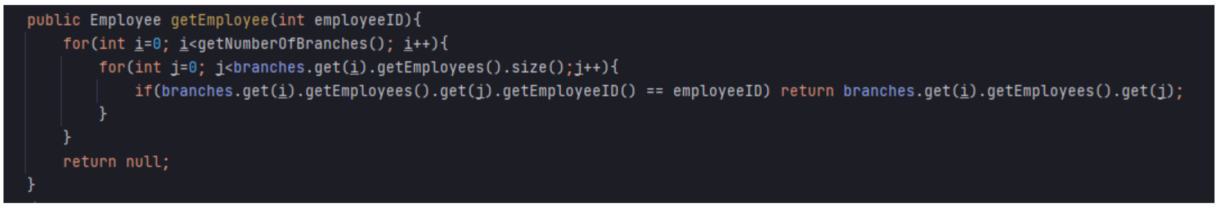


```java
public Costumer CostumerLogin(int costumerID, int password){
    for(int i=0; i<costumers.size(); i++) {
        if (costumers.get(i).getCostumerID() == costumerID && costumers.get(i).getPassword() == password)
            return costumers.get(i);
    }
    return null;
}
```

Costumers are kept in Array List and its get method takes O(1)

So the total running time is O(1)

10. Getting an employee



```java
public Employee getEmployee(int employeeID){
    for(int i=0; i<getNumberOfBranches(); i++){
        for(int j=0; j<branches.get(i).getEmployees().size();j++){
            if(branches.get(i).getEmployees().get(j).getEmployeeID() == employeeID) return branches.get(i).getEmployees().get(j);
        }
    }
    return null;
}
```

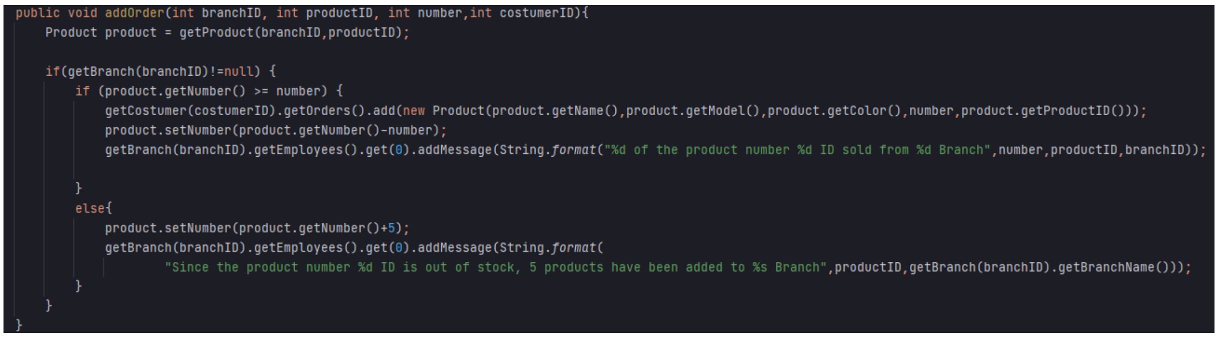
Employees are kept in Array List and its get method takes O(1)

Branches are kept in Linked List and its get method takes O(N)

For loop takes O(N)

If we multiply O(N) and O(N) the total running time will be O($N^2$)

