* GROUP 1 — ALGORITHM (DR. HWANG).

Q. For the recurrence formula from Merge Sort and Max-sub array,

$$T(n) = 2T(n/2) + n \quad \text{for } n > 1$$

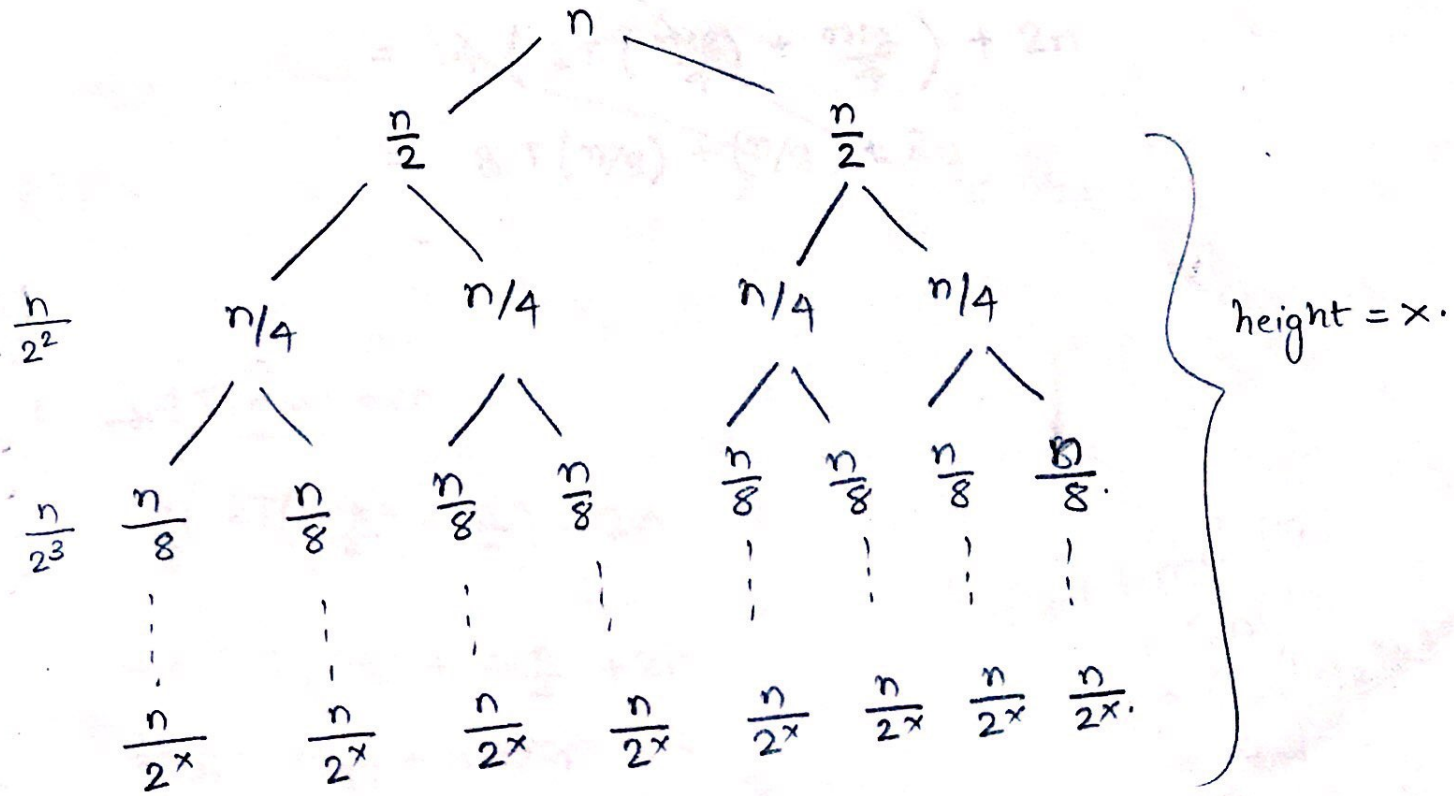
(a) Derive the total running time $T(n)$ by using Recursion Tree Method.

(b) Derive the total running time $T(n)$ by using Iteration method.

