Instructions
There are five questions in the test. You do not need to answer all the questions and may **choose any two questions to complete**. The test is 24h from when you receive it, and you may email galvin@lauretta.io if you have any questions.

Take care and good luck.

Problem 1 - Computer Vision

In this problem, you are given a set of JPEG puzzle pieces and you are required to write a program to solve (reconstruct)the puzzle. The coordinate of each piece is indicated by the number of red and blue dots in the piece. The number of red dots denotes the column index, while the number of blue dots denotes the row index of the puzzle. For example, a piece with 1 RED and 2 BLUE dots indicate that the piece is located at column 1 and row 2 of the puzzle (full image), as shown in Fig.1.

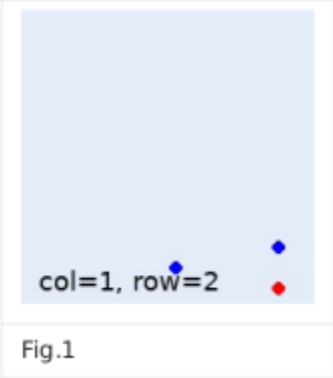


Fig.1

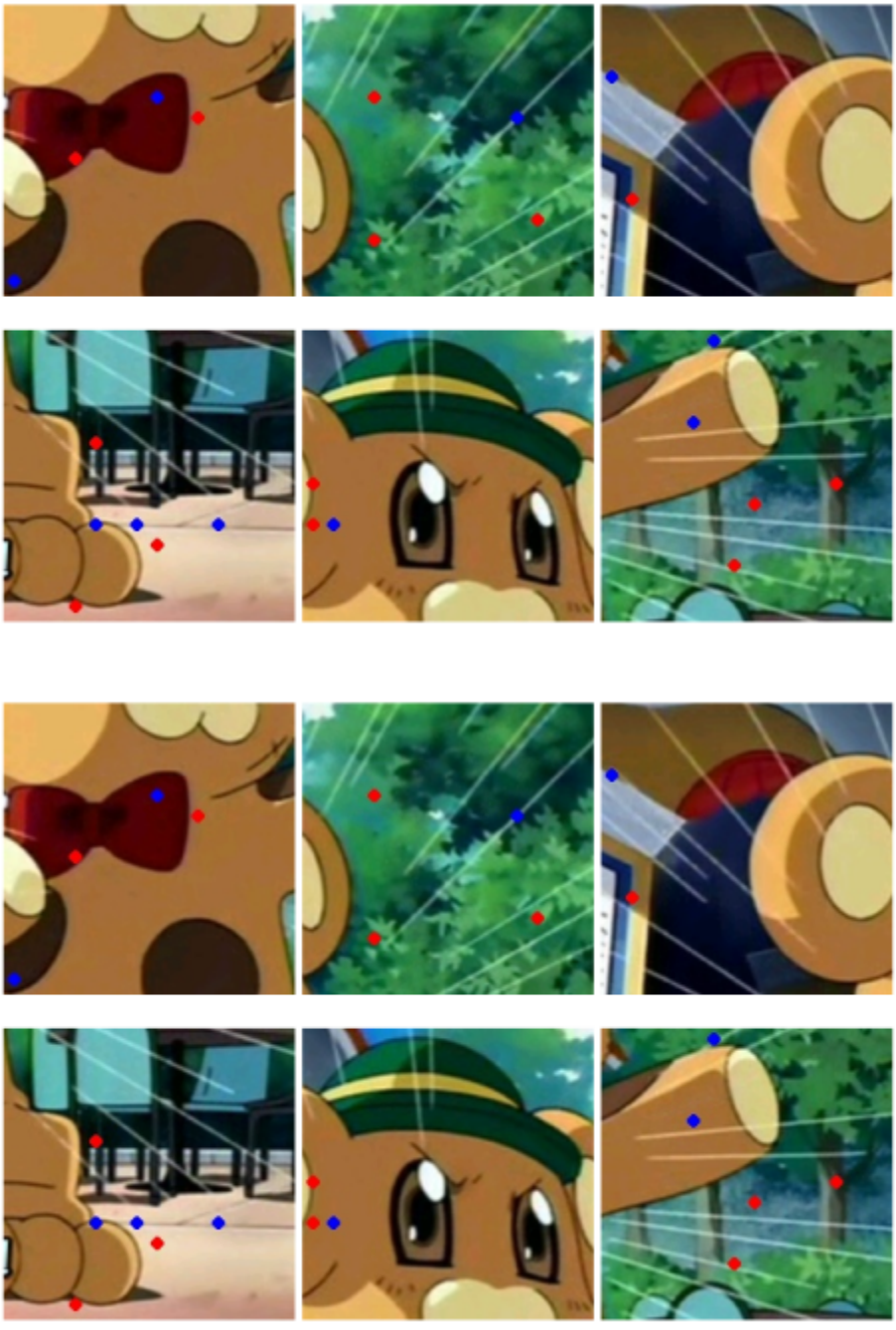
You are allowed to use any programming/scripting language + any computer vision library to complete this assignment.

