GIT Department of Computer Engineering
CSE 222/505 - Spring 2021
Homework # Report

Mehmet Acar
1801042095

1-SYSTEM REQUIREMENTS

Functional Requirements

1-The system must send a error message to user when user logins with wrong e-mail or password.

2-The system must prevent the customer from purchasing products when there is no branch employee on the system.

3-The system must send a warning message when user enters wrong product model number,color number etc…

4- The system must give an error message if the admin wants to remove a branch or branch employee while there is no branch or branch employee on the system.

Non-functional Requirements

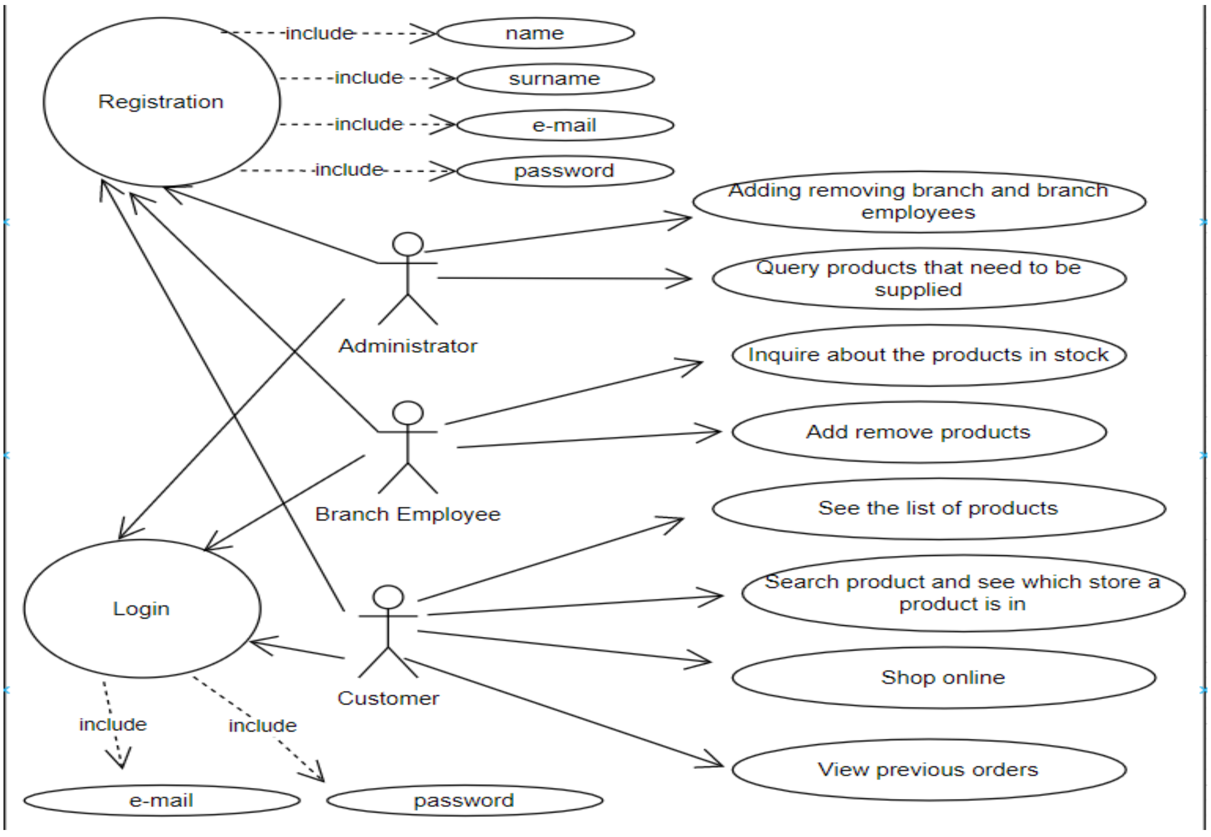
1-Usability

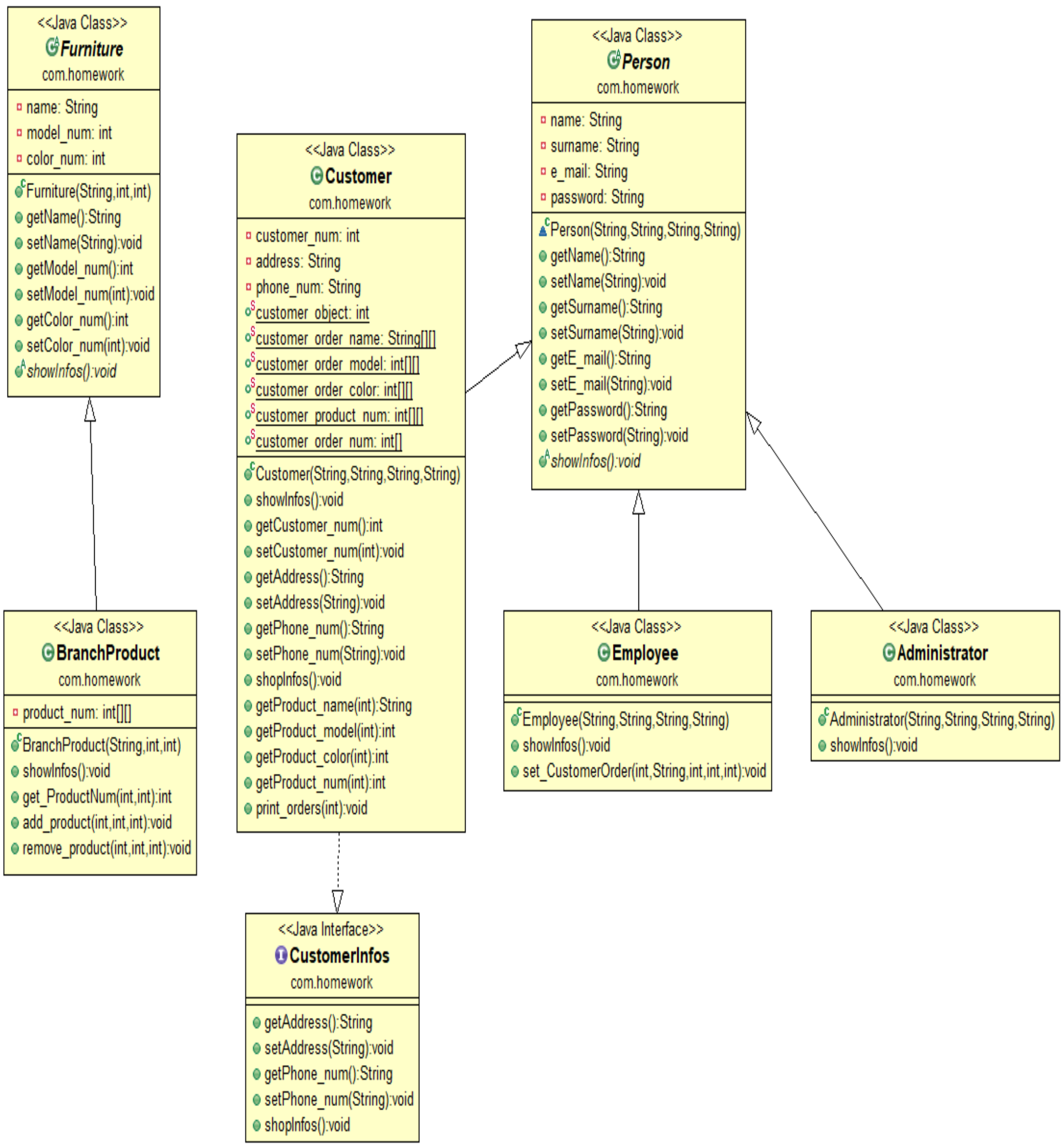
2-Performance

3-Security

4-Reusability

2-USE CASE AND CLASS DIAGRAMS




<<Java Class>>
Furniture
com.homework
name: String
model_num: int
color_num: int
Furniture(String,int,int)
getName():String
setName(String):void
getModel_num():int
setModel_num(int):void
getColor_num():int
setColor_num(int):void
showInfos():void
<<Java Class>>
Customer
com.homework
customer_num: int
address: String
phone_num: String
customer_object: int
customer_order_name: String[][]
customer_order_model: int[][]
customer_order_color: int[][]
customer_product_num: int[][]
customer_order_num: int[]
Customer(String,String,String,String)
showInfos():void
getCustomer_num():int
setCustomer_num(int):void
getAddress():String
setAddress(String):void
getPhone_num():String
setPhone_num(String):void
shopInfos():void
getProduct_name(int):String
getProduct_model(int):int
getProduct_color(int):int
getProduct_num(int):int
print_orders(int):void
<<Java Class>>
Person
com.homework
name: String
surname: String
e_mail: String
password: String
Person(String,String,String,String)
getName():String
setName(String):void
getSurname():String
setSurname(String):void
getE_mail():String
setE_mail(String):void
getPassword():String
setPassword(String):void
showInfos():void
<<Java Class>>
BranchProduct
com.homework
product_num: int[][]
BranchProduct(String,int,int)
showInfos():void
get_ProductNum(int,int):int
add_product(int,int,int):void
remove_product(int,int,int):void
<<Java Class>>
Employee
com.homework
Employee(String,String,String,String)
showInfos():void
set_CustomerOrder(int,String,int,int,int):void
<<Java Class>>
Administrator
com.homework
Administrator(String,String,String,String)
showInfos():void
<<Java Interface>>
CustomerInfos
com.homework
getAddress():String
setAddress(String):void
getPhone_num():String
setPhone_num(String):void
shopInfos():void

3-PROBLEM SOLUTION APPROACH

1-Identify The Problem

Problem is making automation system. There will be administrators, branch employees and customers in the automation system.

2-Gather Information

Administrators manage the branch and branch and employees. Branch employee manage the products,update the customers' previous orders. Customers see list of products, see which store a product is in and shop online.

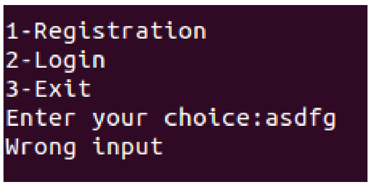
3-Iterate Potential Solutions

Administrators adding and removing branches and branch employees.
Branch employees adding and removing products.
Customers enter address and phone number information when he/she does shopping.

4-TEST CASES

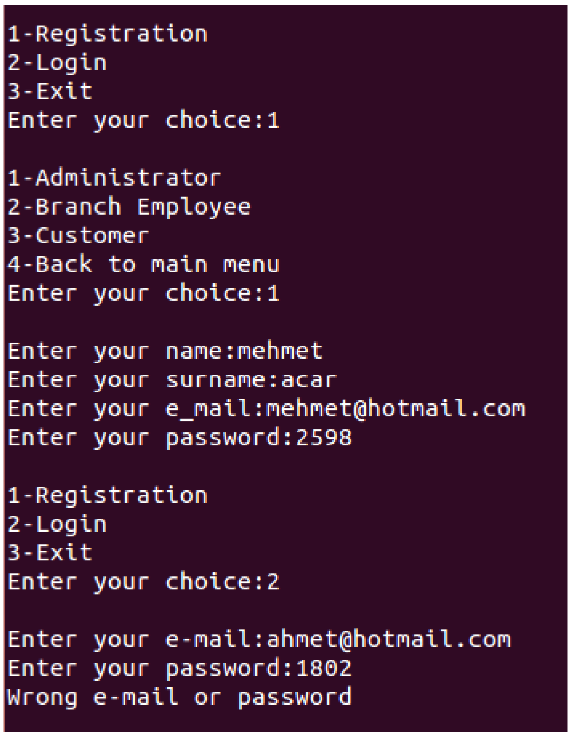
Initially, each number of products is 100

1-If user enters wrong input



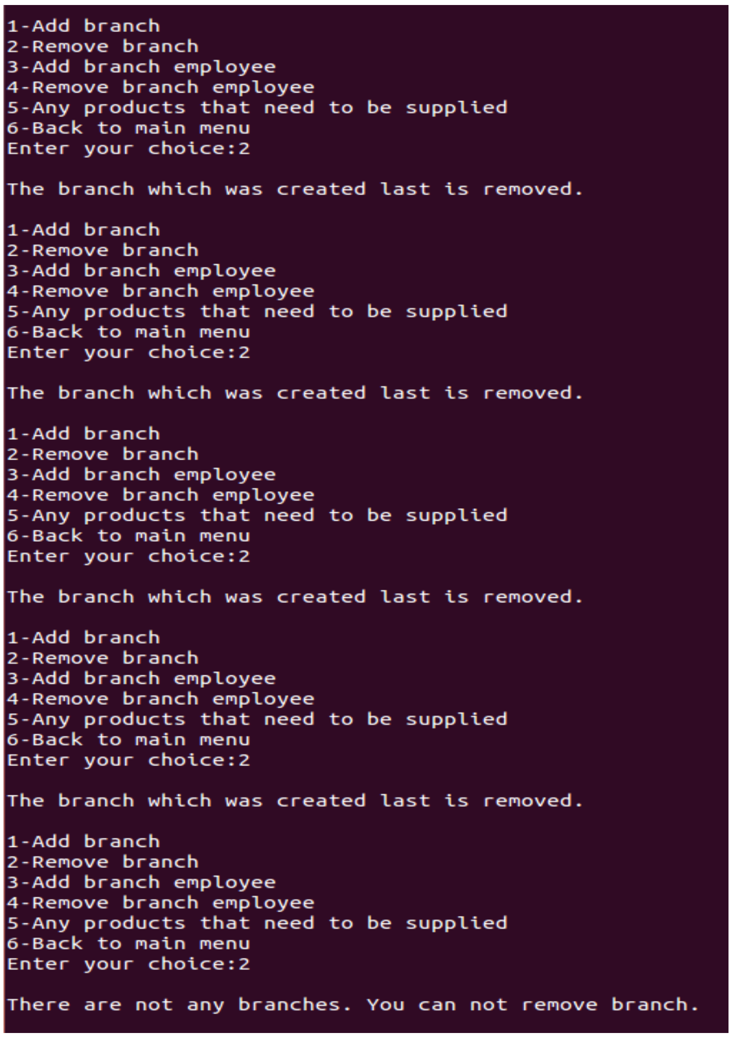
```
1-Registration
2-Login
3-Exit
Enter your choice:asdfg
Wrong input
```