(a) T(n) by Recursion Tree method.:

$$T(n) = 2T(n/2) + n$$

Recursion Tree →



So, $\frac{n}{2^x} \cong 1 \Rightarrow n \cong 2^x$

i.e. $n = 2^x$ (Take log on both sides).

$\log n = \log 2^x$

$\log n = x \log 2$

$x = \frac{\log n}{\log 2} \rightarrow x = \log_2 n.$

Scanned with CamScanner

Total runing time $(T(n)) = Cn(x)$

$= Cn(\lg_2 n)$

$$\boxed{T(n) = \Theta(n \lg n).}$$

Scanned with CamScanner

Q1

(b) T(n) by Iteration method:

$$T(n) = 2T(n/2) + n. \quad - (1)$$

→ put

So, $\boxed{T(n/2) = 2T(n/4) + n/2}$ - put in 1.

$$= 2(2T(n/4) + n/2) + n$$

$$= 4T(n/4) + n + n$$

$$= 4T(n/4) + 2n \quad - (a)$$

Put $\boxed{T(n/4) = 2T(n/2/4) + n/4}$ in (a)

$$= 4\left(2T\left(\frac{n/2}{4}\right) + \frac{n/2}{4}\right) + 2n$$

$$= 8T(n/8) + (n/8) + 2n$$

$$=$$

$\rightarrow 4T(n/4) + 2n.$

$$4\left(2T\left(\frac{n/2}{4}\right) + \frac{n}{4}\right) + 2n.$$

$$= 4 \cdot 2\, T(n/8) + 4 \cdot \frac{n}{4} + 2n$$

$$= 8T(n/8) + n + 2n$$

$$= 8T(n/8) + 3n.$$

$$= 8T\left(\frac{n}{2^3}\right) + 3n.$$

$$= 8T\left(\frac{n}{2^i}\right) + 3n. \rightarrow$$

$$= \cancel{c\, T(n/2^i) + 3n}$$

Since $\frac{n}{2^i} = 1$, $n = 2^i$

$\therefore i = \log_2 n$

Total time (Tn).

$$= c_n(i)$$

$$= c_n(\log_2 n)$$

$$= c_n \lg n$$

$$\boxed{T(n) = \Theta(n \lg n)}$$

Scanned with CamScanner

$$T(n) = 2T(n/2) + n \quad \text{— (1)}$$

Put $\boxed{T(n/2) = 2T(n/4) + n/2}$ in (1)

$$T(n) = 2(2T(n/4) + n/2) + n$$

$$= 4T(n/4) + n + n$$

$$= 4T(n/4) + 2n.$$

$\boxed{\text{Put } T(n/4) = (2T(n/8) + n/4)}$

$$= 4 \cdot (2T(n/8) + n/4) + 2n$$

$$= 8T(n/8) + n + 2n.$$

$$= 8T(n/8) + 3n.$$

$$\vdots$$

$$= 2^3 T\left(\frac{n}{2^3}\right) + 3n.$$

$$\vdots$$

$$= 2^i T\left(\frac{n}{2^i}\right) + 3n.$$

So, here, $\frac{n}{2^i} = 1$ [i.e. 1 subproblem to be solved].

So $n = 2^i$

- Taking log on bothsides, we get

$$\log n = \log 2^i$$

$$\log n = i \log 2$$

$$i = \frac{\log n}{\log 2}$$

$$\boxed{i = \log_2 n}$$

So, $T(n) = Cn(i)$.

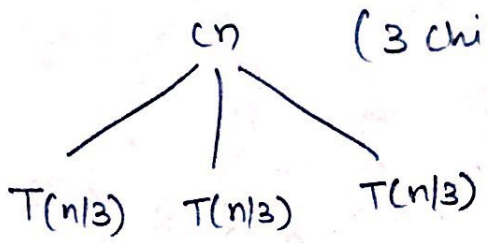
Scanned with CamScanner

* Recursion Tree Method.

Example ① $T(n) = 3T(n/3) + cn$ ← this is $f(n)$.

← in terms of this.

So,



(3 children).

Note: $T(n) = cn + T(n/3) + T(n/3) + T(n/3)$.

