GTU Department of Computer Engineering
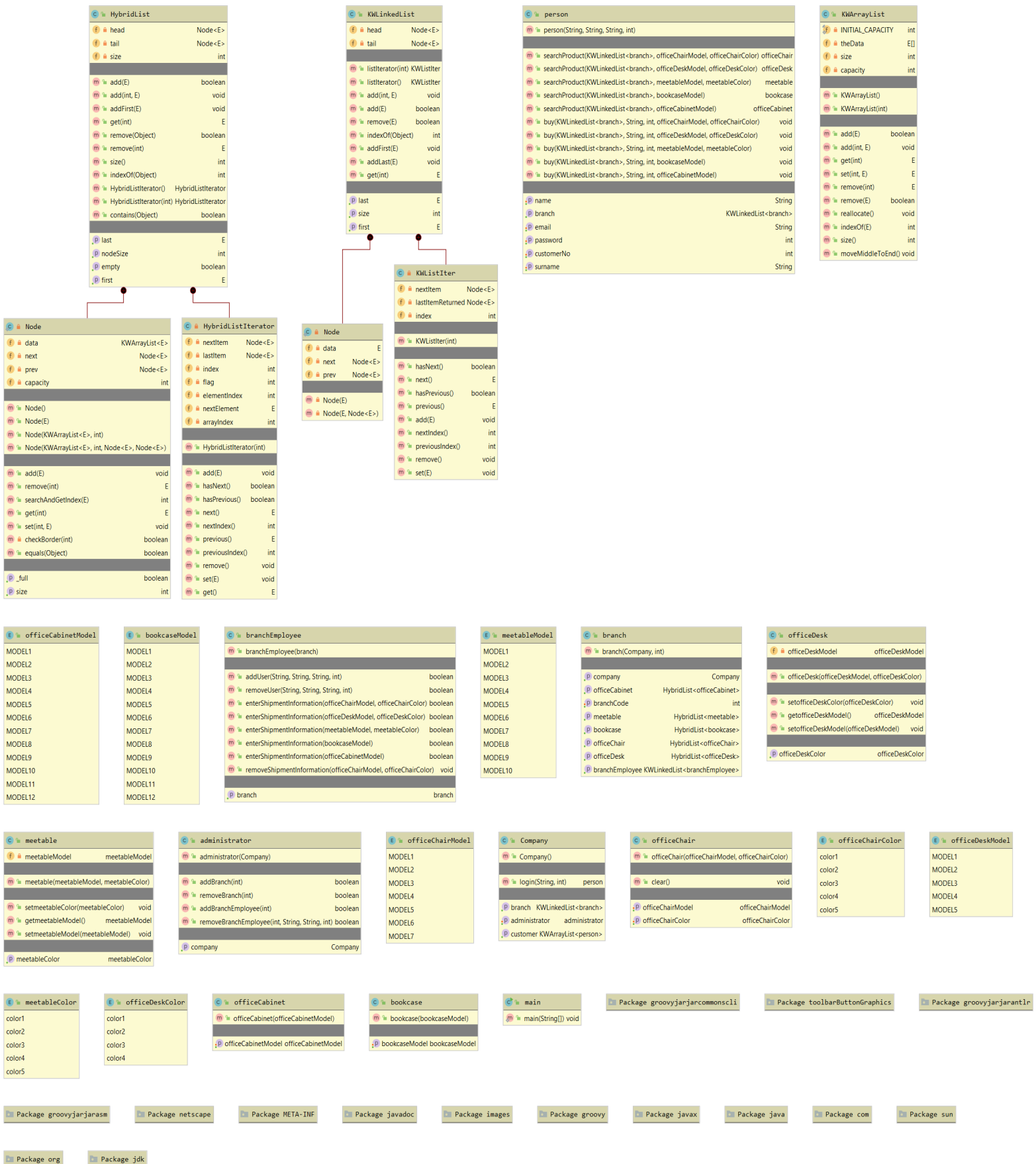# CSE 222/505 - Spring 2021
Homework 3 Report

