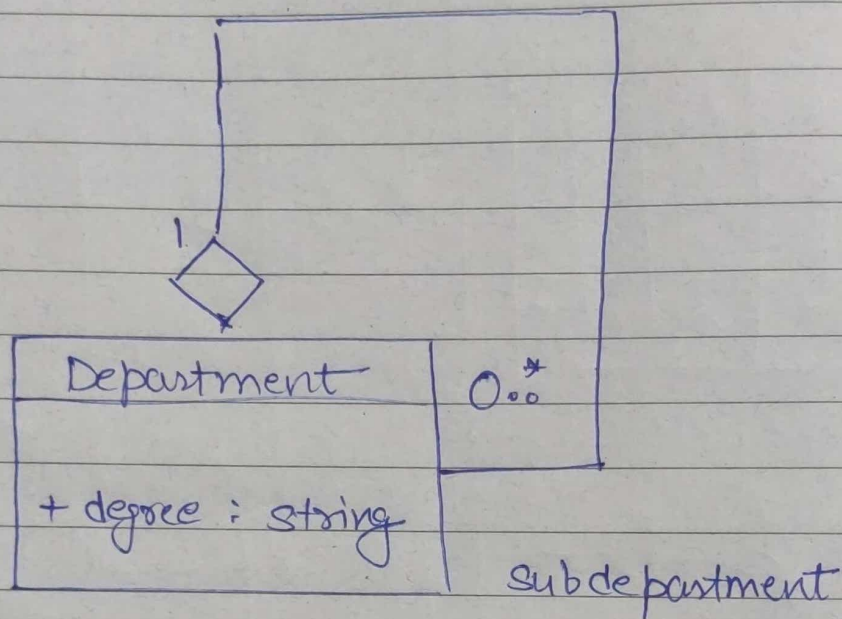Kartik Choudhary [B20CS025]