11. Adding an order to a costumer



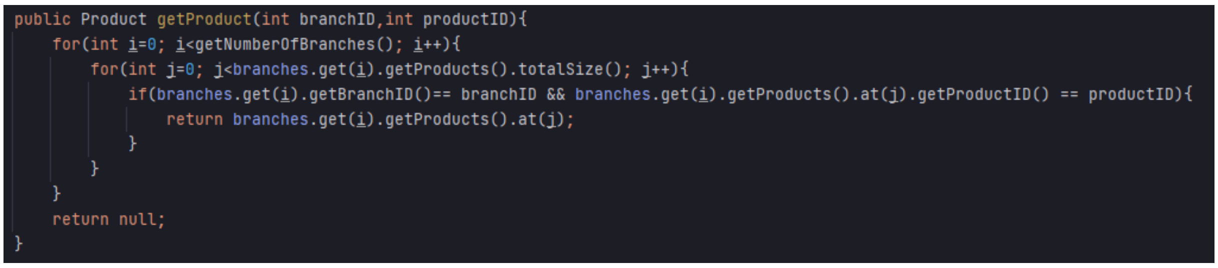
```java
public void addOrder(int branchID, int productID, int number,int costumerID){
    Product product = getProduct(branchID,productID);

    if(getBranch(branchID)!=null) {
        if (product.getNumber() >= number) {
            getCostumer(costumerID).getOrders().add(new Product(product.getName(),product.getModel(),product.getColor(),number,product.getProductID()));
            product.setNumber(product.getNumber()-number);
            getBranch(branchID).getEmployees().get(0).addMessage(String.format("%d of the product number %d ID sold from %d Branch",number,productID,branchID));

        }
        else{
            product.setNumber(product.getNumber()+5);
            getBranch(branchID).getEmployees().get(0).addMessage(String.format(
                    "Since the product number %d ID is out of stock, 5 products have been added to %s Branch",productID,getBranch(branchID).getBranchName()));
        }
    }
}
```

Products are kept in Hybrid List and its at method takes O(n)

Products are kept in Hybrid List and its add method takes O(1)

So total running time is O(N)

12. Get product from the branch



```
public Product getProduct(int branchID,int productID){
    for(int i=0; i<getNumberOfBranches(); i++){
        for(int j=0; j<branches.get(i).getProducts().totalSize(); j++){
            if(branches.get(i).getBranchID()== branchID && branches.get(i).getProducts().at(j).getProductID() == productID){
                return branches.get(i).getProducts().at(j);
            }
        }
    }
    return null;
}
```

Branches are kept in Linked List and its get method takes O(N)

Products are kept in Hybrid List and its at method takes O(N)

If we add these O(N) and O(N) is O(N)

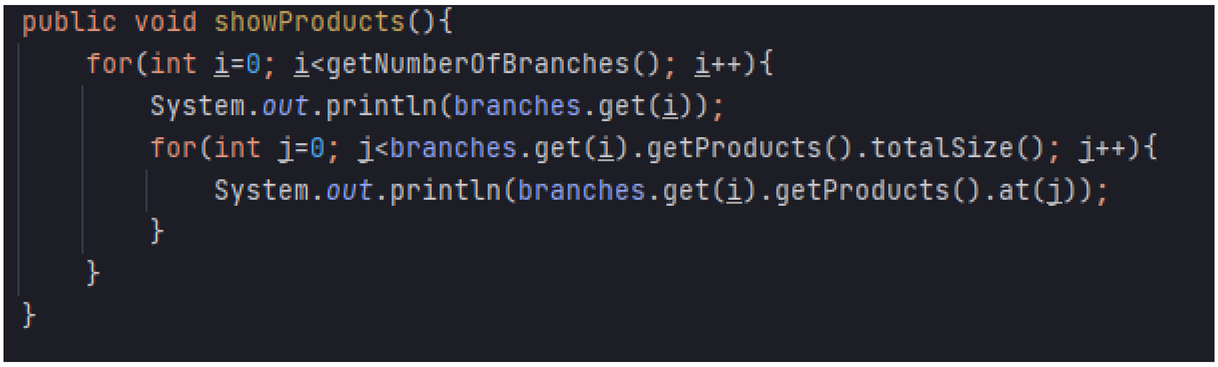
Inner for loop takes O(N) time

If we multiply O(N) and O(N) the total is O($N^2$)

Outer for loop takes O(N)

If we multiply  O(N) and O($N^2$) the total running time is O($N^3$)

13. Lists product in all branches



```
public void showProducts(){
    for(int i=0; i<getNumberOfBranches(); i++){
        System.out.println(branches.get(i));
        for(int j=0; j<branches.get(i).getProducts().totalSize(); j++){
            System.out.println(branches.get(i).getProducts().at(j));
        }
    }
}
```

Products are kept in Hybrid List and its at method takes O(N)