2-Login Error



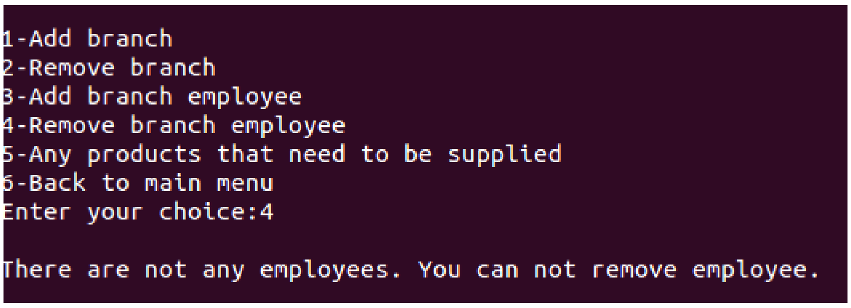
```
1-Registration
2-Login
3-Exit
Enter your choice:1

1-Administrator
2-Branch Employee
3-Customer
4-Back to main menu
Enter your choice:1

Enter your name:mehmet
Enter your surname:acar
Enter your e_mail:mehmet@hotmail.com
Enter your password:2598

1-Registration
2-Login
3-Exit
Enter your choice:2

Enter your e-mail:ahmet@hotmail.com
Enter your password:1802
Wrong e-mail or password
```

3- When all branches was removed



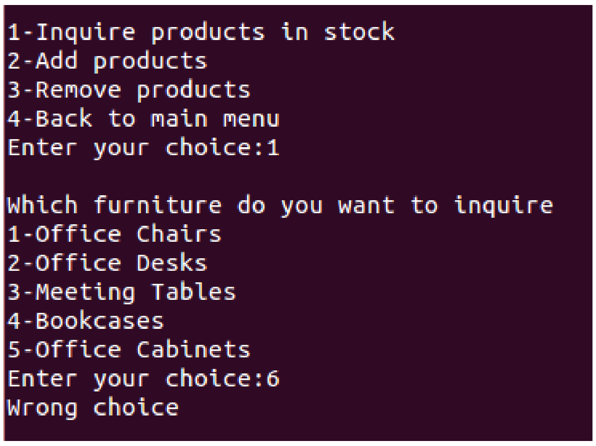
```
1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:2

The branch which was created last is removed.

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:2

The branch which was created last is removed.

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:2

The branch which was created last is removed.

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:2

The branch which was created last is removed.

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:2

There are not any branches. You can not remove branch.
```

4-When there are not any employees in the system



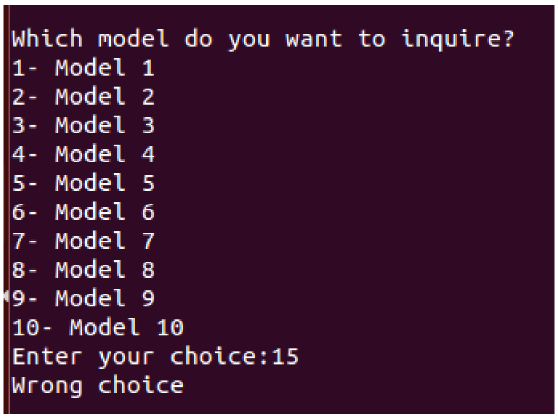
```
1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:4

There are not any employees. You can not remove employee.
```

5-When branch employee enters wrong furniture selection number



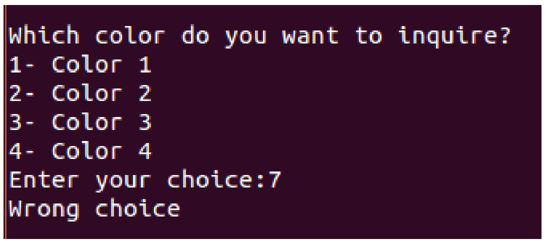
```
1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:1

Which furniture do you want to inquire
1-Office Chairs
2-Office Desks
3-Meeting Tables
4-Bookcases
5-Office Cabinets
Enter your choice:6
Wrong choice
```

6-When branch employee enters wrong model number



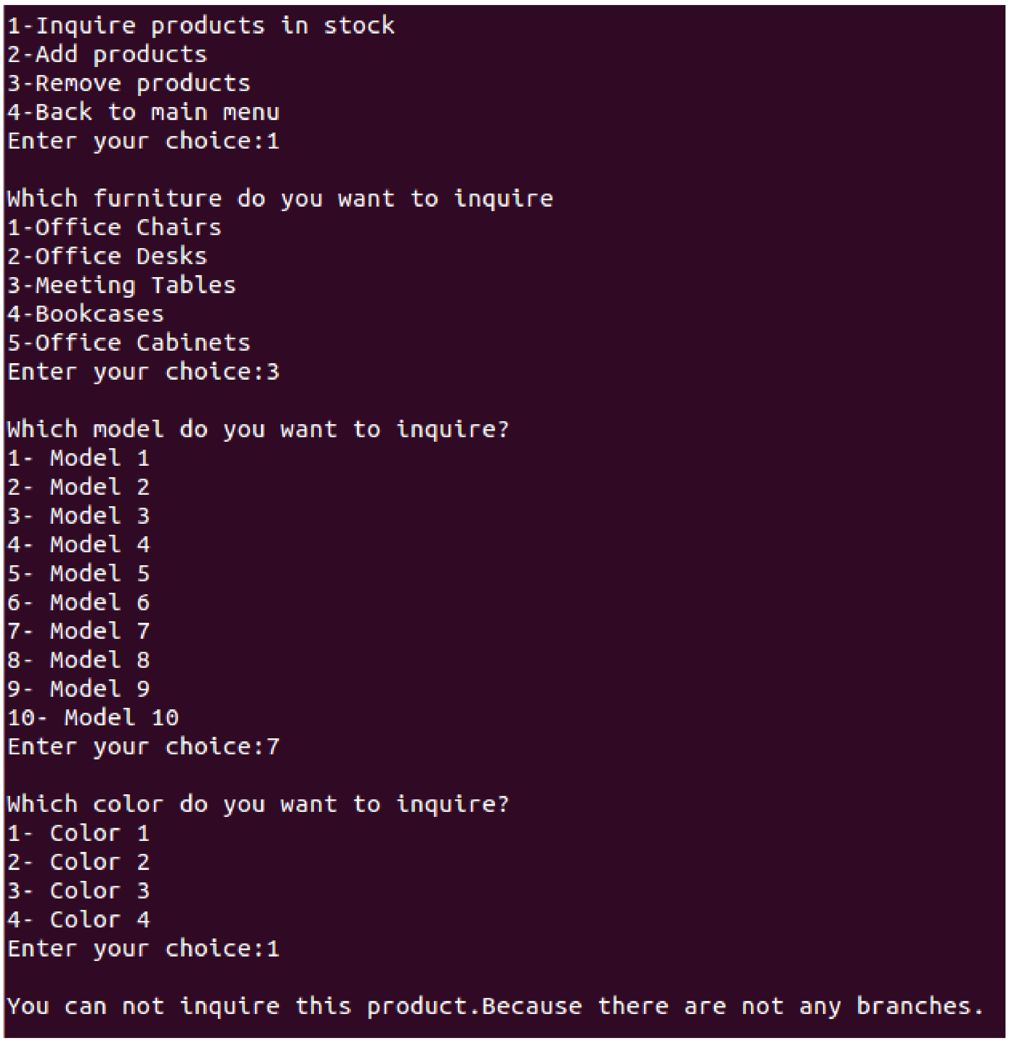
```
Which model do you want to inquire?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
Enter your choice:15
Wrong choice
```

7-When branch employee enters wrong color number



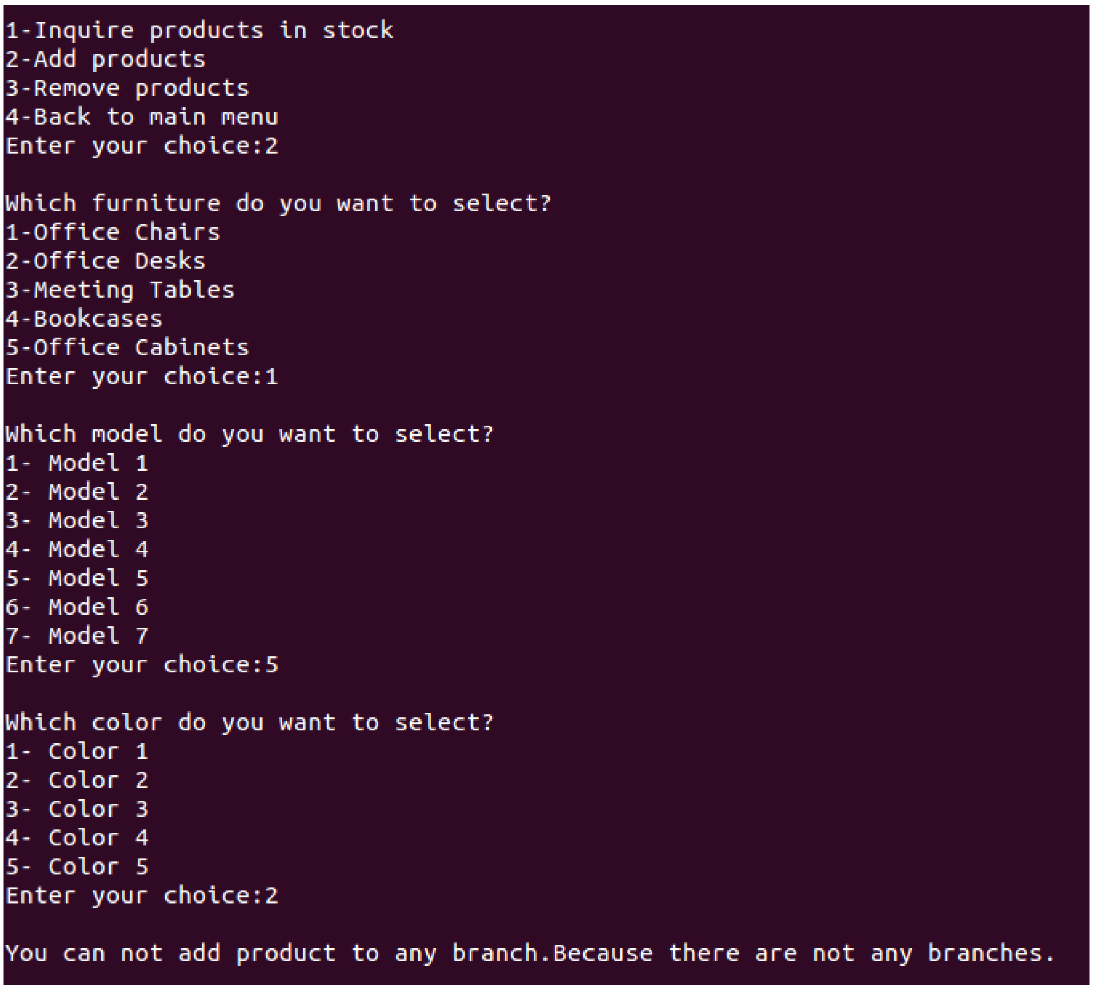
```
Which color do you want to inquire?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
Enter your choice:7
Wrong choice
```