Software engineering (major exam paper)
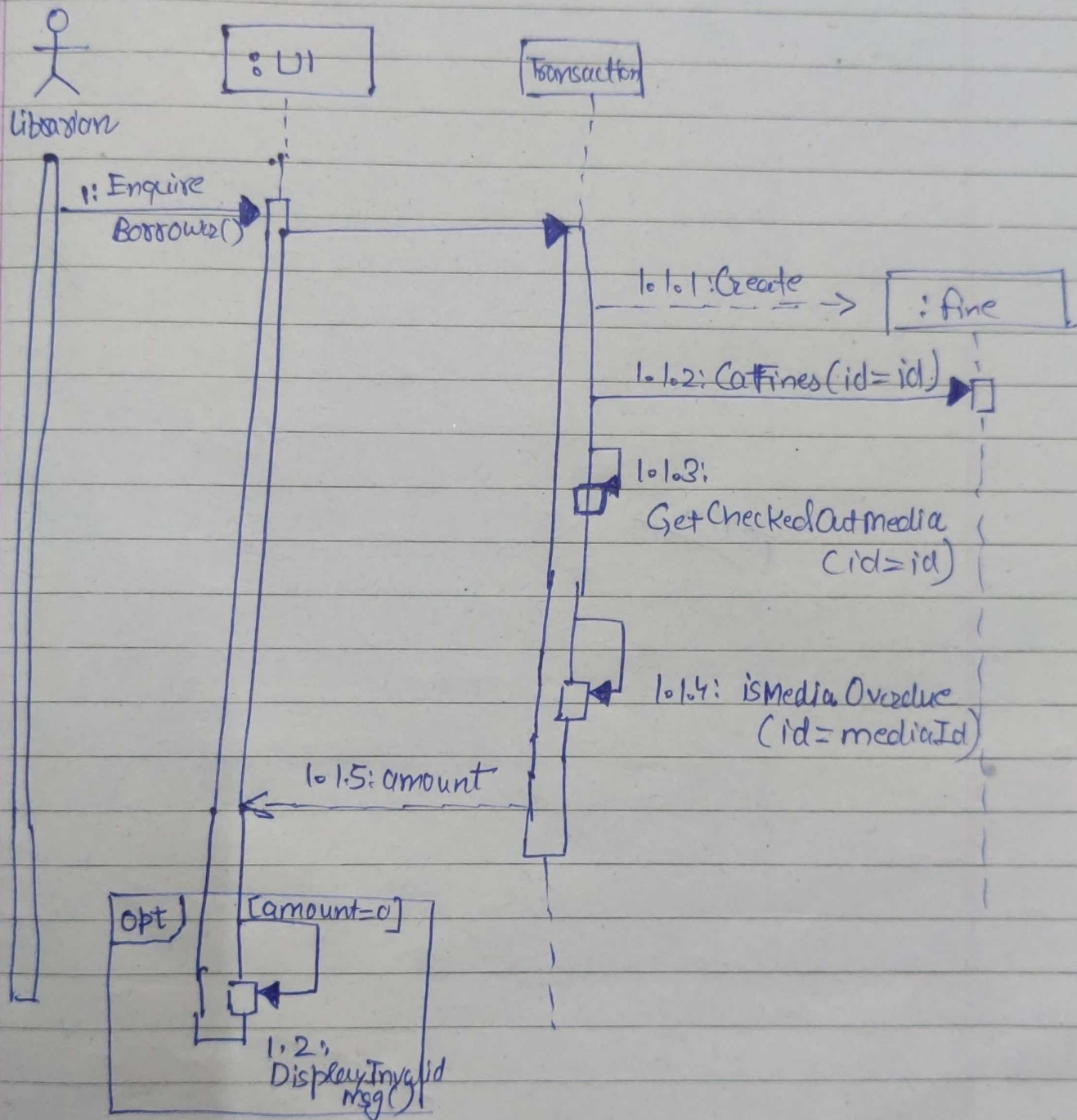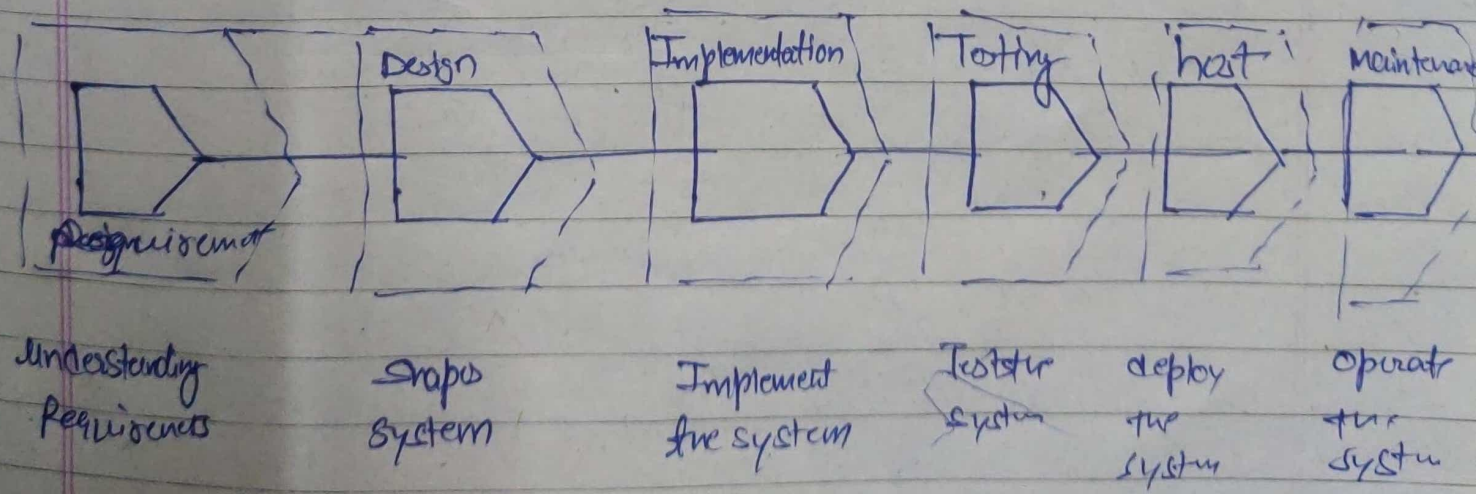
4) a) Object diagram