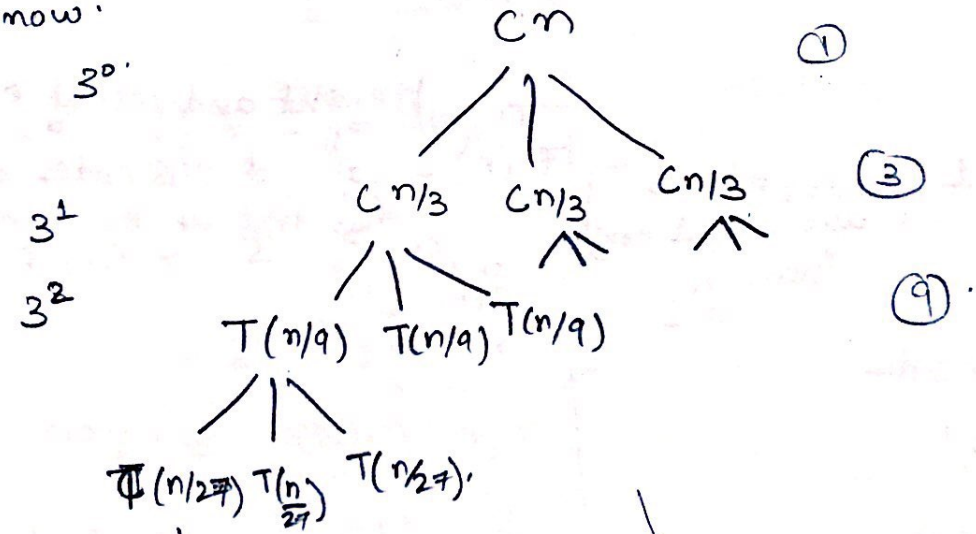
↓

expand these nodes further

So, $T(n/3) = 3T(n/3/3)$

$= 3T(n/9) + c\left(\frac{n}{3}\right)$

so now.

