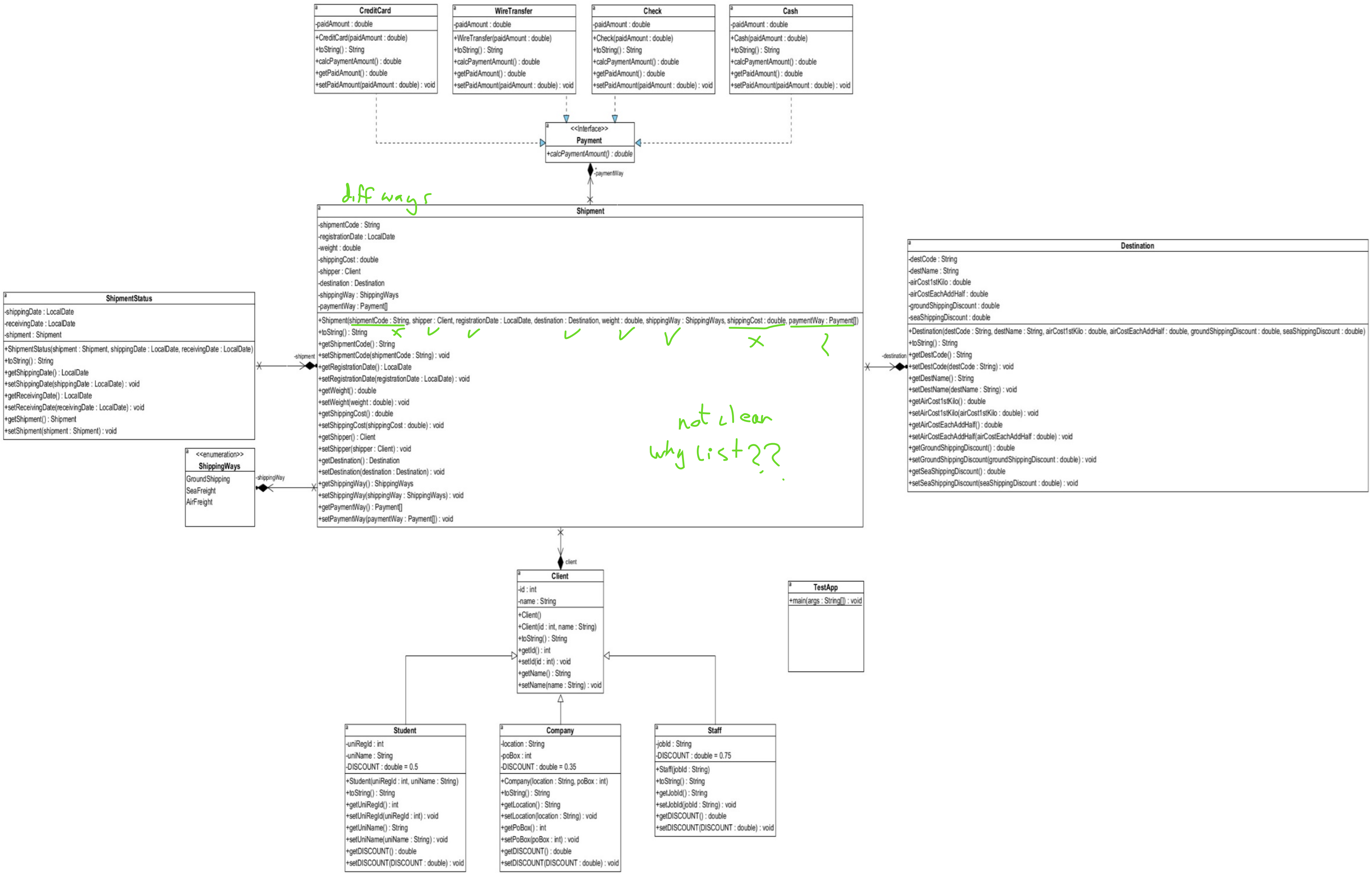
CreditCard
-paidAmount : double
+CreditCard(paidAmount : double)
+toString() : String
+calcPaymentAmount() : double
+getPaidAmount() : double
+setPaidAmount(paidAmount : double) : void
WireTransfer
-paidAmount : double
+WireTransfer(paidAmount : double)
+toString() : String
+calcPaymentAmount() : double
+getPaidAmount() : double
+setPaidAmount(paidAmount : double) : void
Check
-paidAmount : double
+Check(paidAmount : double)
+toString() : String
+calcPaymentAmount() : double
+getPaidAmount() : double
+setPaidAmount(paidAmount : double) : void
Cash
-paidAmount : double
+Cash(paidAmount : double)
+toString() : String
+calcPaymentAmount() : double
+getPaidAmount() : double
+setPaidAmount(paidAmount : double) : void
<<Interface>>
Payment
+calcPaymentAmount() : double
*
-paymentWay
Shipment
-shipmentCode : String
-registrationDate : LocalDate
-weight : double
-shippingCost : double
-shipper : Client
-destination : Destination
-shippingWay : ShippingWays
-paymentWay : Payment[]
+Shipment(shipmentCode : String, shipper : Client, registrationDate : LocalDate, destination : Destination, weight : double, shippingWay : ShippingWays, shippingCost : double, paymentWay : Payment[])
+toString() : String
+getShipmentCode() : String
+setShipmentCode(shipmentCode : String) : void
+getRegistrationDate() : LocalDate
+setRegistrationDate(registrationDate : LocalDate) : void
+getWeight() : double
+setWeight(weight : double) : void
+getShippingCost() : double
+setShippingCost(shippingCost : double) : void
+getShipper() : Client
+setShipper(shipper : Client) : void
+getDestination() : Destination
+setDestination(destination : Destination) : void
+getShippingWay() : ShippingWays
+setShippingWay(shippingWay : ShippingWays) : void
+getPaymentWay() : Payment[]
+setPaymentWay(paymentWay : Payment[]) : void
ShipmentStatus
-shippingDate : LocalDate
-receivingDate : LocalDate
-shipment : Shipment
+ShipmentStatus(shipment : Shipment, shippingDate : LocalDate, receivingDate : LocalDate)
+toString() : String
+getShippingDate() : LocalDate
+setShippingDate(shippingDate : LocalDate) : void
+getReceivingDate() : LocalDate