Refik Orkun Arslan
151044063

Class Diagram

HybridList
- f 🔒 head — Node<E>
- f 🔒 tail — Node<E>
- f 🔒 size — int

- m 🔒 add(E) — boolean
- m 🔒 add(int, E) — void
- m 🔒 addFirst(E) — void
- m 🔒 get(int) — E
- m 🔒 remove(Object) — boolean
- m 🔒 remove(int) — E
- m 🔒 size() — int
- m 🔒 indexOf(Object) — int
- m 🔒 HybridListIterator() — HybridListIterator
- m 🔒 HybridListIterator(int) — HybridListIterator
- m 🔒 contains(Object) — boolean

- p 🔒 last — E
- p 🔒 nodeSize — int
- p 🔒 empty — boolean
- p 🔒 first — E

KWLinkedList
- f 🔒 head — Node<E>
- f 🔒 tail — Node<E>

- m 🔒 listIterator(int) KWListIter
- m 🔒 listIterator() KWListIter
- m 🔒 add(int, E) — void
- m 🔒 add(E) — boolean
- m 🔒 remove(E) — boolean
- m 🔒 indexOf(Object) — int
- m 🔒 addFirst(E) — void
- m 🔒 addLast(E) — void
- m 🔒 get(int) — E

- p 🔒 last — E
- p 🔒 size — int
- p 🔒 first — E

person
- m 🔒 person(String, String, String, int)

- m 🔒 searchProduct(KWLinkedList<branch>, officeChairModel, officeChairColor) officeChair
- m 🔒 searchProduct(KWLinkedList<branch>, officeDeskModel, officeDeskColor) officeDesk
- m 🔒 searchProduct(KWLinkedList<branch>, meetableModel, meetableColor) meetable
- m 🔒 searchProduct(KWLinkedList<branch>, bookcaseModel) bookcase
- m 🔒 searchProduct(KWLinkedList<branch>, officeCabinetModel) officeCabinet
- m 🔒 buy(KWLinkedList<branch>, String, int, officeChairModel, officeChairColor) void
- m 🔒 buy(KWLinkedList<branch>, String, int, officeDeskModel, officeDeskColor) void
- m 🔒 buy(KWLinkedList<branch>, String, int, meetableModel, meetableColor) void
- m 🔒 buy(KWLinkedList<branch>, String, int, bookcaseModel) void
- m 🔒 buy(KWLinkedList<branch>, String, int, officeCabinetModel) void

- p 🔒 name — String
- p 🔒 branch — KWLinkedList<branch>
- p 🔒 email — String
- p 🔒 password — int
- p 🔒 customerNo — int
- p 🔒 surname — String

KWArrayList
- f 🔒 INITIAL_CAPACITY — int
- f 🔒 theData — E[]
- f 🔒 size — int
- f 🔒 capacity — int

- m 🔒 KWArrayList()
- m 🔒 KWArrayList(int)

- m 🔒 add(E) — boolean
- m 🔒 add(int, E) — void
- m 🔒 get(int) — E
- m 🔒 set(int, E) — E
- m 🔒 remove(int) — E
- m 🔒 remove(E) — boolean
- m 🔒 reallocate() — void
- m 🔒 indexOf(E) — int
- m 🔒 size() — int
- m 🔒 moveMiddleToEnd() void

Node
- f 🔒 data — KWArrayList<E>
- f 🔒 next — Node<E>
- f 🔒 prev — Node<E>
- f 🔒 capacity — int

- m 🔒 Node()
- m 🔒 Node(E)
- m 🔒 Node(KWArrayList<E>, int)
- m 🔒 Node(KWArrayList<E>, int, Node<E>, Node<E>)