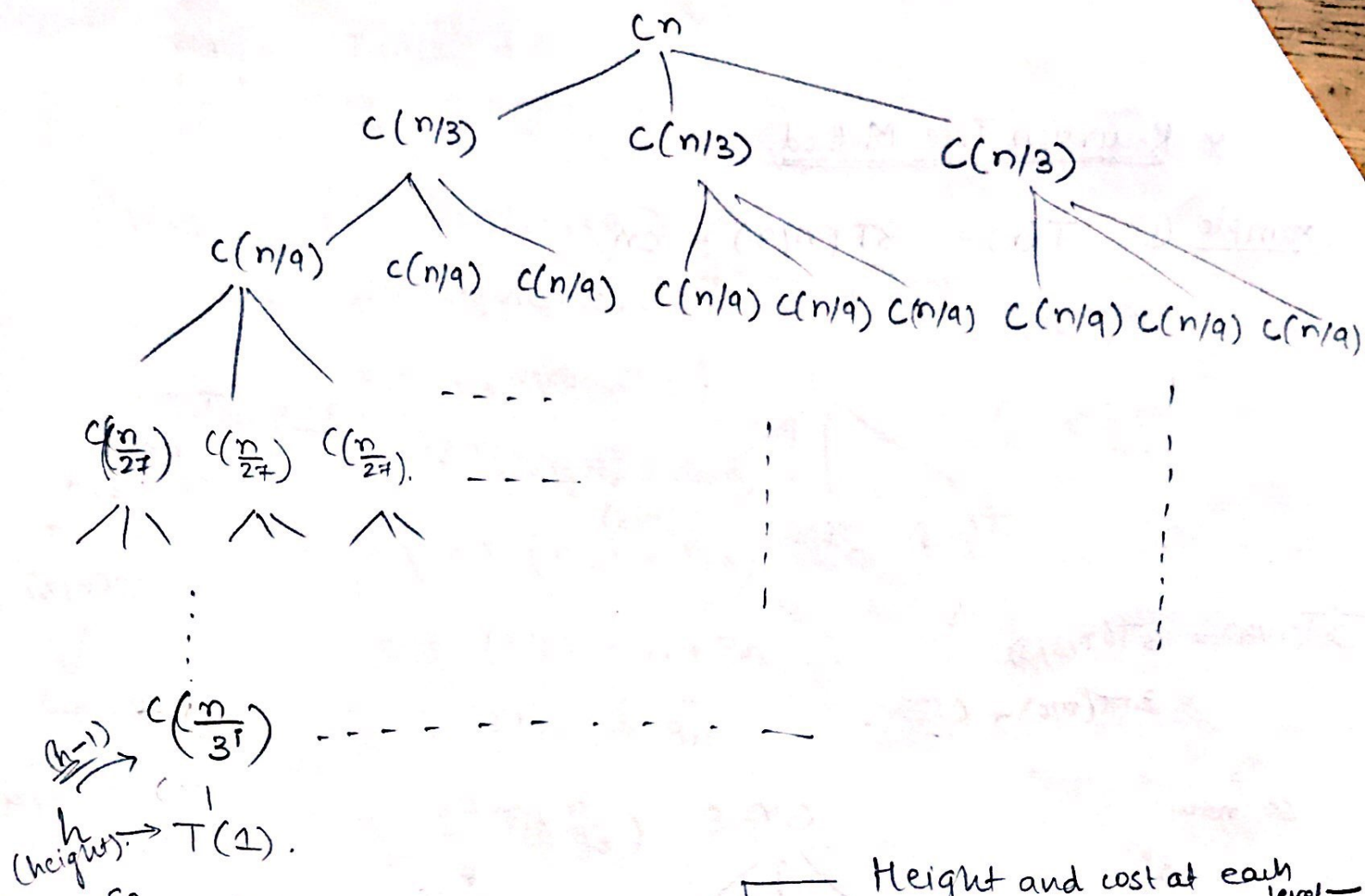


$T(n/9) = 3T(n/27) + c(n/9)$

This tree will look like (at back of the page)

leaf node → $c\left(\frac{n}{3^i}\right)$

Scanned with CamScanner



So, $c\left(\frac{n}{3^i}\right) = 1$ [subproblems are only 1 now from n to 1].

$n = 3^i$

Get log on both sides.

$\log n = \log 3^i$

$\log n = i \log 3$

$i = \frac{\log n}{\log 3}$

$\boxed{i = \log_3 n.}$

So. $T(n) = 3^i \times T(1) + cn + cn + cn \cdots\cdots$ (from Lev. 0 to h-1).