Hints:
Color range of BLUE dot : RGB(0,0,200) ~ RGB(20,20,255)
Color range of RED dot : RGB(200,0,0) ~ RGB(255,20,20)

Sample

Input

From a the following images in "testdata/fumo"



Output

The reconstructed original image is expected to be something like this :

Problem 2 - Parallel Processing

In distributed transactions, we need to guarantee that concurrent updates on a replicated database are seen in the same order everywhere. This requires a totally-ordered multicast

– Update 1: add $100 to an account (initial value = $1000)
– Update 2: add 1% interest to account

– In absence of proper synchronization: replica 1 = $1111, replica 2 = $1110.

Lamport's logical clocks can be used to implement totally-ordered multicast in a completely distributed fashion.

Let's assume there are two processes communicating with each other. Each maintains its own logical clock, multicast events at random time intervals.

Write code that allows the execution of events in a total ordered fashion. Use a UDP socket for this communication, build your own data packet and ack structure. Each process will be acting as client and server at the same time. Use UDP socket to solve the above mentioned problem. You can use any programming language you want.

You can use any data structure you want. We recommend that you can use a dictionary to maintain the IP, PORT for each process, use this dictionary to send packets to every entry inside the dictionary. However, you can use some other data structure for this job.

Note: The output on each process console should only display the events ready for execution i.e. after acquiring all the acks

At Process P1
P1: Executing event <logical_clock_value, Src_Process=x>
P1: Executing event <logical_clock_value, Src_Process=x>

Similarly, at proces P2
P2: Executing event<logical_clock_value, Src_Process=x>
P2: Executing event<logical_clock_value, Src_Process=x>

Problem 3 - ER Diagrams

For the following ER diagrams, convert the conceptual design to relational design using the systematic methodology i.e. 9 step algorithm (**ER-to-Relational Mapping Algorithm** + **Mapping EER Model Constructs to Relations**). Map each part of ERD using the steps given. Write down each step of the mapping separately.