8-When all branches was removed



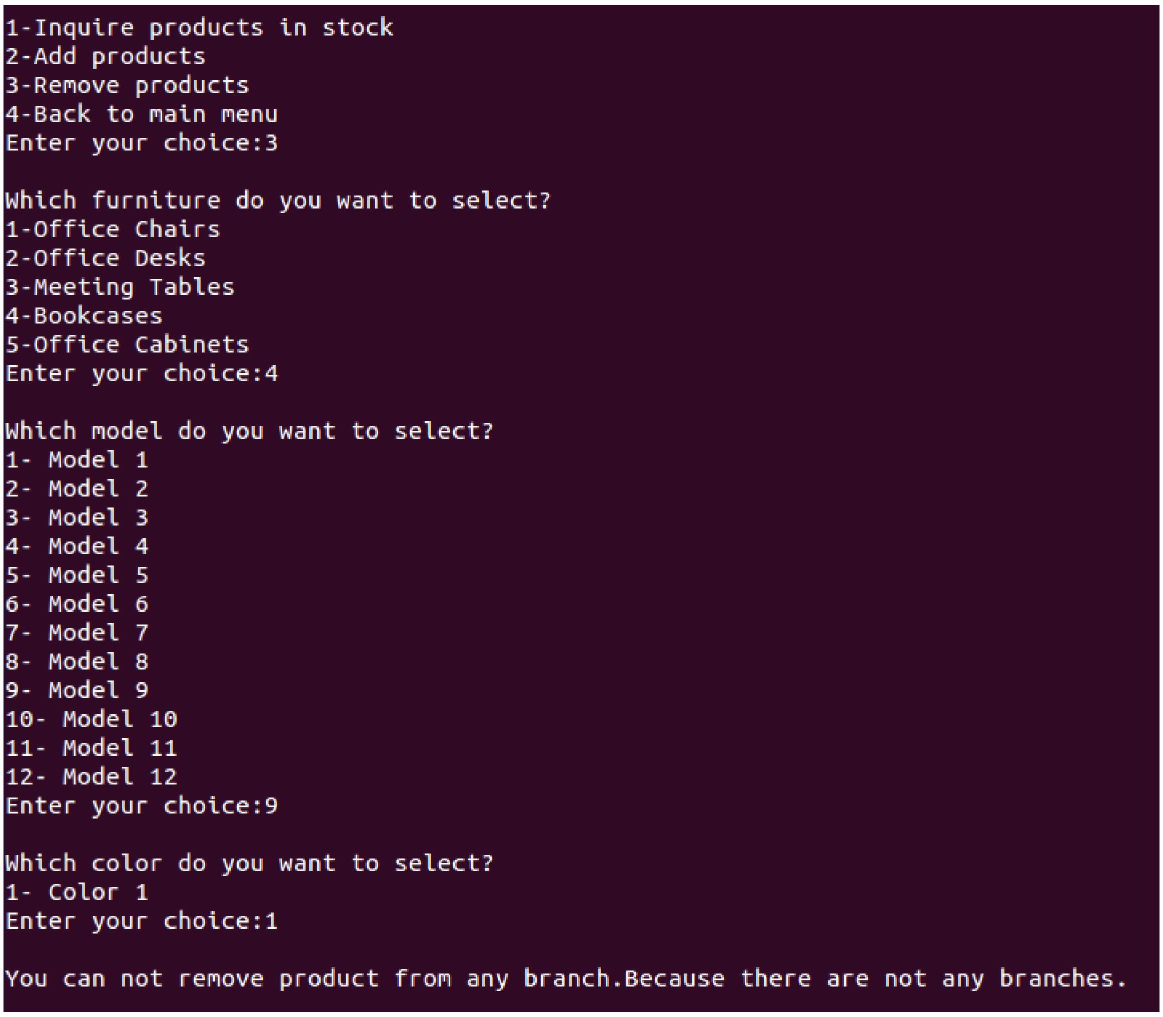
```
1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:1

Which furniture do you want to inquire
1-Office Chairs
2-Office Desks
3-Meeting Tables
4-Bookcases
5-Office Cabinets
Enter your choice:3

Which model do you want to inquire?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
Enter your choice:7

Which color do you want to inquire?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
Enter your choice:1

You can not inquire this product.Because there are not any branches.
```

9-When all branches was removed



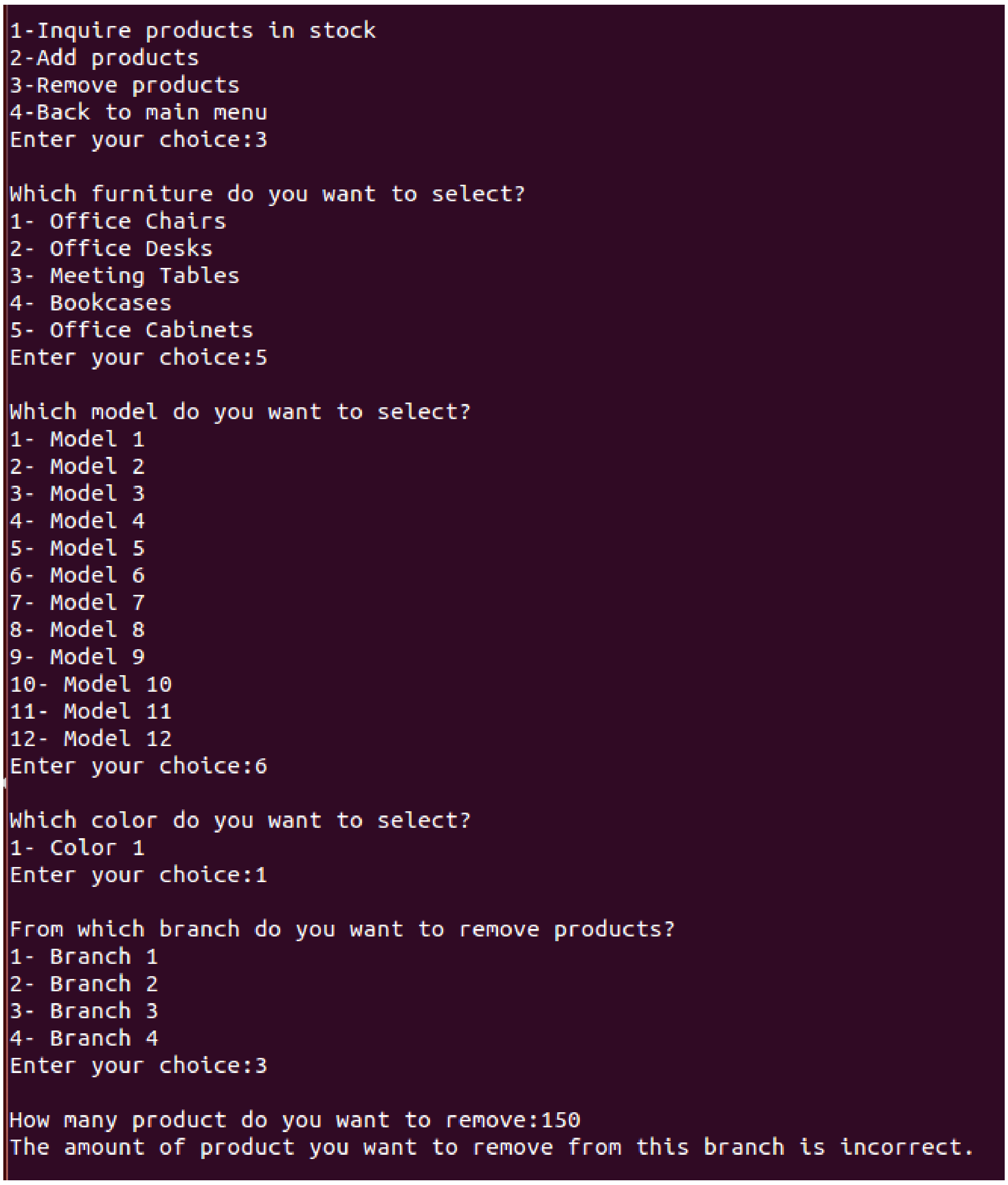
```
1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:2

Which furniture do you want to select?
1-Office Chairs
2-Office Desks
3-Meeting Tables
4-Bookcases
5-Office Cabinets
Enter your choice:1

Which model do you want to select?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
Enter your choice:5

Which color do you want to select?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
5- Color 5
Enter your choice:2

You can not add product to any branch.Because there are not any branches.
```

10-When all branches was removed



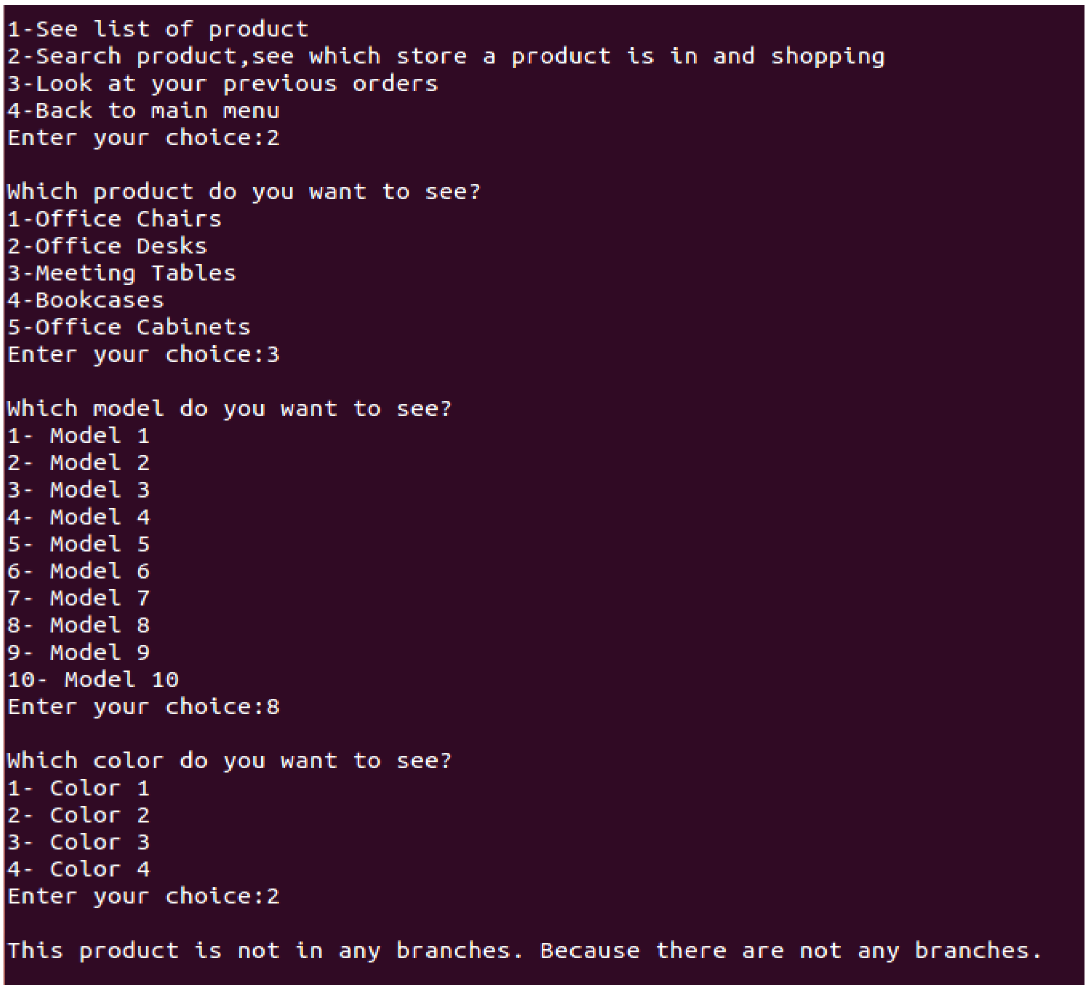
```
1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:3

Which furniture do you want to select?
1-Office Chairs
2-Office Desks
3-Meeting Tables
4-Bookcases
5-Office Cabinets
Enter your choice:4

Which model do you want to select?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
11- Model 11
12- Model 12
Enter your choice:9

Which color do you want to select?
1- Color 1
Enter your choice:1

You can not remove product from any branch.Because there are not any branches.
```

11-If the number of products to be removed is more than the number of products in the selected branch



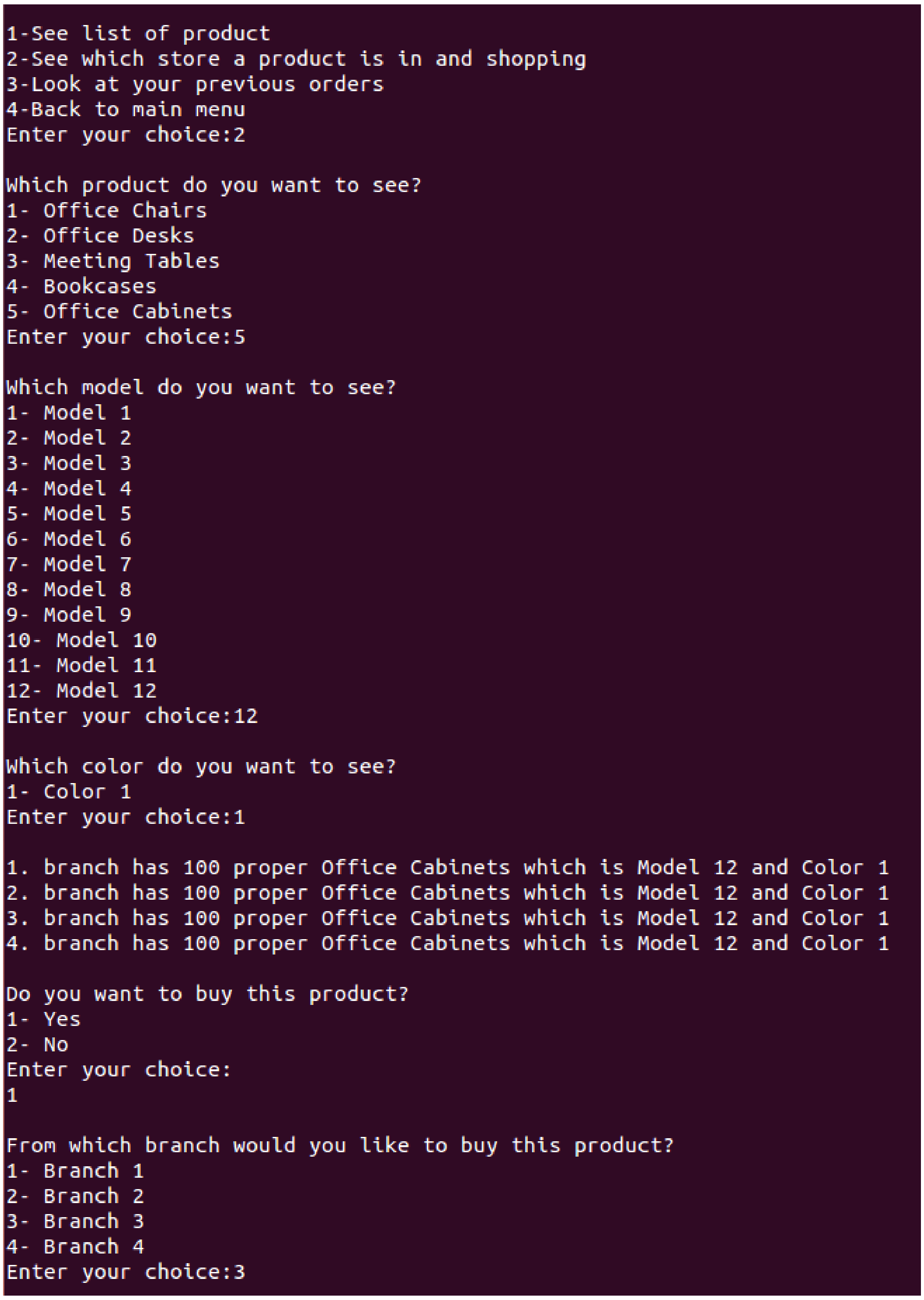
```
1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:3

Which furniture do you want to select?
1- Office Chairs
2- Office Desks
3- Meeting Tables
4- Bookcases
5- Office Cabinets
Enter your choice:5

Which model do you want to select?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
11- Model 11
12- Model 12
Enter your choice:6

Which color do you want to select?
1- Color 1
Enter your choice:1

From which branch do you want to remove products?
1- Branch 1
2- Branch 2
3- Branch 3
4- Branch 4
Enter your choice:3

How many product do you want to remove:150
The amount of product you want to remove from this branch is incorrect.
```