$= 3^h \times T(1) + \sum_{i=0}^{h} cn.$

(lower bound) as constant so, Θ

$= 3^{\log_3 n} T(1) + cn \log_3 n.$

$= n\,T(1) + cn \log_3 n \rightarrow \Theta(n \log n)$

Height and cost at each level

$T(n)$ = value of all nodes of the tree at each levels.

cost at level 1 $= cn$

lev. 2 $= 3\left(\frac{cn}{3}\right) = cn.$

$= 3^2\left(\frac{cn}{9}\right) = cn.$

$\vdots$

$3^i\left(\frac{cn}{3^i}\right) = cn.$

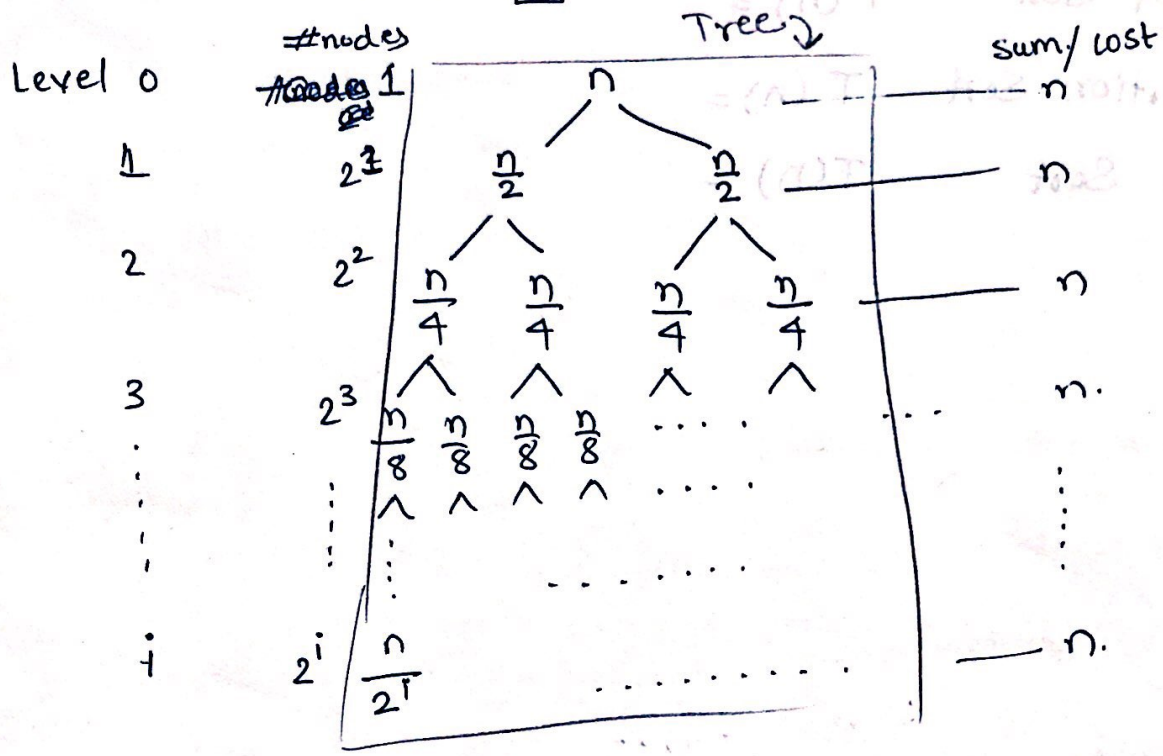
So, level sum will always be cn. (except for h (height).)