- m 🔒 add(E) — void
- m 🔒 remove(int) — E
- m 🔒 searchAndGetIndex(E) — int
- m 🔒 get(int) — E
- m 🔒 set(int, E) — void
- m 🔒 checkBorder(int) — boolean
- m 🔒 equals(Object) — boolean

- p 🔒 _full — boolean
- p 🔒 size — int

HybridListIterator
- f 🔒 nextItem — Node<E>
- f 🔒 lastItem — Node<E>
- f 🔒 index — int
- f 🔒 flag — int
- f 🔒 elementIndex — int
- f 🔒 nextElement — E
- f 🔒 arrayIndex — int

- m 🔒 HybridListIterator(int)

- m 🔒 add(E) — void
- m 🔒 hasNext() — boolean
- m 🔒 hasPrevious() — boolean
- m 🔒 next() — E
- m 🔒 nextIndex() — int
- m 🔒 previous() — E
- m 🔒 previousIndex() — int
- m 🔒 remove() — void
- m 🔒 set(E) — void
- m 🔒 get() — E

Node
- f 🔒 data — E
- f 🔒 next — Node<E>
- f 🔒 prev — Node<E>

- m 🔒 Node(E)
- m 🔒 Node(E, Node<E>)

KWListIter
- f 🔒 nextItem — Node<E>
- f 🔒 lastItemReturned Node<E>
- f 🔒 index — int

- m 🔒 KWListIter(int)

- m 🔒 hasNext() — boolean
- m 🔒 next() — E
- m 🔒 hasPrevious() — boolean
- m 🔒 previous() — E
- m 🔒 add(E) — void
- m 🔒 nextIndex() — int
- m 🔒 previousIndex() — int
- m 🔒 remove() — void
- m 🔒 set(E) — void

officeCabinetModel
- MODEL1
- MODEL2
- MODEL3
- MODEL4
- MODEL5
- MODEL6
- MODEL7
- MODEL8
- MODEL9
- MODEL10
- MODEL11
- MODEL12

bookcaseModel
- MODEL1
- MODEL2
- MODEL3
- MODEL4
- MODEL5
- MODEL6
- MODEL7
- MODEL8
- MODEL9
- MODEL10
- MODEL11
- MODEL12

branchEmployee
- m 🔒 branchEmployee(branch)

- m 🔒 addUser(String, String, String, int) — boolean
- m 🔒 removeUser(String, String, String, int) — boolean
- m 🔒 enterShipmentInformation(officeChairModel, officeChairColor) boolean
- m 🔒 enterShipmentInformation(officeDeskModel, officeDeskColor) boolean
- m 🔒 enterShipmentInformation(meetableModel, meetableColor) boolean
- m 🔒 enterShipmentInformation(bookcaseModel) boolean
- m 🔒 enterShipmentInformation(officeCabinetModel) boolean
- m 🔒 removeShipmentInformation(officeChairModel, officeChairColor) void

- p 🔒 branch — branch

meetableModel
- MODEL1
- MODEL2
- MODEL3
- MODEL4
- MODEL5
- MODEL6
- MODEL7
- MODEL8
- MODEL9
- MODEL10

branch
- m 🔒 branch(Company, int)

- p 🔒 company — Company
- p 🔒 officeCabinet — HybridList<officeCabinet>
- p 🔒 branchCode — int
- p 🔒 meetable — HybridList<meetable>
- p 🔒 bookcase — HybridList<bookcase>
- p 🔒 officeChair — HybridList<officeChair>
- p 🔒 officeDesk — HybridList<officeDesk>
- p 🔒 branchEmployee KWLinkedList<branchEmployee>

officeDesk
- f 🔒 officeDeskModel — officeDeskModel

- m 🔒 officeDesk(officeDeskModel, officeDeskColor)

- m 🔒 setofficeDeskColor(officeDeskColor) — void
- m 🔒 getofficeDeskModel() — officeDeskModel
- m 🔒 setofficeDeskModel(officeDeskModel) — void