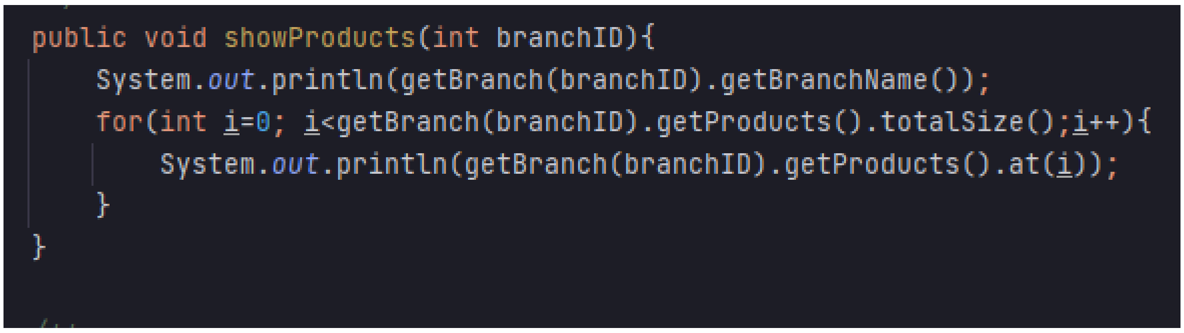
Inner for loop takes O(N)

If we multiply these O(N) and O(N) the result is O($N^2$)

Outer for loop takes O(N)

If we multiply these O(N) and O($N^2$) the total running time is O($N^3$)

14. Lists all product in a branch



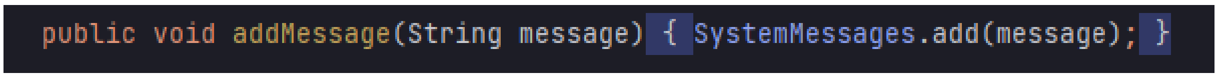
```java
public void showProducts(int branchID){
    System.out.println(getBranch(branchID).getBranchName());
    for(int i=0; i<getBranch(branchID).getProducts().totalSize();i++){
        System.out.println(getBranch(branchID).getProducts().at(i));
    }
}
```

Products are kept in Hybrid List and its at method takes O(N)

For loop takes O(N)

If we multiply these O(N) and O(N) the total running time will be O(N$^2$)

15. Adding a message to system

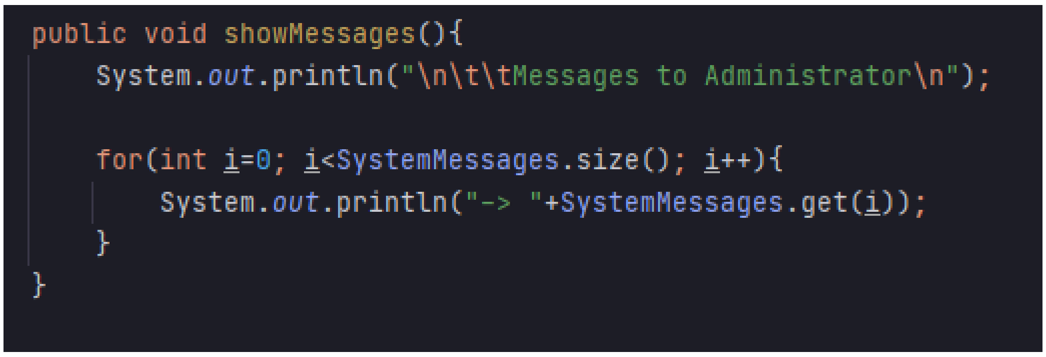


```java
public void addMessage(String message) { SystemMessages.add(message); }
```

SystemMessages are kept in Array List and its add Method takes O(1)

So the total running time is O(1)