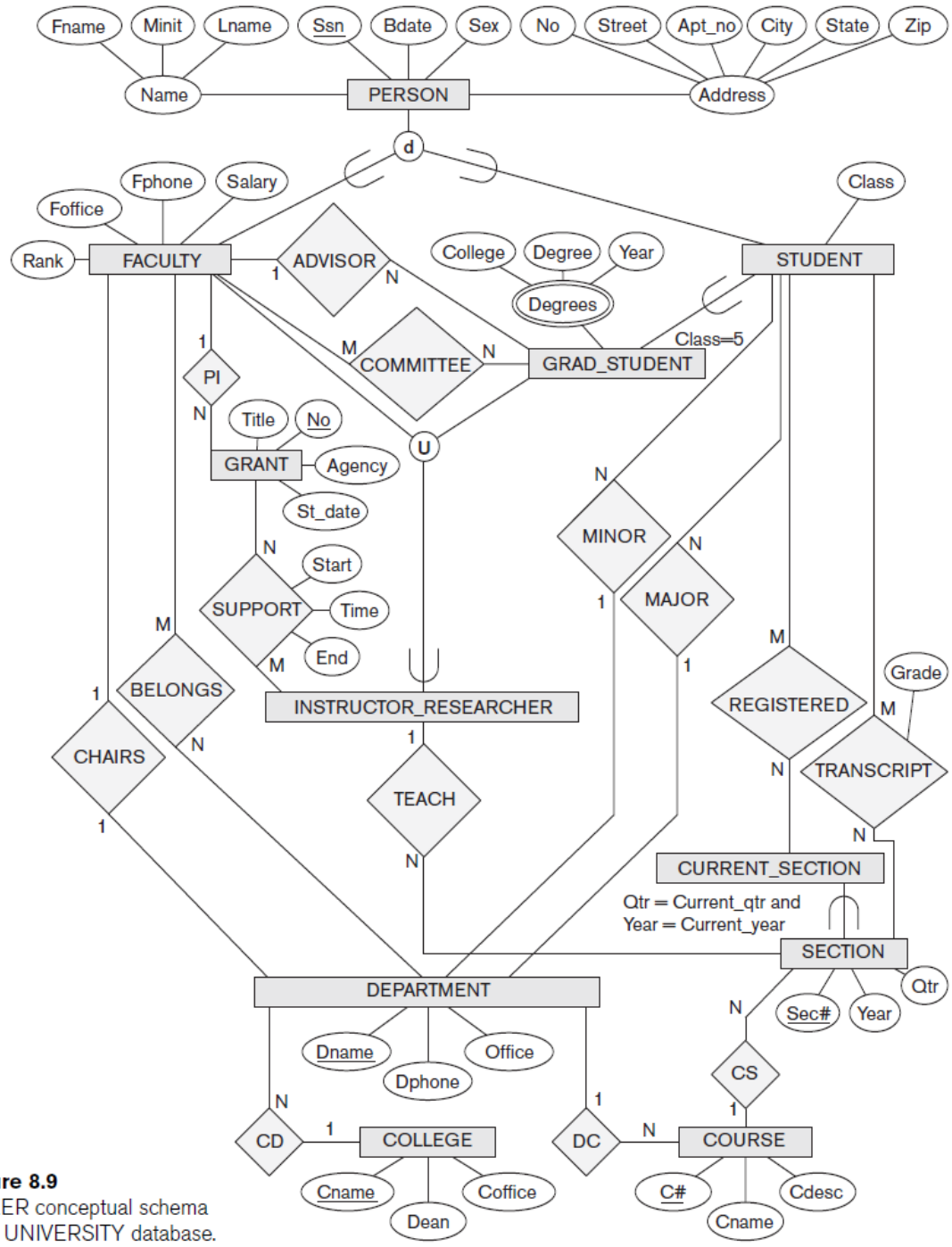
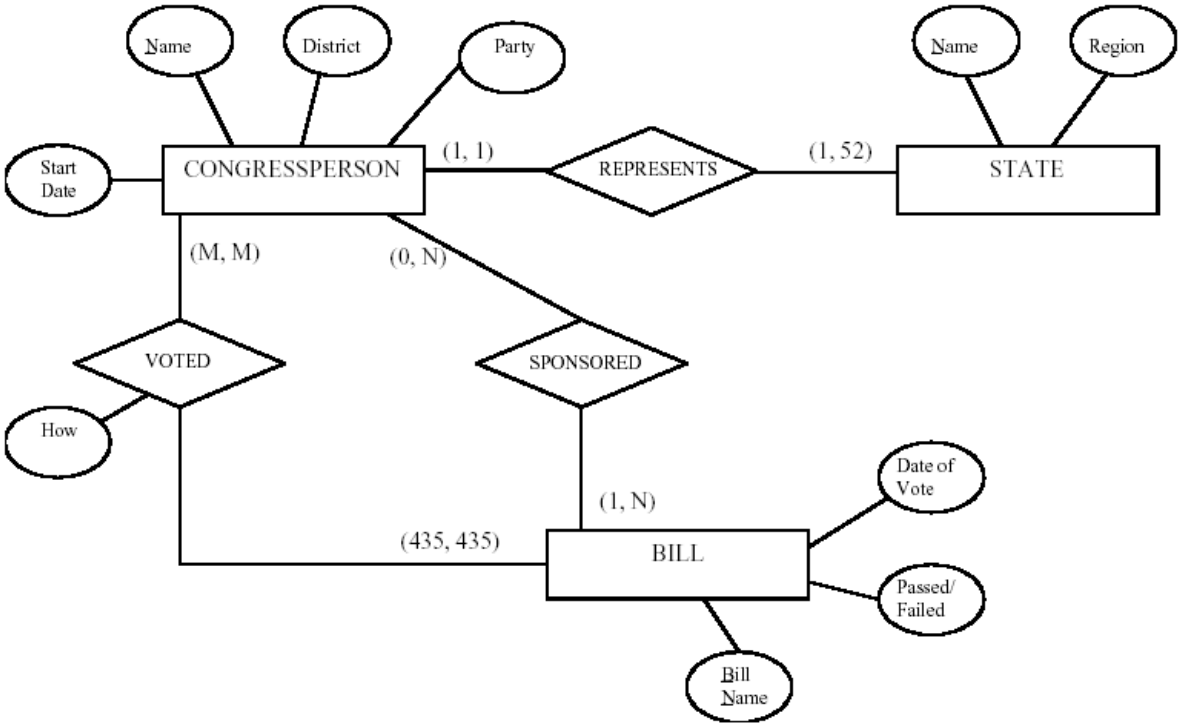