12-When all branches was removed

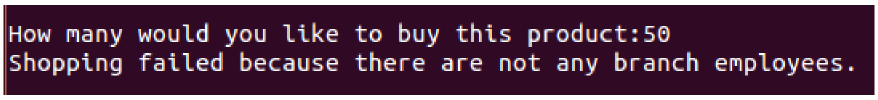


```
1-See list of product
2-Search product,see which store a product is in and shopping
3-Look at your previous orders
4-Back to main menu
Enter your choice:2

Which product do you want to see?
1-Office Chairs
2-Office Desks
3-Meeting Tables
4-Bookcases
5-Office Cabinets
Enter your choice:3

Which model do you want to see?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
Enter your choice:8

Which color do you want to see?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
Enter your choice:2

This product is not in any branches. Because there are not any branches.
```

13-When there are not any branch employees in the system(This case contains 2 photos)

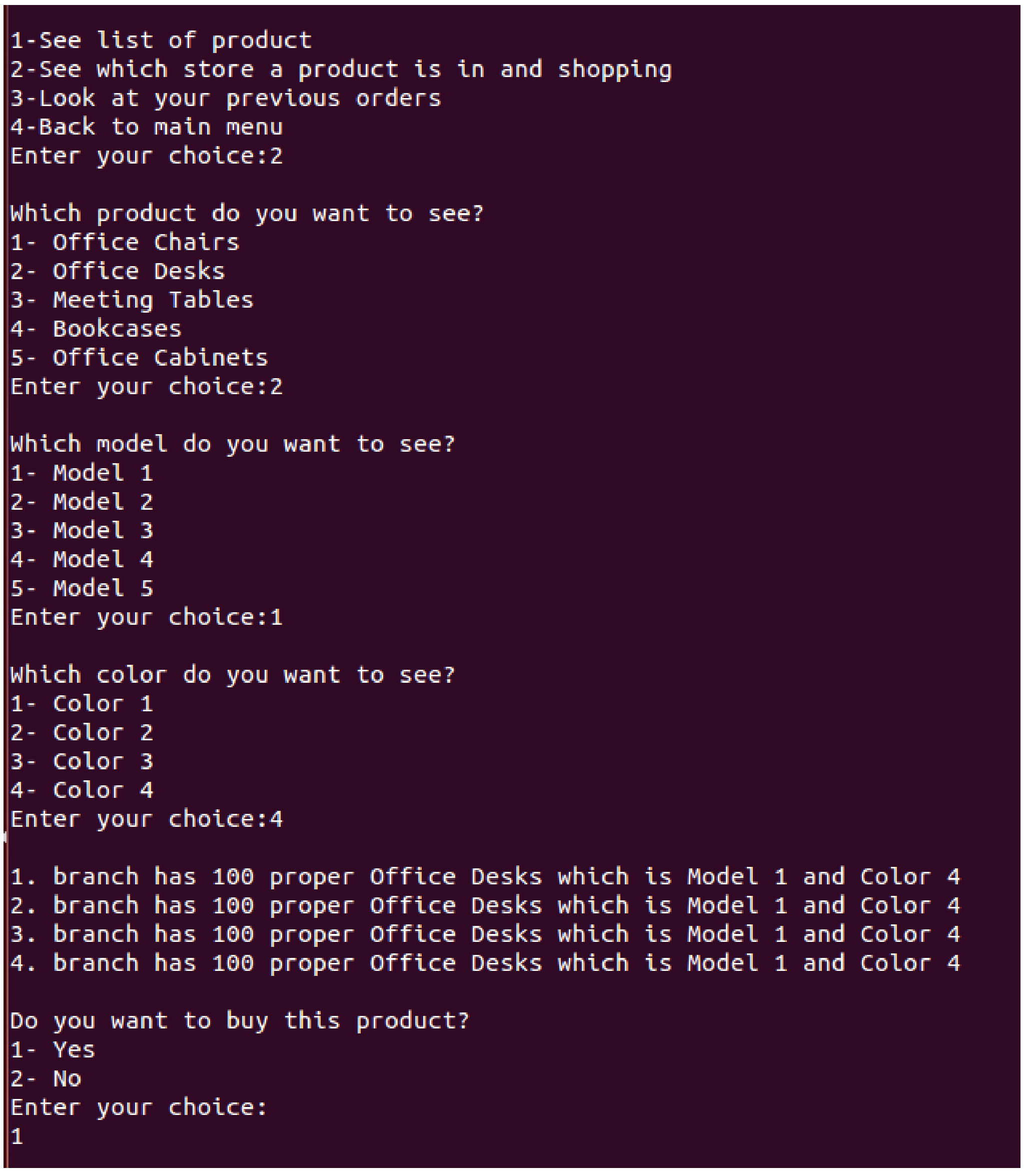


```
1-See list of product
2-See which store a product is in and shopping
3-Look at your previous orders
4-Back to main menu
Enter your choice:2

Which product do you want to see?
1- Office Chairs
2- Office Desks
3- Meeting Tables
4- Bookcases
5- Office Cabinets
Enter your choice:5

Which model do you want to see?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
11- Model 11
12- Model 12
Enter your choice:12

Which color do you want to see?
1- Color 1
Enter your choice:1

1. branch has 100 proper Office Cabinets which is Model 12 and Color 1
2. branch has 100 proper Office Cabinets which is Model 12 and Color 1
3. branch has 100 proper Office Cabinets which is Model 12 and Color 1
4. branch has 100 proper Office Cabinets which is Model 12 and Color 1

Do you want to buy this product?
1- Yes
2- No
Enter your choice:
1

From which branch would you like to buy this product?
1- Branch 1
2- Branch 2
3- Branch 3
4- Branch 4
Enter your choice:3
```

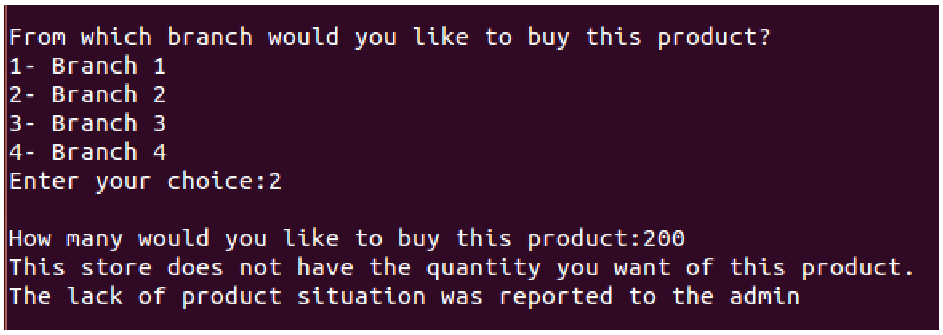


```
How many would you like to buy this product:50
Shopping failed because there are not any branch employees.
```

14-If the number of products to be purchased is more than the number of products in the selected branch(This case contains 2 photos)

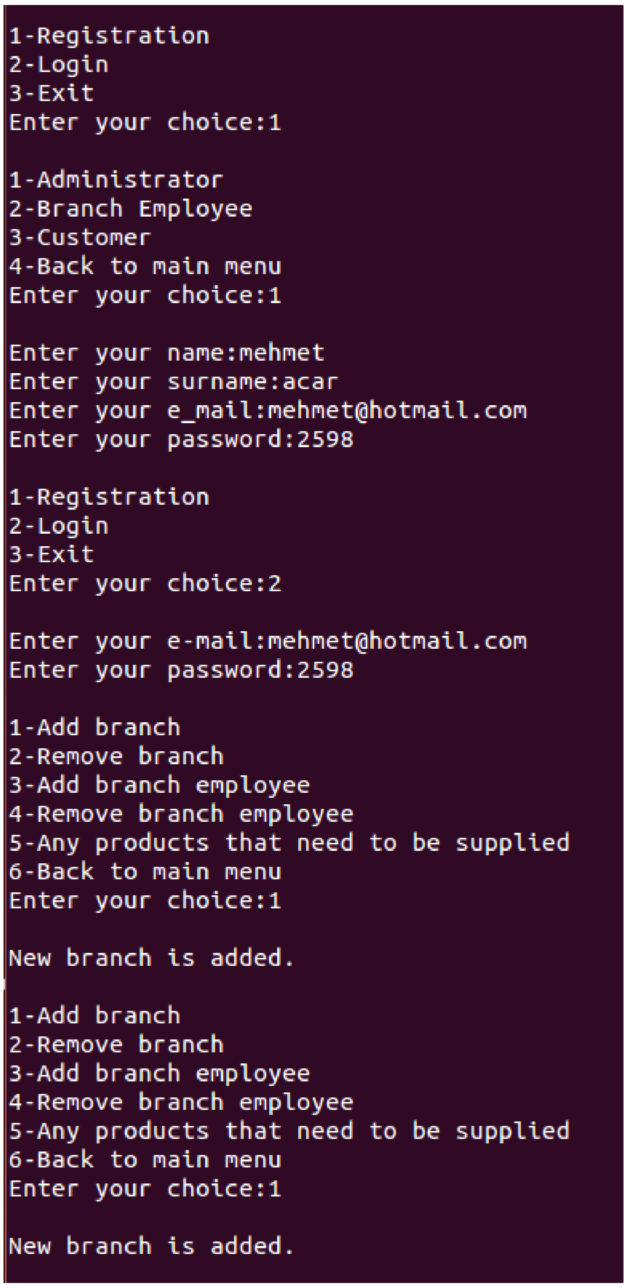


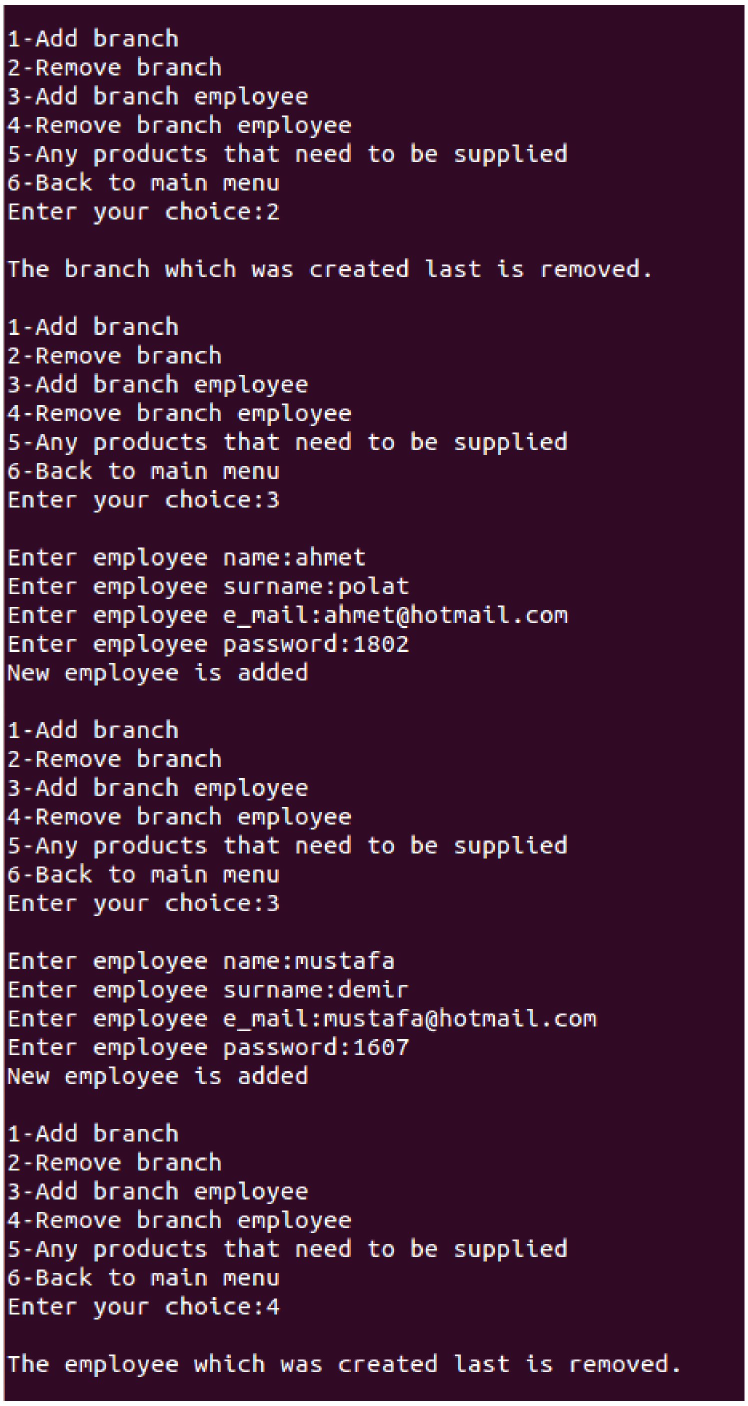
```
1-See list of product
2-See which store a product is in and shopping
3-Look at your previous orders
4-Back to main menu
Enter your choice:2

Which product do you want to see?
1- Office Chairs
2- Office Desks
3- Meeting Tables
4- Bookcases
5- Office Cabinets
Enter your choice:2

Which model do you want to see?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
Enter your choice:1

Which color do you want to see?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
Enter your choice:4

1. branch has 100 proper Office Desks which is Model 1 and Color 4
2. branch has 100 proper Office Desks which is Model 1 and Color 4
3. branch has 100 proper Office Desks which is Model 1 and Color 4
4. branch has 100 proper Office Desks which is Model 1 and Color 4

Do you want to buy this product?
1- Yes
2- No
Enter your choice:
1
```


```
From which branch would you like to buy this product?
1- Branch 1
2- Branch 2
3- Branch 3
4- Branch 4
Enter your choice:2

How many would you like to buy this product:200
This store does not have the quantity you want of this product.
The lack of product situation was reported to the admin
```