- p 🔒 officeDeskColor — officeDeskColor

meetable
- f 🔒 meetableModel — meetableModel

- m 🔒 meetable(meetableModel, meetableColor)

- m 🔒 setmeetableColor(meetableColor) — void
- m 🔒 getmeetableModel() — meetableModel
- m 🔒 setmeetableModel(meetableModel) — void

- p 🔒 meetableColor — meetableColor

administrator
- m 🔒 administrator(Company)

- m 🔒 addBranch(int) — boolean
- m 🔒 removeBranch(int) — boolean
- m 🔒 addBranchEmployee(int) — boolean
- m 🔒 removeBranchEmployee(int, String, String, int) boolean

- p 🔒 company — Company

officeChairModel
- MODEL1
- MODEL2
- MODEL3
- MODEL4
- MODEL5
- MODEL6
- MODEL7

Company
- m 🔒 Company()

- m 🔒 login(String, int) — person

- p 🔒 branch — KWLinkedList<branch>
- p 🔒 administrator — administrator
- p 🔒 customer — KWArrayList<person>

officeChair
- m 🔒 officeChair(officeChairModel, officeChairColor)

- m 🔒 clear() — void

- p 🔒 officeChairModel — officeChairModel
- p 🔒 officeChairColor — officeChairColor

officeChairColor
- color1
- color2
- color3
- color4
- color5

officeDeskModel
- MODEL1
- MODEL2
- MODEL3
- MODEL4
- MODEL5

meetableColor
- color1
- color2
- color3
- color4
- color5

officeDeskColor
- color1
- color2
- color3
- color4

officeCabinet
- m 🔒 officeCabinet(officeCabinetModel)

- p 🔒 officeCabinetModel officeCabinetModel

bookcase
- m 🔒 bookcase(bookcaseModel)

- p 🔒 bookcaseModel bookcaseModel

main
- m 🔒 main(String[]) void

- 📁 Package groovyjarjarcommonscli
- 📁 Package toolbarButtonGraphics
- 📁 Package groovyjarjarantlr
- 📁 Package groovyjarjarasm
- 📁 Package netscape
- 📁 Package META-INF
- 📁 Package javadoc
- 📁 Package images
- 📁 Package groovy
- 📁 Package javax
- 📁 Package java
- 📁 Package com
- 📁 Package sun
- 📁 Package org
- 📁 Package jdk

Problem Solutions Approach

    The job to be done by the admin is different, the job to be done by the customer is different. The work to be done by the employee is different, so they should only do the work they will do by themselves.Administrator's job is a must if the customer does. Because of this, privacy and security are important

    The arraylist type variable is kept in the linked list. Therefore, the method of adding and deleting elements needs to be changed. The node class and the iterator class should be changed accordingly.

Test cases

```java
Company company=new Company();
administrator admin=new administrator(company);
company.setAdministrator(admin);
company.getAdministrator().addBranch( branchCode: 0);
company.getAdministrator().addBranch( branchCode: 1);        create 4 branch by
company.getAdministrator().addBranch( branchCode: 2);        Administrator
company.getAdministrator().addBranch( branchCode: 3);

company.getAdministrator().addBranchEmployee( branchCode: 0);    add branch employee at
company.getAdministrator().addBranchEmployee( branchCode: 0);    branchcode"0"
company.getAdministrator().addBranchEmployee( branchCode: 1);    add branch employee at  branch code "1"
 company.getAdministrator().addBranchEmployee( branchCode: 2);   add branch employee at
 company.getAdministrator().addBranchEmployee( branchCode: 2);   branch code "2"
 company.getAdministrator().addBranchEmployee( branchCode: 2);
 company.getAdministrator().addBranchEmployee( branchCode: 3);   add branch employee at
 company.getAdministrator().addBranchEmployee( branchCode: 3);   branch code "3"
```

```java
company.getBranch().get(0).getBranchEmployee().get(0).enterShipmentInformation(officeChairModel.MODEL1,officeChairColor.color1);
company.getBranch().get(0).getBranchEmployee().get(0).enterShipmentInformation(officeDeskModel.MODEL2,officeDeskColor.color2);
company.getBranch().get(0).getBranchEmployee().get(1).enterShipmentInformation(meetableModel.MODEL1,meetableColor.color4);
company.getBranch().get(0).getBranchEmployee().get(1).enterShipmentInformation(bookcaseModel.MODEL1);

company.getBranch().get(1).getBranchEmployee().get(0).enterShipmentInformation(officeChairModel.MODEL1,officeChairColor.color1);
company.getBranch().get(1).getBranchEmployee().get(0).enterShipmentInformation(officeDeskModel.MODEL2,officeDeskColor.color2);
 company.getBranch().get(1).getBranchEmployee().get(0).enterShipmentInformation(meetableModel.MODEL1,meetableColor.color4);
 company.getBranch().get(1).getBranchEmployee().get(0).enterShipmentInformation(bookcaseModel.MODEL1);
```