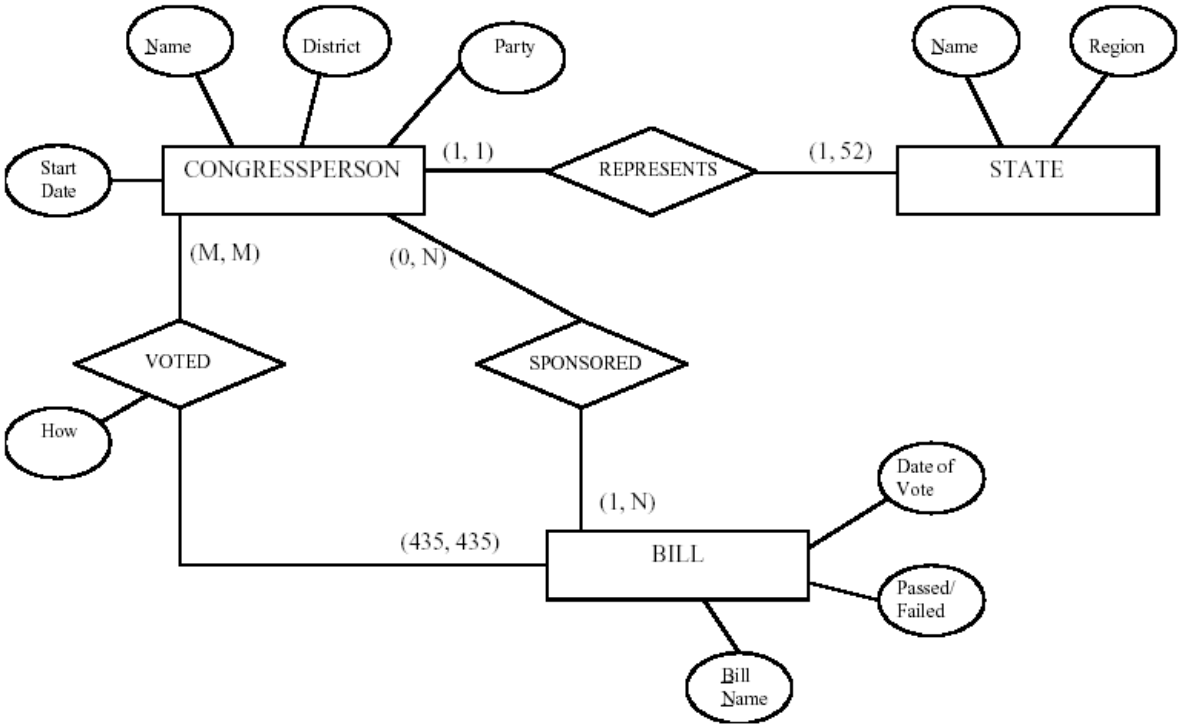