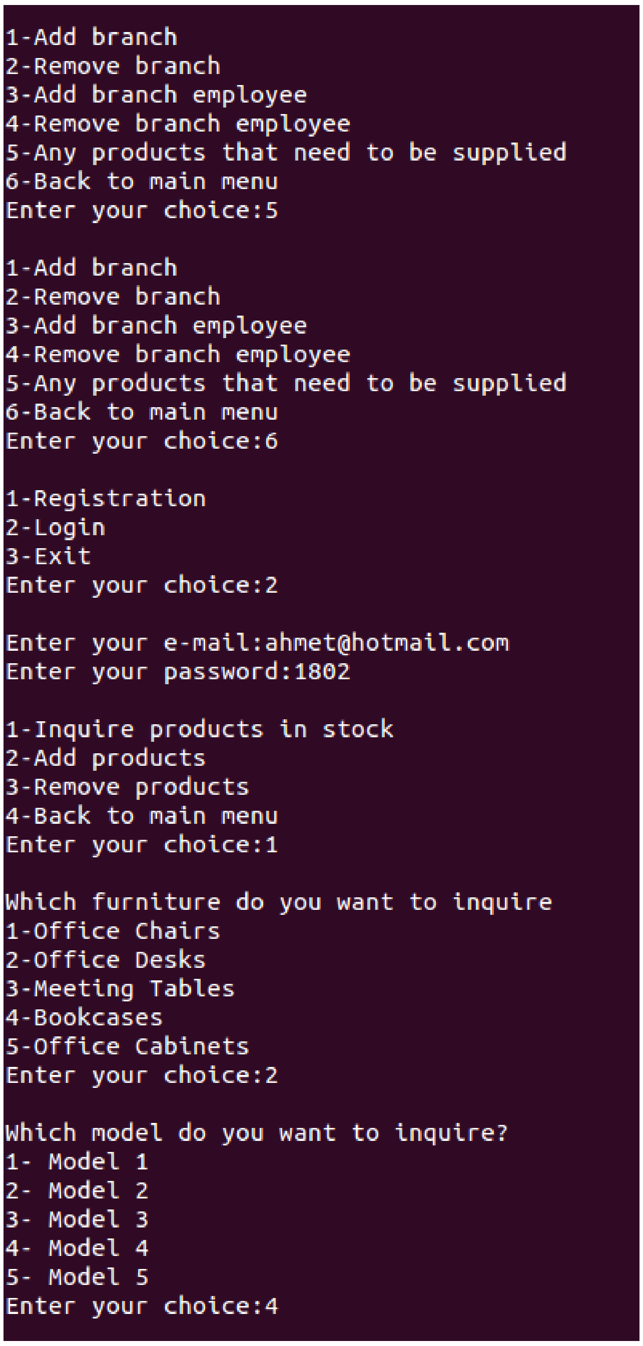
5-RUNNING AND RESULTS

Initially, each number of products is 100

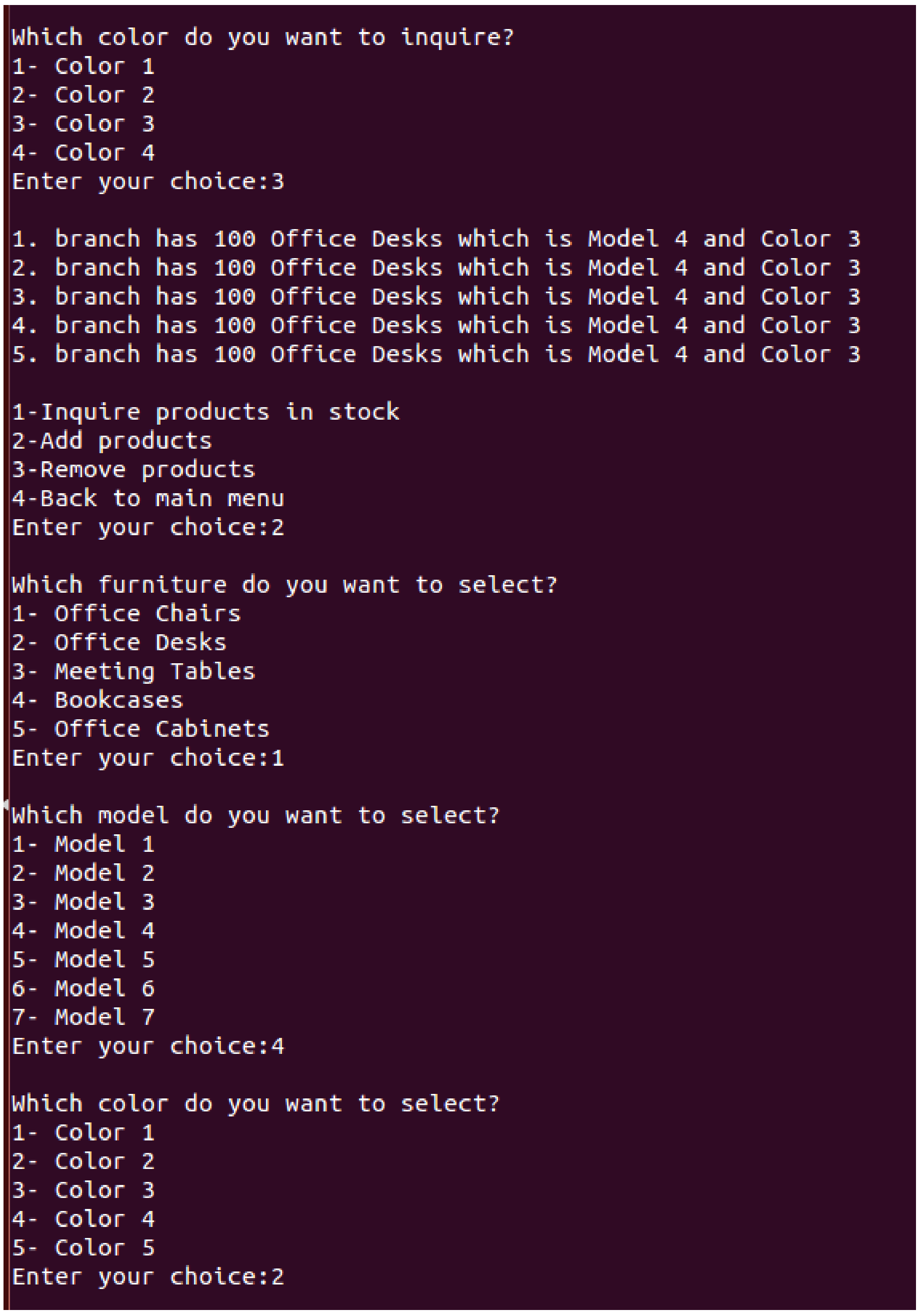

```
1-Registration
2-Login
3-Exit
Enter your choice:1

1-Administrator
2-Branch Employee
3-Customer
4-Back to main menu
Enter your choice:1

Enter your name:mehmet
Enter your surname:acar
Enter your e_mail:mehmet@hotmail.com
Enter your password:2598

1-Registration
2-Login
3-Exit
Enter your choice:2

Enter your e-mail:mehmet@hotmail.com
Enter your password:2598

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:1

New branch is added.

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:1

New branch is added.
```



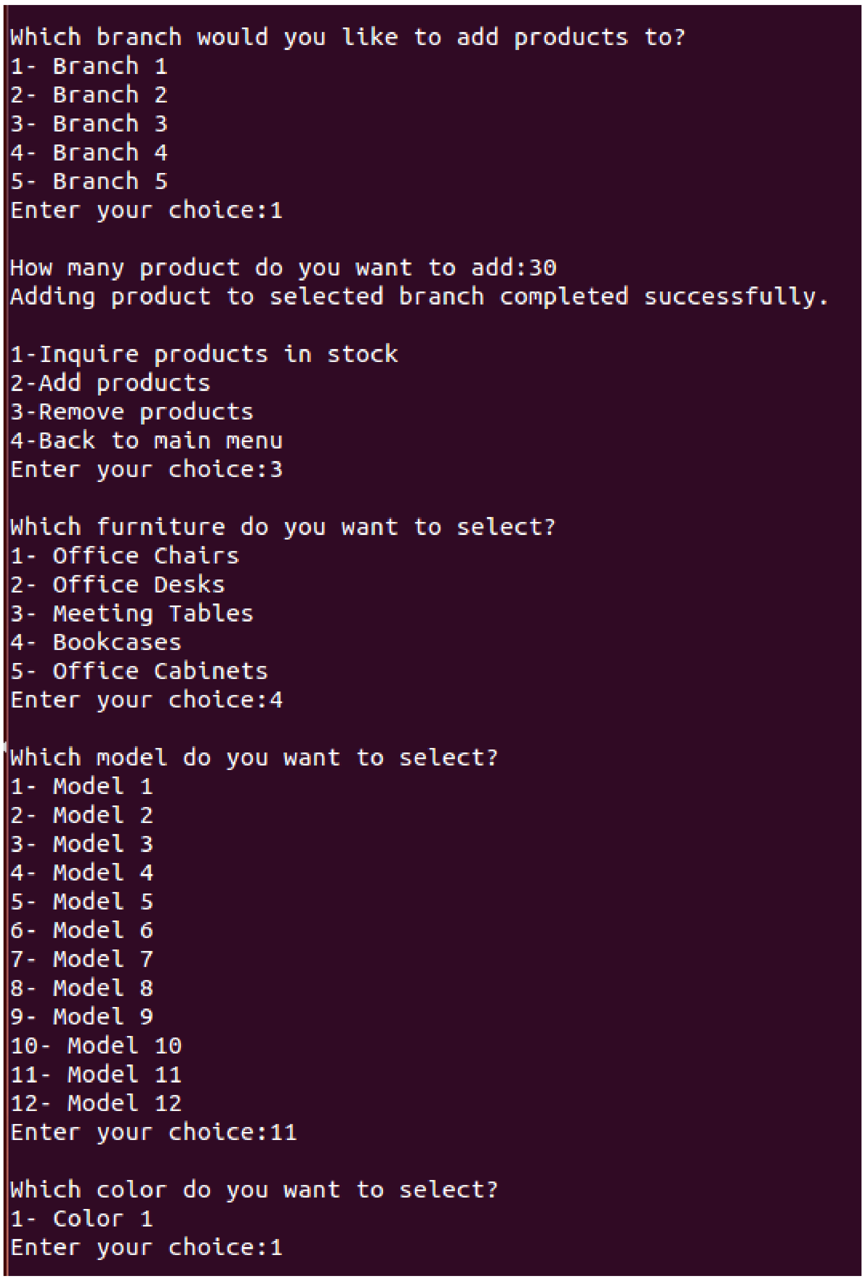
```
1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:2

The branch which was created last is removed.

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:3

Enter employee name:ahmet
Enter employee surname:polat
Enter employee e_mail:ahmet@hotmail.com
Enter employee password:1802
New employee is added

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:3

Enter employee name:mustafa
Enter employee surname:demir
Enter employee e_mail:mustafa@hotmail.com
Enter employee password:1607
New employee is added

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:4

The employee which was created last is removed.
```