b)
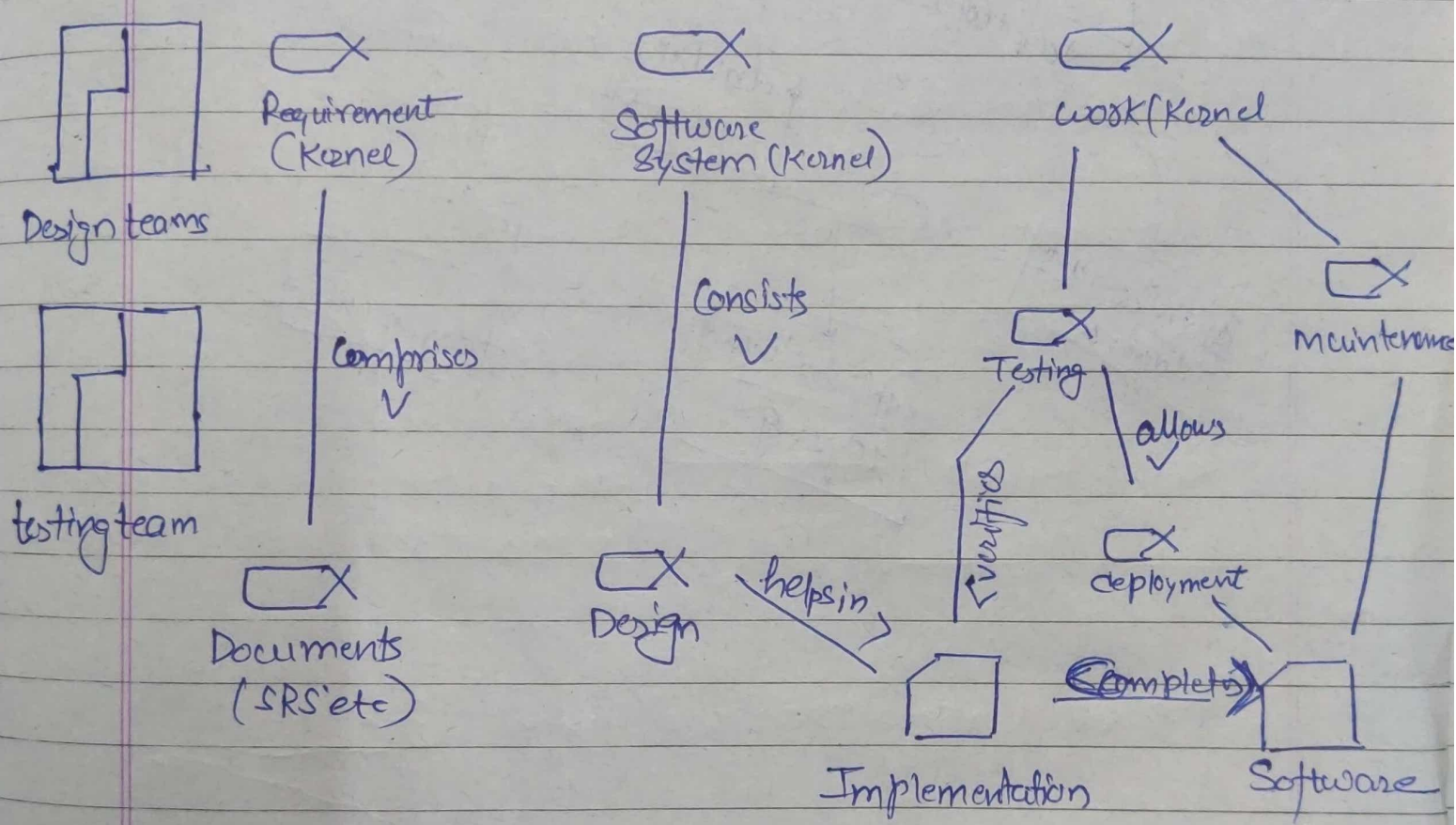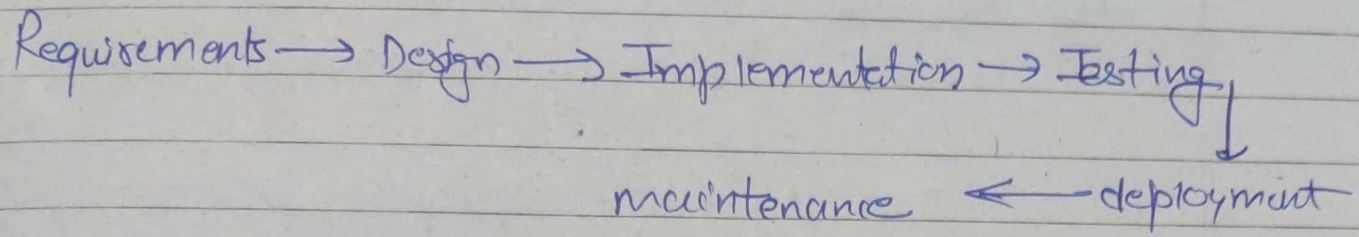