At BranchCode "0" by BranchEmployee "0" "1"
officeChairModel.MODEL1,officeChairColor.color1
officeDeskModel.MODEL2,officeDeskColor.color2
meetableModel.MODEL1,meetableColor.color4
bookcaseModel.MODEL1

At Branch code "1"   by BranchEmployee "0"
officeChairModel.MODEL1,officeChairColor.color1
officeDeskModel.MODEL2,officeDeskColor.color2
meetableModel.MODEL1,meetableColor.color4
bookcaseModel.MODEL1

```java
company.getBranch().get(2).getBranchEmployee().get(0).enterShipmentInformation(officeChairModel.MODEL1,officeChairColor.color1);
company.getBranch().get(2).getBranchEmployee().get(0).enterShipmentInformation(officeDeskModel.MODEL2,officeDeskColor.color2);
company.getBranch().get(2).getBranchEmployee().get(1).enterShipmentInformation(meetableModel.MODEL1,meetableColor.color4);
company.getBranch().get(2).getBranchEmployee().get(1).enterShipmentInformation(bookcaseModel.MODEL1);

company.getBranch().get(3).getBranchEmployee().get(0).enterShipmentInformation(officeChairModel.MODEL1,officeChairColor.color1)
company.getBranch().get(3).getBranchEmployee().get(0).enterShipmentInformation(officeDeskModel.MODEL2,officeDeskColor.color2);
company.getBranch().get(3).getBranchEmployee().get(1).enterShipmentInformation(meetableModel.MODEL1,meetableColor.color4);
company.getBranch().get(3).getBranchEmployee().get(1).enterShipmentInformation(bookcaseModel.MODEL1);
```

ADD PRODUCT

```java
company.getBranch().get(0).getBranchEmployee().get(0).addUser( name: "orkun", surname: "arslan", email: "roa@gmail.com", password: 5858552);
company.getBranch().get(0).getBranchEmployee().get(0).addUser( name: "refik", surname: "arslan", email: "roa_05@gmail.com", password: 525254);
company.getBranch().get(2).getBranchEmployee().get(1).addUser( name: "re", surname: "arsn", email: "r_05@gmail.com", password: 528254);
```

ADD USER

```java
company.login( email: "roa@gmail.com", password: 5858552).searchProduct(company.getBranch(),officeChairModel.MODEL1,officeChairColor.color1);
```

LOGİN AND SEARCHProduct

```java
company.login( email: "roa@gmail.com", password: 5858552).buy(company.getBranch(), Adres: "Adres", phone: 5355555,officeChairModel.MODEL1,officeChairColor.color1
```

Login and but product

Running command and results

```
Found it
Branch :0
Found it
Branch :0
Buy it
```

search product output
found it and print branch code

buy product output
if product avaiable,print buy it