+setReceivingDate(receivingDate : LocalDate) : void
+getShipment() : Shipment
+setShipment(shipment : Shipment) : void
-shipment
<<enumeration>>
ShippingWays
GroundShipping
SeaFreight
AirFreight
-shippingWay
-destination
Destination
-destCode : String
-destName : String
-airCost1stKilo : double
-airCostEachAddHalf : double
-groundShippingDiscount : double
-seaShippingDiscount : double
+Destination(destCode : String, destName : String, airCost1stKilo : double, airCostEachAddHalf : double, groundShippingDiscount : double, seaShippingDiscount : double)
+toString() : String
+getDestCode() : String
+setDestCode(destCode : String) : void
+getDestName() : String
+setDestName(destName : String) : void
+getAirCost1stKilo() : double
+setAirCost1stKilo(airCost1stKilo : double) : void
+getAirCostEachAddHalf() : double
+setAirCostEachAddHalf(airCostEachAddHalf : double) : void
+getGroundShippingDiscount() : double
+setGroundShippingDiscount(groundShippingDiscount : double) : void
+getSeaShippingDiscount() : double
+setSeaShippingDiscount(seaShippingDiscount : double) : void
client
Client
-id : int
-name : String
+Client()
+Client(id : int, name : String)
+toString() : String
+getId() : int
+setId(id : int) : void
+getName() : String
+setName(name : String) : void
TestApp
+main(args : String[]) : void
Student
-uniRegId : int
-uniName : String
-DISCOUNT : double = 0.5
+Student(uniRegId : int, uniName : String)
+toString() : String
+getUniRegId() : int
+setUniRegId(uniRegId : int) : void
+getUniName() : String
+setUniName(uniName : String) : void
+getDISCOUNT() : double
+setDISCOUNT(DISCOUNT : double) : void
Company
-location : String
-poBox : int
-DISCOUNT : double = 0.35
+Company(location : String, poBox : int)
+toString() : String
+getLocation() : String
+setLocation(location : String) : void
+getPoBox() : int
+setPoBox(poBox : int) : void
+getDISCOUNT() : double
+setDISCOUNT(DISCOUNT : double) : void
Staff
-jobId : String
-DISCOUNT : double = 0.75
+Staff(jobId : String)
+toString() : String
+getJobId() : String
+setJobId(jobId : String) : void
+getDISCOUNT() : double
+setDISCOUNT(DISCOUNT : double) : void