Figure 8.9
An EER conceptual schema for a UNIVERSITY database.


Name
District
Party
Name
Region
Start Date
CONGRESSPERSON
(1, 1)
REPRESENTS
(1, 52)
STATE
(M, M)
(0, N)
VOTED
SPONSORED
How
(1, N)
Date of Vote
(435, 435)
BILL
Passed/ Failed
Bill Name

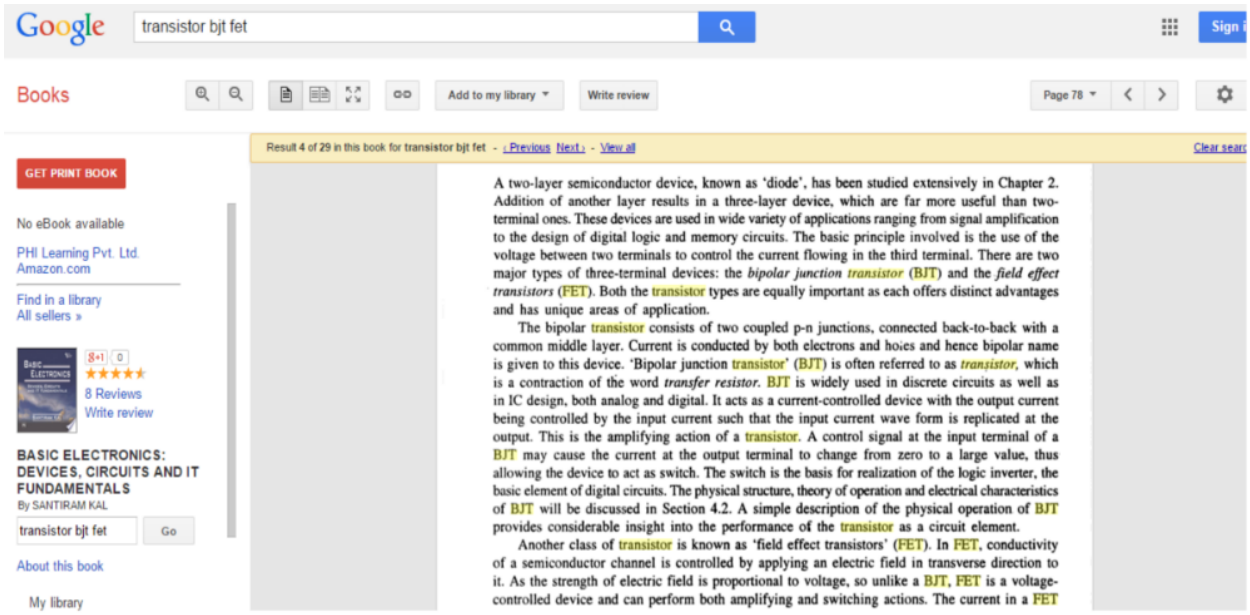

Name
District
Party
Name
Region
Start Date
CONGRESSPERSON
(1, 1)
REPRESENTS
(1, 52)
STATE
(M, M)
(0, N)
VOTED
SPONSORED
How
(1, N)
Date of Vote
(435, 435)
BILL
Passed/ Failed
Bill Name

Problem 4 - Front End Design

A rich man wants to digitize and search his home library and he would like you to develop an application for him. He has fiction books, documents from his work as well as a large collection of books about his favourite subject, bird watching.

One inspiration he has is Google Book Search. That uses scanned images with Optical Character Recognition (OCR) to allow books to be searched.

Develop a simple front end prototype using any web frontend framework that the rich man can use to imagine what the app would look like.



Problem 5 - Back End Development

With the server language of your choice, create an authentication API that does the following.

1. Takes a login and password
2. Checks against a list (either in a database, or saved in a text file) whether than login and password is valid
3. The passwords in the list cannot be readable for a person. Any form of basic encryption is acceptable
4. Before checking if the username is within the database, the API will check if the log in begins with the 2 letter country code of the login location. For example, if the login comes from Singapore, a user name like **sggalvin** will be acceptable,
5. A different response should be returned for a username that does not meet the 2 letter country code criteria.