$T(1)$ will be some constant k.

Scanned with CamScanner

⊛. Recursion Tree Method: (Merge Sort).

$$T(n) = 2T(n/2) + \underbrace{n}$$



So, $\frac{n}{2^i} = 1 \quad i = \log_2 n.$

$$T(n) = 2^i \sum_{i=0}^{\log_2 n} n$$

$$= 2^{\log_2 n}(T(1)) + n\log_2 n$$

$$= \underbrace{n(T(1))}_{\text{some constant}} + n\log_2 n$$

$$T(n) = \Theta(n\log_2 n).$$

Scanned with CamScanner

GROUP 2 - SURVEY (DR. CHEN).

Q4. Explain the following software process models and point out their pros and cons:

(a) Formal Methods
(b) Agile development model.

(a) Formal Methods - mathematical specification of software systems.
- Has 2 steps:
  (i) Write requirements in formal language.
  (ii) Translate requirements into source code.
- Best suited for mission critical projects. Very high s/w quality (if this is used).

Pros:
- efficient than natural languages. (i.e. not ambiguous).
- can identify errors before they reach code.
- can be submitted to automatic code generators.

Cons:
- produces specifications that cannot be interpreted by customer.
- Requires extensive training to be properly used.
- It is time-consuming.
- usually can be used for mission critical projects.

(b) Agile Development Method - an incremental approach.
- Similar to incremental model, but has short term plans
- Uses small increments which are of about 2-4 weeks.
- Best suited for projects where requirements change rapidly during development process.

Pros:
- Emphasizes on customer collaboration over contract.
- Often interaction b/w customers and users.
- Respond to changes ~~rather than~~ faster.
- Stresses on working s/w rather than document.

Cons:
- team may not interact with other teams/members.
- maintaining simplicity needs extra efforts.
- Difficult for process-driven organization to change defined working model.

Scanned with CamScanner

* Software Development Process Models.

1. Code and Fix – adhoc. (for very short/startup projects).
2. Linear Sequential (Waterfall) ↓
   - Requirement Gathering & analysis
   - Design
   - Implementation
   - Testing
   - Maintenance.
3. Incremental Model
4. Evolutionary
   - (a) Prototyping model.
   - (b) Spiral Model.
5. Concurrent Development Model
6. Component Assembly model.
7. Formal Methods. (v.imp)
8. Agile development. (v.imp)

Scanned with CamScanner

GROUP 3 – DATABASE (DR. HWANG).

Fall 2018

(Q7) For the following real world universities problem including info. for Class, Course, Instructor, Student and Classroom.

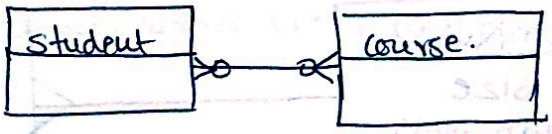
(a) Indicate all functional dependencies among all essential attributes including those you created.