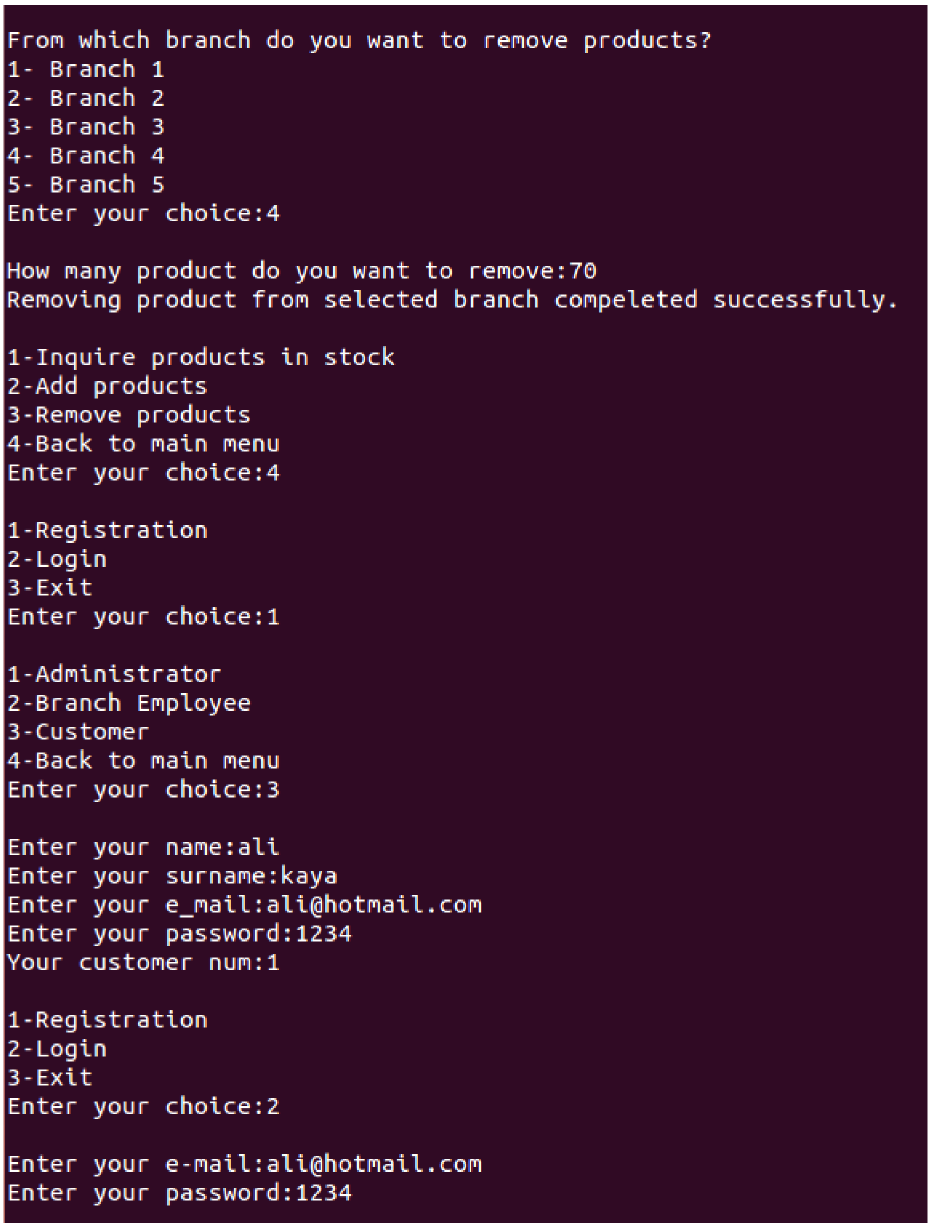


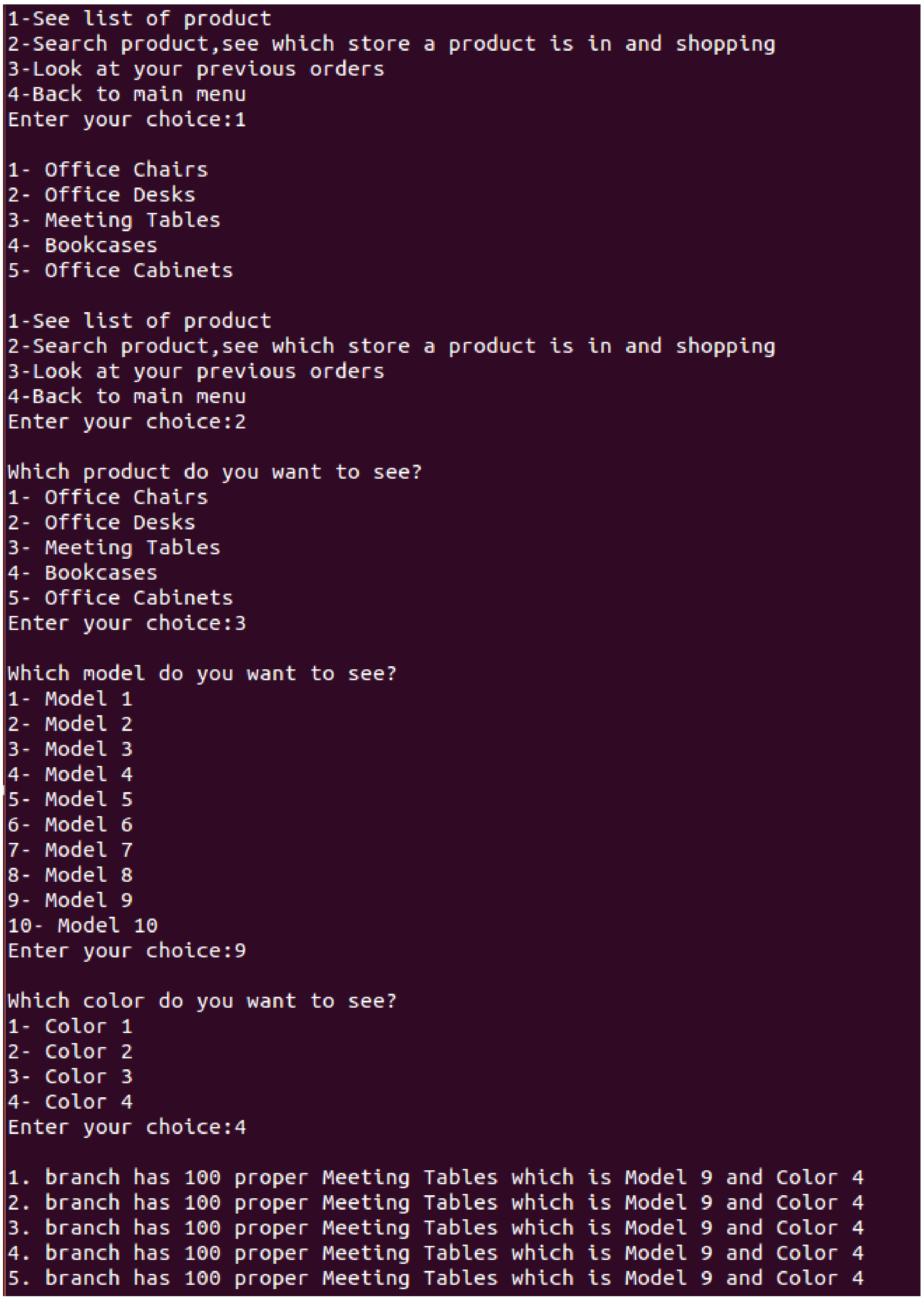
```
1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:5

1-Add branch
2-Remove branch
3-Add branch employee
4-Remove branch employee
5-Any products that need to be supplied
6-Back to main menu
Enter your choice:6

1-Registration
2-Login
3-Exit
Enter your choice:2

Enter your e-mail:ahmet@hotmail.com
Enter your password:1802

1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:1

Which furniture do you want to inquire
1-Office Chairs
2-Office Desks
3-Meeting Tables
4-Bookcases
5-Office Cabinets
Enter your choice:2

Which model do you want to inquire?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
Enter your choice:4
```



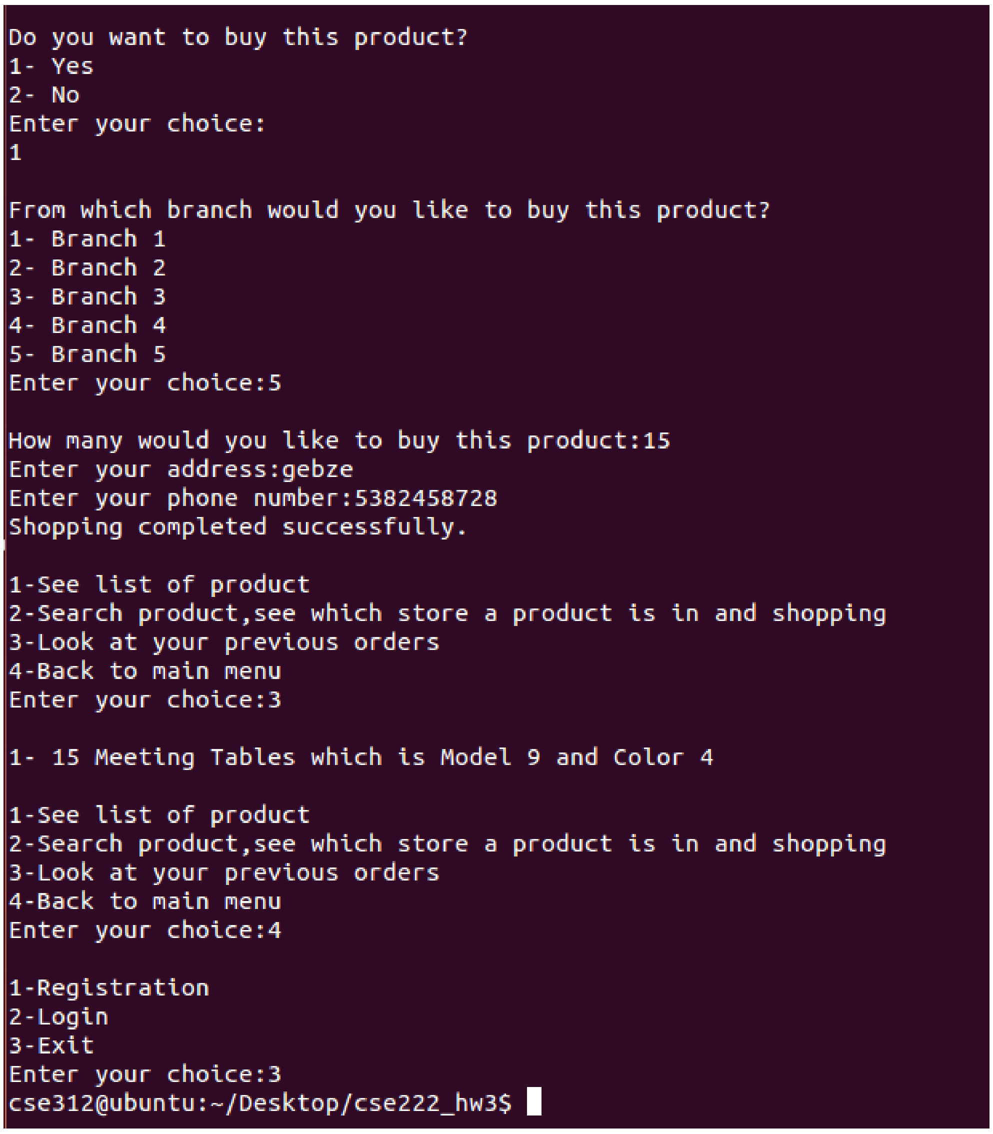
```
Which color do you want to inquire?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
Enter your choice:3

1. branch has 100 Office Desks which is Model 4 and Color 3
2. branch has 100 Office Desks which is Model 4 and Color 3
3. branch has 100 Office Desks which is Model 4 and Color 3
4. branch has 100 Office Desks which is Model 4 and Color 3
5. branch has 100 Office Desks which is Model 4 and Color 3

1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:2

Which furniture do you want to select?
1- Office Chairs
2- Office Desks
3- Meeting Tables
4- Bookcases
5- Office Cabinets
Enter your choice:1

Which model do you want to select?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
Enter your choice:4

Which color do you want to select?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
5- Color 5
Enter your choice:2
```


```
Which branch would you like to add products to?
1- Branch 1
2- Branch 2
3- Branch 3
4- Branch 4
5- Branch 5
Enter your choice:1

How many product do you want to add:30
Adding product to selected branch completed successfully.

1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:3

Which furniture do you want to select?
1- Office Chairs
2- Office Desks
3- Meeting Tables
4- Bookcases
5- Office Cabinets
Enter your choice:4

Which model do you want to select?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
11- Model 11
12- Model 12
Enter your choice:11

Which color do you want to select?
1- Color 1
Enter your choice:1
```


```
From which branch do you want to remove products?
1- Branch 1
2- Branch 2
3- Branch 3
4- Branch 4
5- Branch 5
Enter your choice:4

How many product do you want to remove:70
Removing product from selected branch compeleted successfully.

1-Inquire products in stock
2-Add products
3-Remove products
4-Back to main menu
Enter your choice:4

1-Registration
2-Login
3-Exit
Enter your choice:1

1-Administrator
2-Branch Employee
3-Customer
4-Back to main menu
Enter your choice:3

Enter your name:ali
Enter your surname:kaya
Enter your e_mail:ali@hotmail.com
Enter your password:1234
Your customer num:1

1-Registration
2-Login
3-Exit
Enter your choice:2

Enter your e-mail:ali@hotmail.com
Enter your password:1234
```



```
1-See list of product
2-Search product,see which store a product is in and shopping
3-Look at your previous orders
4-Back to main menu
Enter your choice:1

1- Office Chairs
2- Office Desks
3- Meeting Tables
4- Bookcases
5- Office Cabinets

1-See list of product
2-Search product,see which store a product is in and shopping
3-Look at your previous orders
4-Back to main menu
Enter your choice:2

Which product do you want to see?
1- Office Chairs
2- Office Desks
3- Meeting Tables
4- Bookcases
5- Office Cabinets
Enter your choice:3

Which model do you want to see?
1- Model 1
2- Model 2
3- Model 3
4- Model 4
5- Model 5
6- Model 6
7- Model 7
8- Model 8
9- Model 9
10- Model 10
Enter your choice:9

Which color do you want to see?
1- Color 1
2- Color 2
3- Color 3
4- Color 4
Enter your choice:4

1. branch has 100 proper Meeting Tables which is Model 9 and Color 4
2. branch has 100 proper Meeting Tables which is Model 9 and Color 4
3. branch has 100 proper Meeting Tables which is Model 9 and Color 4
4. branch has 100 proper Meeting Tables which is Model 9 and Color 4
5. branch has 100 proper Meeting Tables which is Model 9 and Color 4
```



```
Do you want to buy this product?
1- Yes
2- No
Enter your choice:
1

From which branch would you like to buy this product?
1- Branch 1
2- Branch 2
3- Branch 3
4- Branch 4
5- Branch 5
Enter your choice:5

How many would you like to buy this product:15
Enter your address:gebze
Enter your phone number:5382458728
Shopping completed successfully.

1-See list of product
2-Search product,see which store a product is in and shopping
3-Look at your previous orders
4-Back to main menu
Enter your choice:3

1- 15 Meeting Tables which is Model 9 and Color 4

1-See list of product
2-Search product,see which store a product is in and shopping
3-Look at your previous orders
4-Back to main menu
Enter your choice:4

1-Registration
2-Login
3-Exit
Enter your choice:3
cse312@ubuntu:~/Desktop/cse222_hw3$
```