Class diagram

Since there is only one class with subclass (subdepartment) there will be a self loop

(c) Sequence diagram

Librarian

: UI

Transaction

1: Enquire Borrower()

1.1.1: Create → : Fine

1.1.2: CalFines(id=id)

1.1.3: GetCheckedOutmedia (id=id)

1.1.4: isMedia Overdue (id=mediaId)

1.1.5: amount

Opt [amount=0]

1.2: Display Invalid msg()

2) Waterfall is a predictive software model. Its flow is as follows:

Requirements → Design → Implementation → Testing

maintenance ← deployment

Design teams

Requirement (Kernel) — Comprises →

Software System (Kernel) — Consists →

Work (Kernel)

Maintenance

testing team

Documents (SRS etc)

Design — helps in →

Implementation

Testing ← verifies

allows

deployment — Completes →

Software

Requirement | Design | Implementation | Testing | host | Maintenance

Understanding Requirements | Shapes System | Implement the system | Test the System | deploy the system | Operate the system

(1.) Newsletter Pkg, Email Pkg, Contact Pkg and Survey Pkg
depend upon Lead Pkg for List <lead>
lead Pkg depends upon Util Pkg for the
Icommunicable interface.

a) Afferent Coupling $\boxed{C_a = 4}$

b) Stability of lead Pkg. $= \dfrac{C_a}{C_a + C_e} = \dfrac{4}{4+1} = 0.8$

(c) Unstable Packages :

Instability for Email Pkg $\Rightarrow C_a = 0$ (No package depends
on it)

$C_e = 3$ (It depends on
Acc Pkg, Contact Pkg,
lead Pkg)

$$J = \dfrac{3}{3+0} = 1$$

$\therefore$ Unstable Package

Newsletter Pkg

$C_a = 0$

$C_e = 3$ (Acc Pkg, Contact Pkg, lead Pkg)
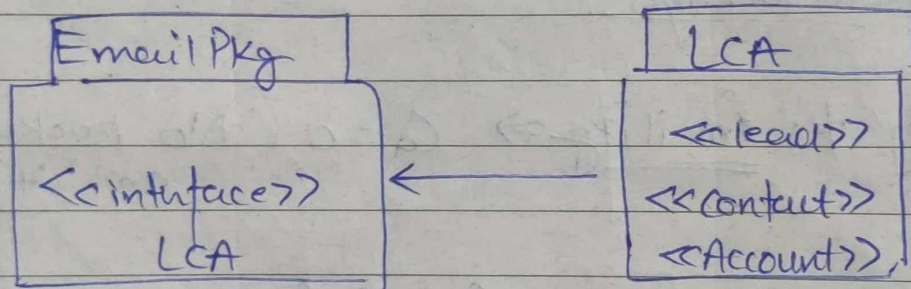
$J = \dfrac{3}{3+0} = 1 \Rightarrow$ Unstable Package

Survey Pkg

$$C_a = 8$$
$$C_e = 3 \quad (\text{Acc Pkg, Contact Pkg, Lead Pkg})$$

$$I = \frac{3}{3+0} = 1 \quad (\text{unstable package})$$

(d) To reduce the value of efferent coupling Ce
We can create an interface for lead,
Contact and Account (similar to that of
I (Communicable)

Email Pkg	LCA
<<interface>> LCA	<<lead>> <<contact>> <<Account>>

The Ce is now reduced from 3 to 1

(3) a) Structural design pattern is used in the code as seen when the class IGeometry was used as a datatype while constructing the list

b) Design principle it is following is: Reusability as we don't have to repeat the code for Geometry

c) Test requirements: { list of objects To Accomodate is not empty, n. area() doesnot return NULL, n. area ≥ 0 }

© * let max (this.totalArea) = 1000.0
* list of objects to Accomodate → L

Input	description	Expected Output
① this.totalArea = 0 L = [n]	n.area () = 0.0	True (valid)
② this.totalArea = 5 L = (x, y, z)	x.area () = 0.0 y.area () = 7.90 z.area () = 4.10	false (valid)
③ this.totalArea = 999 L = (a,b,c,d,e)	a.area () = 0.0 b.area () = 0.0 c.area () = 9.01 d.area () = 11.17 e.area () = 1.00	True (valid)
④ this.totalArea = 500 L = (a,b)	a.area () = 1001.99 b.area () = 51.73	false (Invalid)