(b) Draw an E/R diag. by creating essential attributes for all info. with pk's, fk's and relationships.

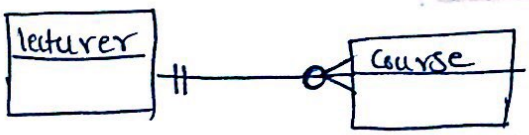
(*) CROW FOOT NOTATIONS.

- one-to-one

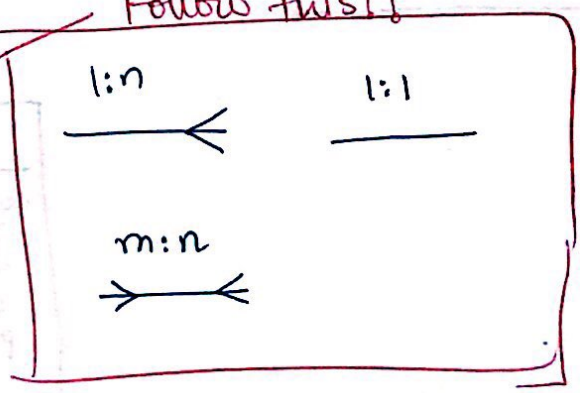
- many-to-many



- one-to-many

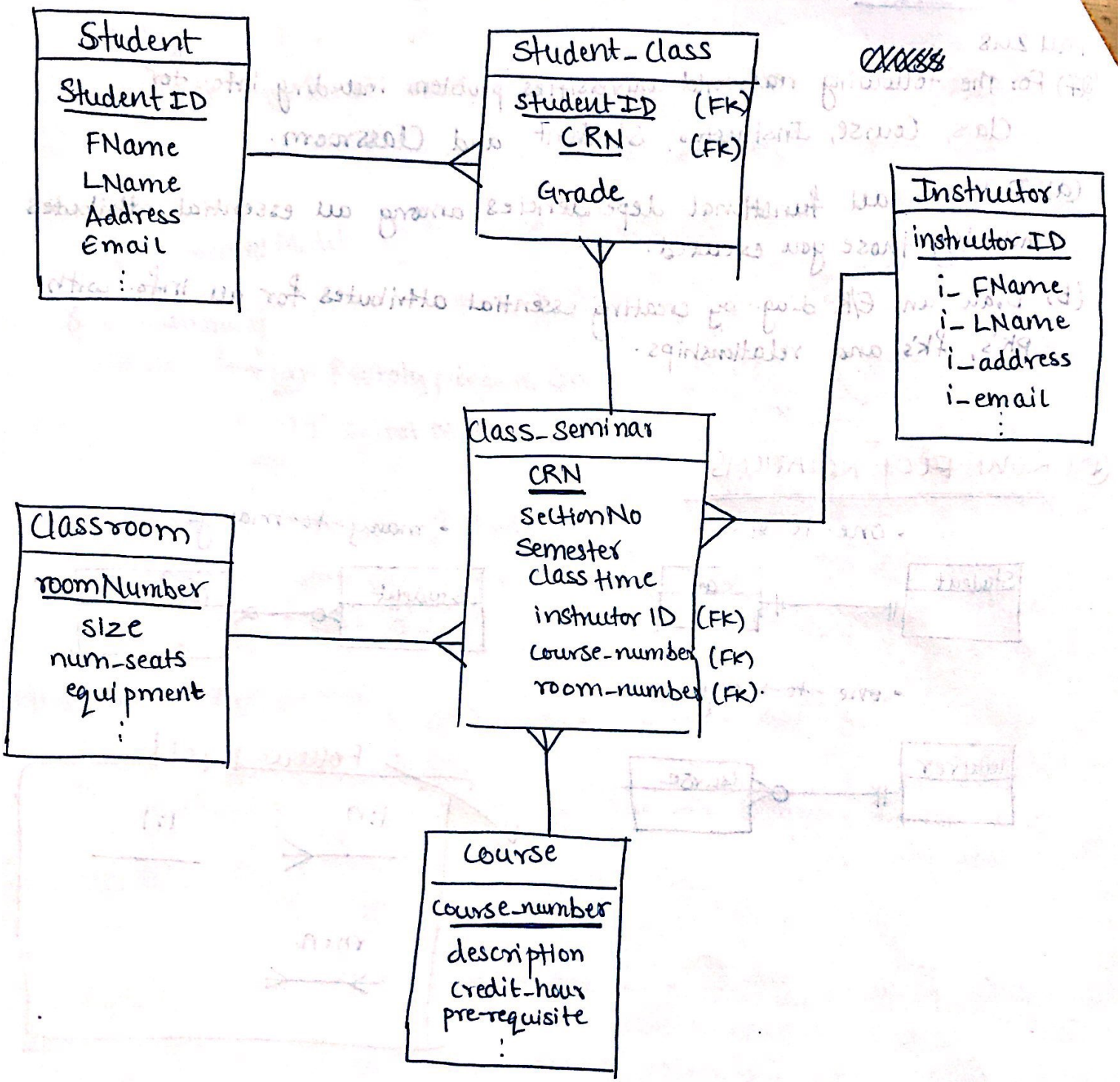


Follow this!!