diff ways

not clear
why list??

CMPS251 Course Project
Spring2024

جامعة قطر
QATAR UNIVERSITY

Shipping Company

1- Implement the UML class diagram illustrated in *Figure01*.

2- Print the next menu once the program executed:

Choose one of the next options:

1. Add Client ➔ As per the constrains below
2. Add Destination ➔ As per the constrains below
3. Create new shipment ➔ As per the constrains below
4. Update an existing shipment ➔ As per the constrains below
5. Sending/Receiving shipment ➔ As per the constrains below
6. Report1 – List the existing clients ➔ As per the constrains below
7. Report2 – List the destination details ➔ As per the constrains below
8. Report3 – List the shipment details ➔ As per the constrains below
9. Report4 – List the queued shipments ➔ As per the constrains below
10. Report5 – List the client shipments ➔ As per the constrains below
11. Report6 – List the income ➔ As per the constrains below
12. Report7 – List the shipments not yet collected ➔ As per the constrains below
13. Save and Exit ➔ As per the constrains below

3- Constrains of *Add Client*:

Allow the user to add four different types of clients; as for the client id, it has to be generated automatically, starting by 1001, and keep incrementing the last digit for each new client. Save the created clients in the **clients** Array List created in the class *TestApp*.

4- Constrains of *Add Destination*:

Destination code has to be generated automatically, starting by Des1001, and keep incrementing the last digit for each new destination. All the instance data members should be inserted for each instance, as for the *groundShippingDiscount* and *seaShippingDiscount*, both representing a percentage value.

5- Constrains of *Create new shipment*:

As for the shipment instance, give the user the chance to fill all the data members except the payment way, as follows:

a. *Shipment code*: to be generated automatically, starting by Ship1001, and keep incrementing the last digit for each new shipment.
b. *Registration date*: should be today's date, filled automatically,
c. *Shipment weight*: it has to be filled by the user.
d. *Shipment cost*: it has to be calculated automatically as follows:
   a. If the weight is below one kilo gram, round it up to one kilo gram.
   b. If the weight is above one kilo gram, round it up to nearest half.
   c. Calculate the shipping cost as per the two data members in the destination class, *airCost1stKil*, and *airCostEachAddHalf*.
e. *Shipper/Destination*: has to be one of the existing clients/destinations.
f. *Shipping way*: it has to be Air freight by default; in addition, give the user the chance to change it to Ground/Sea shipping. moreover, if the shipping way changed from Air to Ground or Sea, reflect this on the shipping cost by applying the discount giving as per the data members *groundShippingDiscount* and *seaShippingDiscount* in the destination class.
g. *Payment way*: The payment of the shipping cost could be:
   a. Postponed till later
   b. Paid in partial ← سؤال
   c. Paid in total

   Moreover, the total payments should not exceed the shipping cost. Consider adding the next additional fees for each payment way:

   ?

   a. *CreditCard payment*: Add 2.5% of the paid amount on the paid amount.
   b. *WireTransfer payment*: Add QAR 100 on the paid amount.
   c. *Check payment*: No additional fees will be applied for the check payment.
   d. *Cash payment*: No additional fees will be applied for the cash payment.

   need doc ussest

   not clear

h. Save the created *shipment* instance in the **shipments** Array List created in the class TestApp.
i. Once the shipping cost paid in 100%, create an instance of the class *shipmentStatus* and initialize the *shipment* data member by the instance paid in 100% recently, in Q5.h; let the *shippingDate* and *receivingDate* null; moreover, save the created *shipmentStatus* instance in the **shipmentsStatus** Array List created in the class TestApp.

✓ ↗ In the main

6- Constrains of *Update an existing shipment*:

Allow the user to select any of the existing shipments by inserting the shipment code and listing the rest of the information, consider the next:

   a. Allow the user to change the shipment shipping way as long as the selected shipment has NO paid amount yet.
   b. Once the selected shipment has a paid amount and a pending amount as well; don't allow modifying the shipping way, and allow the user to only pay the pending amount in total/in partial using any of the payment ways.
   c. Once the selected shipment has NO more pending amount, don't allow any more modification in that instance.

7- Constrains of *Sending/Receiving shipment*:

Allow the user to send/receive any of the existing shipments if and only if the full amount of the shipping cost has been paid. Allow the user to do the following:

a. Select a specific shipment using the shipment code, if the shipping cost of the selected shipment is not yet paid in full, display a message in this regard.
b. Assign a sending date, it has to be greater than or equal the shipment registration date.
c. Assign a receiving date, it has to be greater than the shipment sending date.
d. Allow sending and receiving the shipment in two different operations.

8- Constrains of *Report1 – List of all existing clients*:

Listing all the existing clients grouped by the Client type, such as the follows:

```
3 Client(s):

Client1 ID          Client1 name

Client2 ID          Client2 name

Client3 ID          Client3 name

1 Student(s):

Student1 ID         Student1 name

1 Company(s):

Company1 ID         Company1 name

1 Staff(s):

Staff1 ID           Staff1 name
```

9- Constrains of *Report2 – List the destination details*:

Get the destination code from the user, and list the destination information.

10- Constrains of *Report3 – List the shipment details*:

Get the shipment code from the user and list the shipment information, including the sending and receiving dates.

11- Constrains of *Report4 – List the queued shipments*:

Shipment has shipping cost been paid in full, with no sending date yet.

12- Constrains of *Report5 – List the client shipments* :

Get the client ID from the user, and list the client shipments for the selected user, as follows:

Ship. Code   Reg. Date   Weight   Cost   Dest. Name   Sending date   Receiving date

13- Constrains of *Report6 – List the income*:

List the overall income for the shipping company.

For example: The company overall income is QAR 555555.55

14- Constrains of *Report7 – List the shipments not yet collected*:

List the shipment code and sender name for shipments has been sent and not yet collected.

15- Constrains of *Save and Exit*:

Save all the changes done and allow to load these changes while running the program again.

Note1: For all the above reports, make sure to display the output in proper format using all the different types covered in this semester, such as: String.format(), printf(), \n, \t, %s, %d, and %f.

Note2: As for the amounts of money, make sure to display it with two digits after the decimal point.

Due date: Saturday May4th, by midnight

Good luck!