16. Listing all messages from the system



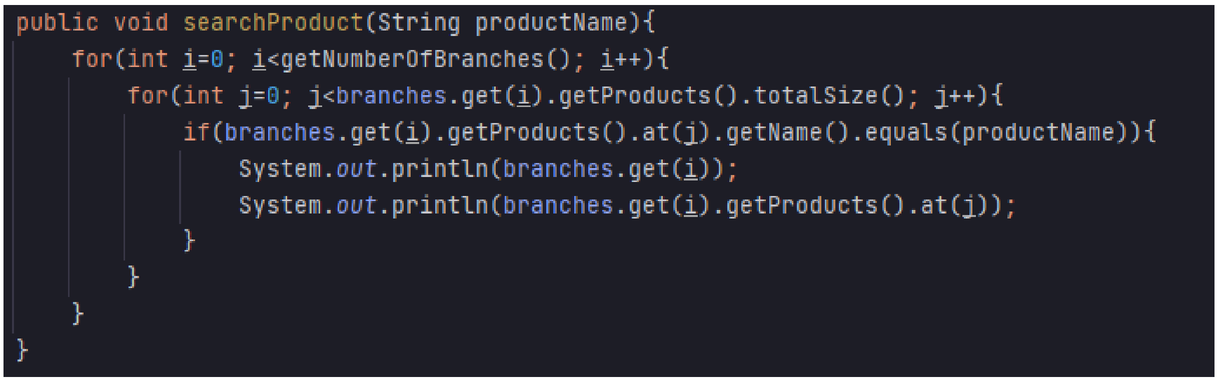
```java
public void showMessages(){
    System.out.println("\n\t\tMessages to Administrator\n");

    for(int i=0; i<SystemMessages.size(); i++){
        System.out.println("-> "+SystemMessages.get(i));
    }
}
```

SystemMessages are kept in Array List and its get method takes O(1)

For loop takes O(N)

If we multiply these O(N) and O(1) the total running time will be O(N)

17. Searching a product on all branches



```java
public void searchProduct(String productName){
    for(int i=0; i<getNumberOfBranches(); i++){
        for(int j=0; j<branches.get(i).getProducts().totalSize(); j++){
            if(branches.get(i).getProducts().at(j).getName().equals(productName)){
                System.out.println(branches.get(i));
                System.out.println(branches.get(i).getProducts().at(j));
            }
        }
    }
}
```

Products are kept in Hybrid List and its at method takes O(N)

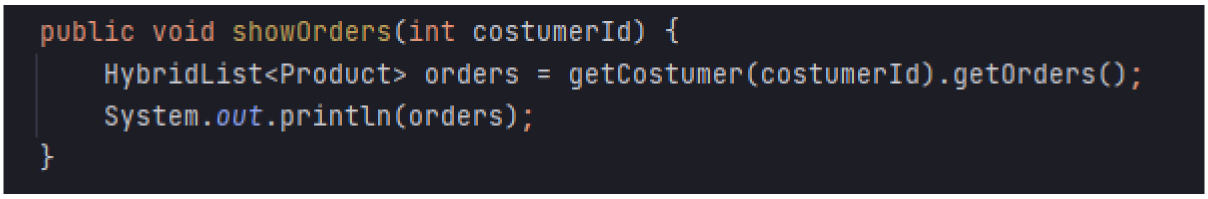
Inner for loop takes O(N)

If we multiply O(N) and O(N) the result will be O(N$^2$)

Outer for loop takes O(N)

If we multiply these O(N) and O(N$^2$) the total running time will be O(N$^3$)

18. Listing orders of a costumer.

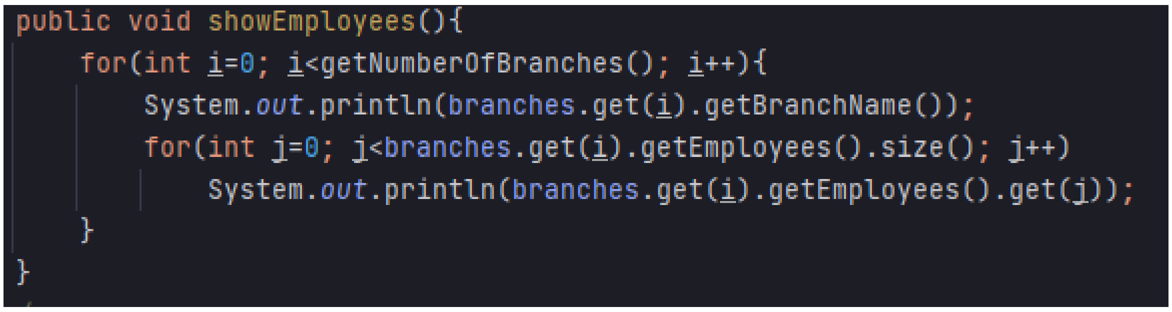


```java
public void showOrders(int costumerId) {
    HybridList<Product> orders = getCostumer(costumerId).getOrders();
    System.out.println(orders);
}
```