Scanned with CamScanner

University / Enrollment ER



Scanned with CamScanner

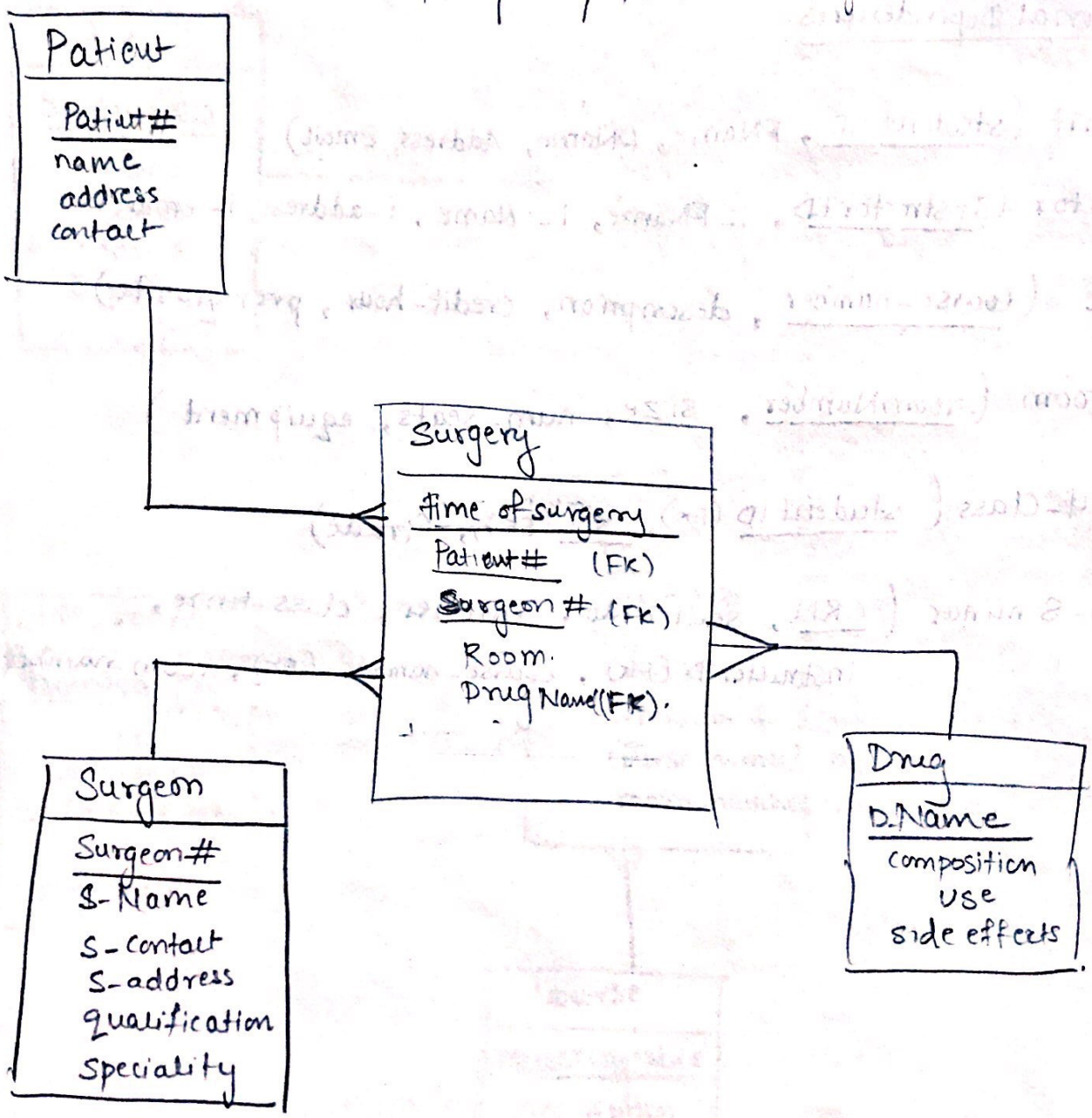
- Functional Dependencies

1. Student (StudentID, FName, LName, Address, Email)
2. Instructor (InstructorID, i_FName, i_LName, i-address, i-email)
3. Course (Course-number, description, credit-hour, prerequisite)
4. classroom (roomNumber, size, num-seats, equipment)
5. student-Class (studentID (FK), CRN (FK), Grade)
6. Class-Seminar (CRN, sectionNo, semester, class-time, instructorID (FK), course-number (FK), room-number (FK))

Scanned with CamScanner

ER

Hospital / Patient - Surgeon.



Scanned with CamScanner

GROUP 4 — FORMAL LANGUAGES (DR. GAO).

(or 10) Q11 For each, circle all correct statements. OR write "None" if all wrong.

(a) Correct statements are:

(a) A language can have infinite number of strings.

(b) Each string in a language must be finite.

(c) A string in a language must be non-empty.

(d) A language must be non-empty.

(b) Correct statements are:

(a) A DFSM must halt.

(b) A NDFSM must halt

(c) A NDFSM without ε-transitions must halt. (same as DFSM).

(c) correct statements:

(a) A DPDA must halt.

(b) A NDPDA must halt. → ND has ε transitions. So infinite loops (may not halt).

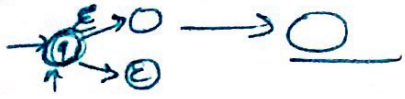
(c) A NDPDA without ε-transitions must halt. → with no ε-transition ≅ DPDA.

(d)

(a) Let M be DFSM, If ε ∈ L(M), start state of M must be an accepting state.

(b) Let M be DFSM, If the start state of M is an accepting state, then ε ∈ L(M).

(c) Let M be NDFSM. If ε ∈ L(M), then the start state of M must be an accepting state.



Scanned with CamScanner

? (d) Let M be NDFSM. If start state of M is an accepting state, then $\varepsilon \in L(M)$. — T

(e)

(a) Regular languages are closed under union.

(b) Regular languages are closed under intersection.

(c) Regular languanges are closed under complement.

(d) Regular languages are closed under set difference.

(f)

(a) Cmtext free languages are closed under union.

(b) " " " are closed under intersection.

(c) " " " " " " complement

(d) " " " " " " set difference.

(g)

DTM ≈ NDTM (no extra computability).

(a) A DTM must halt. DTM may not always halt

(b) A DTM must halt in atmost $|w|$ steps. $O(n) \times (\text{or } 0)$

(c) A NDTM must halt. $O(2^n)$

(d) A NDTM must halt in at most $|w|$ steps.

NONE

(h)

(a) In terms of computability, DFSM = NDFSM < DPDA < NDPDA < DTM = NDTM

(b) " " " " , DFSM = NDFSM < DPDA = NDPDA < DTM = NDTM.

(c) " " " " , DFSM < NDFSM < DPDA < NDPDA < DTM < NDTM.

Scanned with CamScanner

(i)
(a) Decidable languages are closed under union.
(b) " " " " " intersection.
(c) " " " " " complement.
(d) Semi-decidable languages are closed under complement.
SD not closed under complement.

(j)
(a) SAT is the first language proved to be in NP-complete. (T)
(b) 3-SAT is the first language proved to be in NP-complete.
(c) Both SAT and 3-SAT are NP-Hard.

L(M) in NP-complete
↳ ① NP
② NP-Hard.

SD → closed under:
① Union
② Intersection.

D → closed under
① complement
② Union
③ Intersection
④ set difference.

Regular → closed under
→ all these (4).

Context-free → closed under
① Union [only].

Scanned with CamScanner