KWArrayListUser.java Methods Time Complexity

1.

```
public KWArrayListUser() {
                    capacity = INITIAL_CAPACITY;
                    user_num = new String[capacity];
                    name = (E[]) new Object[capacity];
                    surname = (E[]) new Object[capacity];
                    e_mail = (E[]) new Object[capacity];
                    password = (E[]) new Object[capacity];
          }
```

T(n) = Ɵ(1)

2.

```
public boolean add(E name,E surname,E e_mail,E password) {
                    if (size == capacity) {
                    reallocate();
                    }
                    int user_num=size+1;
                    this.user_num[size]="User "+user_num;
                    this.name[size] = name;
                    this.surname[size] = surname;
                    this.e_mail[size] = e_mail;
                    this.password[size] = password;
                    size++;
                    return true;
                    }
```

$T_b(n)$ = Ɵ(1)

$T_w(n)$ = Ɵ(n)

$T_{amortized}(n)$ = Ɵ(1)

3.

```
public void add(int index,E name,E surname,E e_mail,E password) {
                    if (index < 0 || index > size) {
                    throw new ArrayIndexOutOfBoundsException(index);
                    }
                    if (size == capacity) {
                    reallocate();
                    }
                    // Shift data in elements from index to size - 1
                    for (int i = size; i > index; i--) {
                    this.name[i] = this.name[i - 1];
                    this.surname[i] = this.surname[i - 1];
                    this.e_mail[i] = this.e_mail[i - 1];
                    this.password[i] = this.password[i - 1];
                    }
                    // Insert the new item.
                    int user_num=size+1;
                    this.user_num[size]="User "+user_num;
                    this.name[index] = name;
                    this.surname[index] = surname;
                    this.e_mail[index] = e_mail;
                    this.password[index] = password;


                    size++;
          }
```

T(n) = O(n)

4.

```
public E getName(int index) {
                    if (index < 0 || index >= size) {
                    throw new ArrayIndexOutOfBoundsException(index);
                    }
```

```
        return this.name[index];
        }
    T(n) =  Θ(1)
```

Like this get method, other get methods' time compexity in this file is T(n) = Θ(1)

5.

```
public E setName(int index, E newName) {
        if (index < 0 || index >= size) {
        throw new ArrayIndexOutOfBoundsException(index);
    }

        E oldName = this.name[index];
        this.name[index] = newName;
        return oldName;
    }
    T(n) =  Θ(1)
```

Like this set method, other set methods' time compexity is T(n) = Θ(1)

6.

```
public String remove(int index) {
        int i;
        if (index < 0 || index >= size) {
        throw new ArrayIndexOutOfBoundsException(index);
        }
        String returnUser = this.user_num[index];
        for (i = index + 1; i < size; i++) {
          this.name[i - 1] = this.name[i];
        }
        for (i = index + 1; i < size; i++) {
                this.surname[i - 1] = this.surname[i];
        }
        for (i = index + 1; i < size; i++) {
                this.e_mail[i - 1] = this.e_mail[i];
        }
        for (i = index + 1; i < size; i++) {
                this.password[i - 1] = this.password[i];
        }
        size--;
        return returnUser;
}
T(n) = O(n)
```

KWSingleLinkedListBranch.java Methods Time Complexity

1.

```
public void addFirst(HybridList furniture) {
 head = new Node(furniture,head);
 int branch_num=size+1;
 head.branch_name="Branch "+branch_num;
 size++;
}
T(n) =  Θ(1)
```

2.

```
private void addAfter(Node node,HybridList furniture) {
node.next = new Node(furniture, node.next);
int branch_num=size+1;
node.next.branch_name="Branch "+branch_num;
size++;
}
T(n) =  Θ(1)
```

3.

```
private String removeAfter(Node node) {
        Node temp = node.next;
        if (temp != null) {
        node.next = temp.next;
        size--;
        return temp.branch_name;
        }
        else {
        return null;
        }
}
T(n) =  Θ(1)
```

4.

```
private String removeFirst() {
        Node temp = head;
        if (head != null) {
        head = head.next;
        }
        // Return data at old head or null if list is empty
        if (temp != null) {
        size--;
        return temp.branch_name;
        }
        else {
        return null;
        }
}
T(n) =  Θ(1)
```

5.

```
private Node getNode(int index) {
  Node node = head;
        for (int i = 0; i < index && node != null; i++) {
        node = node.next;
        }
  return node;
}
```

T(n) = O(n)

6.

```
public String get(int index,int furniture_index) {
        if (index < 0 || index >= size) {
        throw new IndexOutOfBoundsException(Integer.toString(index));
        }
        if(furniture_index<0 || furniture_index>4) {
                throw new IndexOutOfBoundsException(Integer.toString(furniture_index));
        }
        Node node = getNode(index);
        return node.furniture_name[furniture_index];
}
```

T(n) = O(n)

Like this method, get2 and get3 methods' time complexity in this file is T(n) = O(n)

7.

```
public int getSize() {
        return size;
}
```

T(n) = Θ(1)

8.

```
public int getProductNum(int index,int furniture_index,int model_index,int color_index) {

        if (index < 0 || index >= size) {
                throw new IndexOutOfBoundsException(Integer.toString(index));
                }
        Node node = getNode(index);

        if(furniture_index<0 || furniture_index>4) {
                throw new IndexOutOfBoundsException(Integer.toString(furniture_index));
        }
        if(model_index<0 || model_index>=node.model_num[furniture_index]) {
            throw new IndexOutOfBoundsException(Integer.toString(model_index));
        }
        if(color_index<0 || color_index>=node.color_num[furniture_index]) {
            throw new IndexOutOfBoundsException(Integer.toString(color_index));
        }

        if(furniture_index==0){
            return node.furniture1_product_num[model_index][color_index];
        }
        else if(furniture_index==1) {
            return node.furniture2_product_num[model_index][color_index];
        }
        else if(furniture_index==2) {
            return node.furniture3_product_num[model_index][color_index];
        }
        else if(furniture_index==3) {
            return node.furniture4_product_num[model_index][color_index];
        }
        else {
            return node.furniture5_product_num[model_index][color_index];
        }
    }
```

T(n) = O(n)

9.

```
    public String set(int index, int furniture_index,String furniture_name) {
        if (index < 0 || index >= size) {
        throw new IndexOutOfBoundsException(Integer.toString(index));
        }
        if(furniture_index<0 || furniture_index>4) {
        throw new IndexOutOfBoundsException(Integer.toString(furniture_index));
```

```
            }
            Node node = getNode(index);
            String result = node.furniture_name[furniture_index];
            node.furniture_name[furniture_index] = furniture_name;
            return result;
    }
```

T(n) = O(n)

Like this method, other set methods' time complexity in this file is T(n) = O(n)

10.

```
      public void add(int index, HybridList furniture) {
            if (index < 0 || index > size) {
            throw new IndexOutOfBoundsException(Integer.toString(index));
      }
       if (index == 0) {
              addFirst(furniture);
            }
       else {
                    Node node = getNode(index-1);
                    addAfter(node,furniture);
       }
    }
```

T(n) = O(n)

11.

```
    public boolean add(HybridList furniture) {
            add(size, furniture);
            return true;
    }
```

T(n) = O(n)

12.

```
    public String remove(int index) {
            String res;
```

```
        if(index==0) {
          res=removeFirst();
        }
        else {
            Node node = getNode(index-1);
            res=removeAfter(node);
        }
        return res;
    }
    T(n) = O(n)
```

13.

```
      public String remove() {
        String res=remove(0);
        return res;
    }
    T(n) =  Θ(1)
```

14.

```
public void add_product(int index,int furniture_index,int model_index,int color_index,int
product_val) {
        Node node = getNode(index);
        if(furniture_index==0) {
            node.furniture1_product_num[model_index][color_index]+=product_val;
        }
        else if(furniture_index==1) {
            node.furniture2_product_num[model_index][color_index]+=product_val;
        }
        else if(furniture_index==2) {
            node.furniture3_product_num[model_index][color_index]+=product_val;
        }
        else if(furniture_index==3) {
            node.furniture4_product_num[model_index][color_index]+=product_val;
```

```
                }
                else if(furniture_index==4) {
                        node.furniture5_product_num[model_index][color_index]+=product_val;
                }
        }
        T(n) =  O(n)
```

15.

```
public void remove_product(int index,int furniture_index,int model_index,int color_index,int
product_val) {
                Node node = getNode(index);
                if(furniture_index==0) {
                        node.furniture1_product_num[model_index][color_index]-=product_val;
                }
                else if(furniture_index==1) {
                        node.furniture2_product_num[model_index][color_index]-=product_val;
                }
                else if(furniture_index==2) {
                        node.furniture3_product_num[model_index][color_index]-=product_val;
                }
                else if(furniture_index==3) {
                        node.furniture4_product_num[model_index][color_index]-=product_val;
                }
                else if(furniture_index==4) {
                        node.furniture5_product_num[model_index][color_index]-=product_val;
                }
        }
        T(n) = O(n)
```

KWSingleLinkedListFurniture.java Methods Time Complexity

1.

```
public void addFirst(String furniture_name,int model_num,int color_num) {
                head = new Node(furniture_name,model_num,color_num,head);
                size++;
               }
```

T(n) = ϴ(1)

2.

```
private void addAfter(Node node, String furniture_name,int model_num,int color_num) {
               node.next = new Node(furniture_name,model_num,color_num, node.next);
               size++;
               }
```

T(n) = ϴ(1)

3.

```
private String removeAfter(Node node) {
                              Node temp = node.next;
                              if (temp != null) {
                              node.next = temp.next;
                              size--;
                              return temp.furniture_name.get(0);
                              }
                              else {
                              return null;
                              }
               }
```

T(n) = ϴ(1)

4.

```
private String removeFirst() {
                              Node temp = head;
                              if (head != null) {
                              head = head.next;
```

```
        }
        // Return data at old head or null if list is empty
        if (temp != null) {
        size--;
        return temp.furniture_name.get(0);
        }
        else {
        return null;
        }
    }
    T(n) =  Θ(1)
```

5.

```
private Node getNode(int index) {
      Node node = head;
        for (int i = 0; i < index && node != null; i++) {
        node = node.next;
        }
      return node;
    }
    T(n) =  O(n)
```

6.

```
public String get(int index) {
    if (index < 0 || index >= size) {
    throw new IndexOutOfBoundsException(Integer.toString(index));
    }
    Node node = getNode(index);
    return node.furniture_name.get(0);
    }
    T(n) =  O(n)
```

Like this method, get2 and get3 methods’ time complexity in this file is T(n) = O(n)

7.

```
public int getSize() {
    return size;
}
```

T(n) = Θ(1)

8.

```
public String set(int index, String furniture_name) {
                if (index < 0 || index >= size) {
                throw new IndexOutOfBoundsException(Integer.toString(index));
                }
                Node node = getNode(index);
                String result = node.furniture_name.get(0);
                node.furniture_name.set(0,furniture_name);
                return result;
        }
```

T(n) = O(n)

Like this method, set2 and set3 methods' time complexity in this file is T(n) = O(n)

9.

```
public void add(int index, String furniture_name,int model_num,int color_num) {
                if (index < 0 || index > size) {
                throw new IndexOutOfBoundsException(Integer.toString(index));
          }
          if (index == 0) {
                  addFirst(furniture_name,model_num,color_num);
                }
          else {
                        Node node = getNode(index-1);
                        addAfter(node,furniture_name,model_num,color_num);
          }
        }
```

T(n) = O(n)

10.

```
public boolean add(String furniture_name,int model_num,int color_num) {
        add(size, furniture_name,model_num,color_num);
        return true;
}
```

T(n) = O(n)