Orders are kept in Hybrid List and its toString method takes O(N)

So the total running time is O(N)

19. Listing employees in company



```
public void showEmployees(){
    for(int i=0; i<getNumberOfBranches(); i++){
        System.out.println(branches.get(i).getBranchName());
        for(int j=0; j<branches.get(i).getEmployees().size(); j++)
            System.out.println(branches.get(i).getEmployees().get(j));
    }
}
```

Employees are kept in Array List so its get method takes O(1)

Branches are kept in Linked List and its get method takes O(N)

If we add these, the result is O(N)

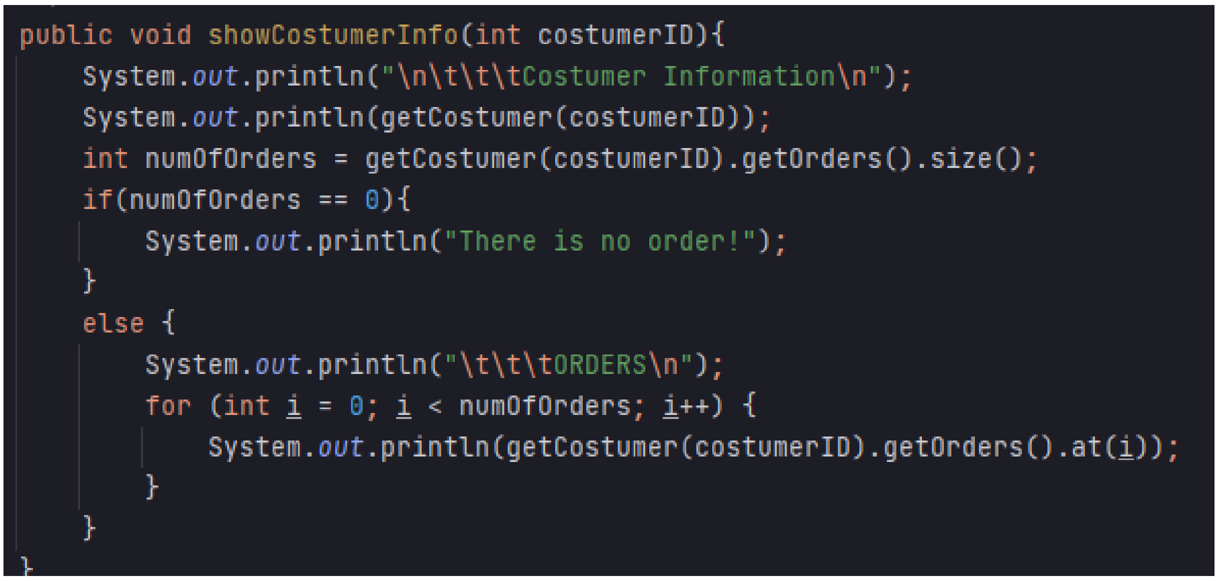
Inner for loop takes O(N)

If we multiply these O(N) and O(N) the result is O($N^2$)

Outer for loop takes O(N)

If we multipy O(N) and O($N^2$) the total running time will be O($N^3$)

20. Listing costumers information



```
public void showCostumerInfo(int costumerID){
    System.out.println("\n\t\t\tCostumer Information\n");
    System.out.println(getCostumer(costumerID));
    int numOfOrders = getCostumer(costumerID).getOrders().size();
    if(numOfOrders == 0){
        System.out.println("There is no order!");
    }
    else {
        System.out.println("\t\t\tORDERS\n");
        for (int i = 0; i < numOfOrders; i++) {
            System.out.println(getCostumer(costumerID).getOrders().at(i));
        }
    }
}
```

Costumers are kept in ArrayList and its get method takes O(1) time.

Products are kept in Hybrid List and its at method takes O(N)

For loop takes O(N)

If we multiply these O(N) and O(N) the result is O($N^2$)

And If we add O(1) and O($N^2$) the total running time will be O($N^2$)