11.

```
public String remove(int index) {
        String res;
        if(index==0) {
          res=removeFirst();
        }
        else {
                Node node = getNode(index-1);
                res=removeAfter(node);
        }
        return res;
}
```

T(n) = O(n)

12.

```
public String remove() {
        String res=remove(0);
        return res;
}
```

T(n) = Θ(1)

KWArrayListFurniture.java Methods Time Complexity

1.

```
public KWArrayListFurniture() {
            capacity = INITIAL_CAPACITY;
            theData = (E[]) new Object[capacity];
}
```

$T(n) = \Theta(1)$

2.

```
public boolean add(E anEntry) {
            if (size == capacity) {
                reallocate();
            }
                theData[size]=anEntry;
                size++;
                return true;
}
```

$T_b(n) = \Theta(1)$

$T_w(n) = \Theta(n)$

$T_{amortized}(n) = \Theta(1)$

3.

```
public void add(int index, E anEntry) {
            if (index < 0 || index > size) {
                        throw new ArrayIndexOutOfBoundsException(index);
            }
                        if (size == capacity) {
                            reallocate();
                        }
                        // Shift data in elements from index to size - 1
                        for (int i = size; i > index; i--) {
```

```
                        theData[i] = theData[i - 1];
                    }
                    // Insert the new item.
                    theData[index] = anEntry;
                    size++;
            }
            T(n) = O(n)
```

4.

```
public E get(int index) {
            if (index < 0 || index >= size) {
                throw new ArrayIndexOutOfBoundsException(index);
            }
            return theData[index];
}
            T(n) =  Θ(1)
```

5.

```
public E set(int index, E newValue) {
            if (index < 0 || index >= size) {
                throw new ArrayIndexOutOfBoundsException(index);
             }
                E oldValue = theData[index];
                theData[index] = newValue;
                return oldValue;
}
            T(n) =  Θ(1)
```

6.

```
public E remove(int index) {
            if (index < 0 || index >= size) {
               throw new ArrayIndexOutOfBoundsException(index);
            }
               E returnValue = theData[index];
            for (int i = index + 1; i < size; i++) {
                theData[i - 1] = theData[i];
             }
             size--;
             return returnValue;
}
        T(n) = O(n)
```

HybridList.java Methods Time Complexity

1.

```
public String get(int index) {
        String furniture_name;
        furniture_name=furniture.get(index);
        return furniture_name;
  }
        T(n) = O(n)
```

Like this method, get2 and get3 methods’ time complexity in this file is T(n) = O(n)

2.

```
public int getSize() {
     return furniture.getSize();
}
        T(n) =  Θ(1)
```

3.

```
public String set(int index, String furniture_name) {
        String result;
        result=furniture.set(index, furniture_name);
        return result;
}
        T(n) = O(n)
```

Like this method, set2 and set3 methods’ time complexity in this file is T(n) = O(n)

4.

```
public void add(int index, String furniture_name,int model_num,int color_num) {
        furniture.add(index,furniture_name,model_num,color_num);
}
        T(n) = O(n)
```

5.

```
  public String remove(int index) {
        String res;
        res=furniture.remove(index);
        return res;
}
        T(n) = O(n)
```

Administrator.java Methods Time Complexity

1.

```
public Administrator(int index,E name,E surname,E e_mail,E password) {
       this.add(index,name, surname, e_mail, password);
}
        T(n) = O(n)
```

2.

```
public void addAdministrator(int index,E name,E surname,E e_mail,E password) {
        this.add(index,name, surname, e_mail, password);
}
        T(n) = O(n)
```

3.

```
public void addBranch(KWSingleLinkedListBranch branch,HybridList furniture) {
        branch.add(branch.getSize(), furniture);
}
        T(n) = O(n)
```

4.

```
public String removeBranch(KWSingleLinkedListBranch branch) {
        return branch.remove(branch.getSize()-1);
}
```

$T(n) = O(n)$

5.

```
public boolean add_Employee(Employee<E> employee,E name,E surname,E e_mail,E password) {
         return employee.add(name, surname, e_mail, password);
}
```

$T_b(n) = \Theta(1)$

$T_w(n) = \Theta(n)$

$T_{amortized}(n) = \Theta(1)$

6.

```
public E get_Name(Employee<E> employee) {
        return employee.getName(employee.getSize()-1);
}
```

$T(n) = \Theta(1)$

7.

```
public E get_Surname(Employee<E> employee) {
        return employee.getSurname(employee.getSize()-1);
}
```

$T(n) = \Theta(1)$

8.

```
public String remove_Employee(Employee<E> employee) {
        return employee.remove(employee.getSize()-1);
}
```

$T(n) = O(n)$

9.

```
public int getBranchSize(KWSingleLinkedListBranch branch) {
                return branch.getSize();   -> T(n) = Θ(1)
}
```

10.

```
public int getModelNum(HybridList furniture,int index) {
        return furniture.get2(index);
}
```

T(n) = O(n)

11.

```
public int getColorNum(HybridList furniture,int index) {
        return furniture.get3(index);
}
```

T(n) = O(n)

12.

```
public int get_ProductNum(KWSingleLinkedListBranch branch,int branch_index,int
furniture_index,int model_index,int color_index) {
        return branch.getProductNum(branch_index, furniture_index, model_index, color_index);
}
```

T(n) = O(n)

13.

```
public String getFurnitureName(KWSingleLinkedListBranch branch,int branch_index,int
furniture_index) {
        return branch.get(branch_index, furniture_index);
}
```

T(n) = O(n)

14.

```
public int getFurnitureNum(HybridList furniture) {
        return furniture.getSize();
}
```

T(n) = ϴ(1)

15.

```
public void MessageByEmployee() {
        System.out.println("The lack of product situation was reported to the admin");
}
```

T(n) = O(1)

Employee.java Methods Time Complexity

1.

```
public void addEmployee(int index,E name,E surname,E e_mail,E password) {
            this.add(index, name, surname, e_mail, password);
 }
```

$T(n) = O(n)$

2.

```
public boolean addEmployee(E name,E surname,E e_mail,E password) {
             return this.add(name, surname, e_mail, password);
}
```

$T_b(n) = \Theta(1)$

$T_w(n) = \Theta(n)$

$T_{amortized}(n) = \Theta(1)$

3.

```
public void set_CustomerOrder(int customer_num,String product_name,int product_model,int product_color,int product_num) {
                                    Customer.customer_order_name[customer_num-1][Customer.customer_order_num[customer_num-1]]=product_name;
                                    Customer.customer_order_model[customer_num-1][Customer.customer_order_num[customer_num-1]]=product_model;
                                    Customer.customer_order_color[customer_num-1][Customer.customer_order_num[customer_num-1]]=product_color;
                                    Customer.customer_product_num[customer_num-1][Customer.customer_order_num[customer_num-1]]=product_num;
                                    Customer.customer_order_num[customer_num-1]++;
}
```

$T(n) = \Theta(1)$

4.

```
public int getBranchSize(KWSingleLinkedListBranch branch) {
            return branch.getSize();
}
```

$T(n) = \Theta(1)$

5.

```
public int getFurnitureSize(HybridList furniture) {
        return furniture.getSize();
}
```

T(n) = ϴ(1)

6.

```
public int getModelNum(HybridList furniture,int furniture_index) {
        return furniture.get2(furniture_index);
}
```

T(n) = O(n)

7.

```
public int getColorNum(HybridList furniture,int furniture_index) {
        return furniture.get3(furniture_index);
}
```

T(n) = O(n)

8.

```
public int get_ProductNum(KWSingleLinkedListBranch branch,int branch_index,int
furniture_index,int model_index,int color_index) {
        return branch.getProductNum(branch_index, furniture_index, model_index, color_index);
}
```

T(n) = O(n)

9.

```
public String getFurnitureName(HybridList furniture,int index) {
          return furniture.get(index);
}
```

T(n) = O(n)

10.

```
public String getFurnitureName(KWSingleLinkedListBranch branch,int branch_index,int
furniture_index) {
          String furniture_name=branch.get(branch_index, furniture_index);
```

```
        return furniture_name;
}
        T(n) = O(n)
```

11.

```
public void add(KWSingleLinkedListBranch branch,int branch_index,int furniture_index,int
model_index,int color_index,int product_val) {
        branch.add_product(branch_index, furniture_index, model_index, color_index, product_val);
}
        T(n) = O(n)
```

12.

```
public void remove(KWSingleLinkedListBranch branch,int branch_index,int furniture_index,int
model_index,int color_index,int product_val) {
branch.remove_product(branch_index, furniture_index, model_index, color_index, product_val);
}
        T(n) = O(n)
```

13.

```
public int get_EmployeeNum() {
        return this.getSize();
}
        T(n) = Θ(1)
```

14.

```
public void informAdmin(Administrator<E> admin) {
        admin.MessageByEmployee();
}
        T(n) = O(1)
```

Customer.java Methods Time Complexity

1.

```
public Customer(int index,E name,E surname,E e_mail,E password) {
        this.add(index, name, surname, e_mail, password);
        this.customer_num[index]=customer_object;
        customer_object++;
}
            T(n) = O(n)
2.
public void addCustomer(int index,E name,E surname,E e_mail,E password) {
            this.add(index,name, surname, e_mail, password);
            this.customer_num[index]=customer_object;
            customer_object++;
}
            T(n) = O(n)
3.
public void showInfos(int index) {
            System.out.println("Your customer num:" + this.customer_num[index]);
}
            T(n) = O(1)
4.
public int getCustomer_num(int index) {
            return customer_num[index];
}
            T(n) = ϴ(1)
5.
public void setCustomer_num(int index,int customer_num) {
            this.customer_num[index] = customer_num;
}
            T(n) = ϴ(1)
6.
public int getFurnitureNum(HybridList furniture) {
              return furniture.getSize();
 }
            T(n) = ϴ(1)
```

7.

```
public String getFurnitureName(HybridList furniture,int index) {
        return furniture.get(index);
}
```

T(n) = O(n)

8.

```
public int get_ProductNum(KWSingleLinkedListBranch branch,int branch_index,int
furniture_index,int model_index,int color_index) {
        return branch.getProductNum(branch_index, furniture_index, model_index, color_index);
}
```

T(n) = O(n)

9.

```
public String getFurnitureName(KWSingleLinkedListBranch branch,int branch_index,int
furniture_index) {
        return branch.get(branch_index, furniture_index);
}
```

T(n) = O(n)

10.

```
public int getEmployeeNum(Employee<E> employee) {
        return employee.get_EmployeeNum();
}
```

T(n) = Θ(1)

11.

```
public int getBranchSize(KWSingleLinkedListBranch branch) {
        return branch.getSize();
}
```

T(n) = Θ(1)

12.

```
public int getModelNum(HybridList furniture,int furniture_index) {
        return furniture.get2(furniture_index);
}
```

T(n) = O(n)

13.

```
public int getColorNum(HybridList furniture,int furniture_index) {
	return furniture.get3(furniture_index);
}
```

T(n) = O(n)

14.

```
@Override
public String getAddress(int index) {
	return address[index];
}
```

T(n) = Θ(1)

15.

```
@Override
public void setAddress(int index,String address) {
	this.address[index] = address;
}
```

T(n) = Θ(1)

16.

```
@Override
public String getPhone_num(int index) {
	return phone_num[index];
}
```

T(n) = Θ(1)

17.

```
@Override
public void setPhone_num(int index,String phone_num) {
	this.phone_num[index] = phone_num;
}
```

T(n) = Θ(1)

18.

```
@Override
public void shopInfos(int index) {
        String address,phone_num;
        Scanner scanner=new Scanner(System.in);
        System.out.printf("Enter your address:");
        address=scanner.nextLine();
        System.out.printf("Enter your phone number:");
        phone_num=scanner.nextLine();
        setAddress(index,address);
        setPhone_num(index,phone_num);
}
```

T(n) = Θ(1)

19.

```
@Override
public void shopInfos(int index,String address,String phone_num) {
        setAddress(index,address);
        setPhone_num(index,phone_num);
}
```

T(n) = Θ(1)

20.

```
public void remove(KWSingleLinkedListBranch branch,int branch_index,int furniture_index,int
model_index,int color_index,int product_val) {
branch.remove_product(branch_index, furniture_index, model_index, color_index, product_val);
}
```

T(n) = O(n)

21.

```
public String getProduct_name(int customer_num) {
        return customer_order_name[customer_num-1][customer_order_num[customer_num-1]-1];
}
```

T(n) = Θ(1)

22.

```
public int getProduct_model(int customer_num) {
return customer_order_model[customer_num-1][customer_order_num[customer_num-1]-1];
}
```

T(n) = Θ(1)

23.

```
public int getProduct_color(int customer_num) {
return customer_order_color[customer_num-1][customer_order_num[customer_num-1]-1];
}
```

T(n) = Θ(1)

24.

```
public int getProduct_num(int customer_num) {
return customer_product_num[customer_num-1][customer_order_num[customer_num-1]-1];
}
```

T(n) = Θ(1)

25.

```
public void print_orders(int customer_num) {


            int i;
           System.out.printf("\n");
           for(i=0;i<customer_order_num[customer_num-1];i++) {
               System.out.printf("%d- %d ",i+1,customer_product_num[customer_num-1][i]);
               System.out.printf("%s ",customer_order_name[customer_num-1][i]);
               System.out.printf("which is Model %d ",customer_order_model[customer_num-1][i]);
               System.out.printf("and Color %d\n",customer_order_color[customer_num-1][i]);
           }
}
```

T(n) = O(n)

Mehmet Acar

1801042095