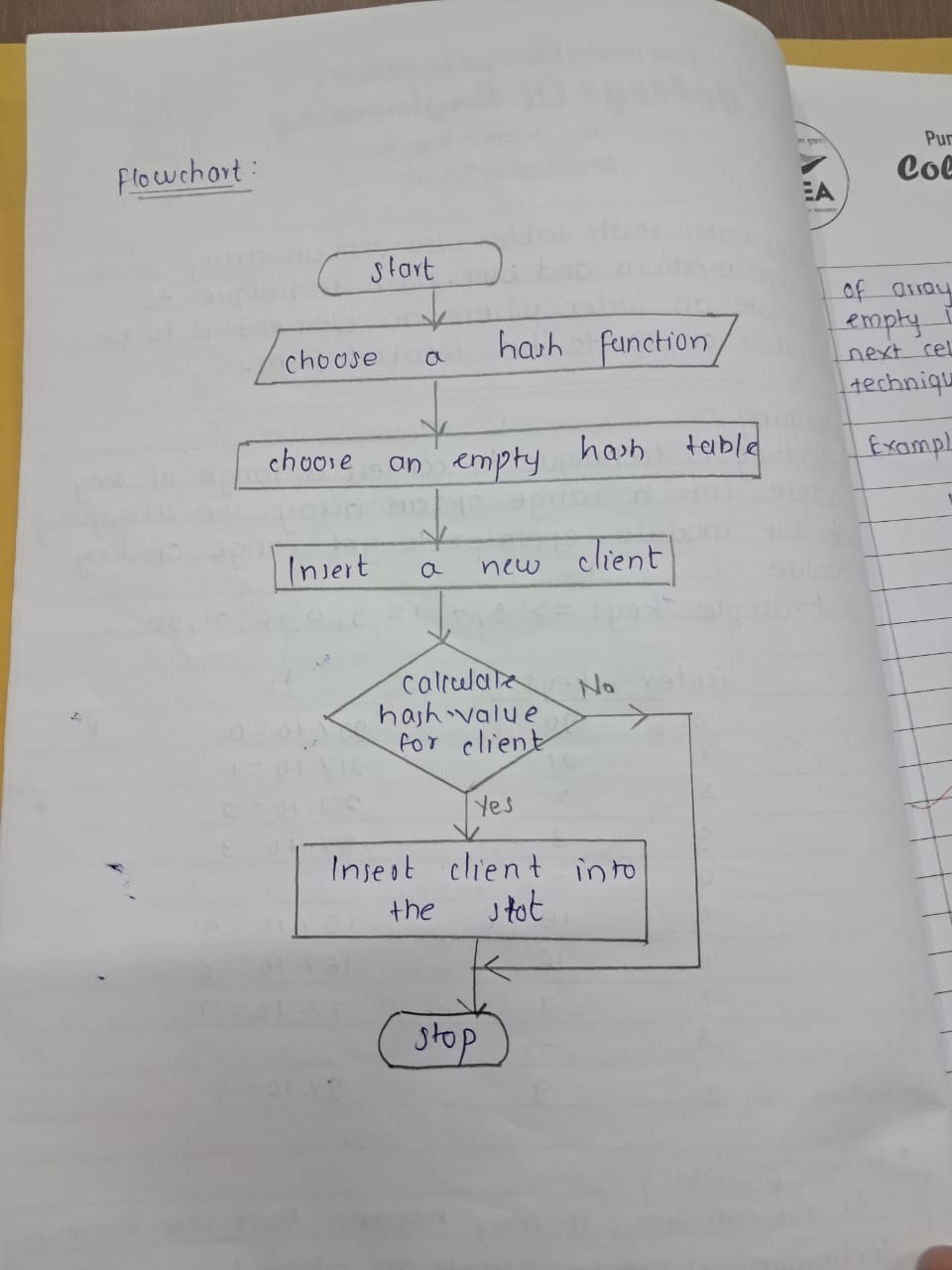
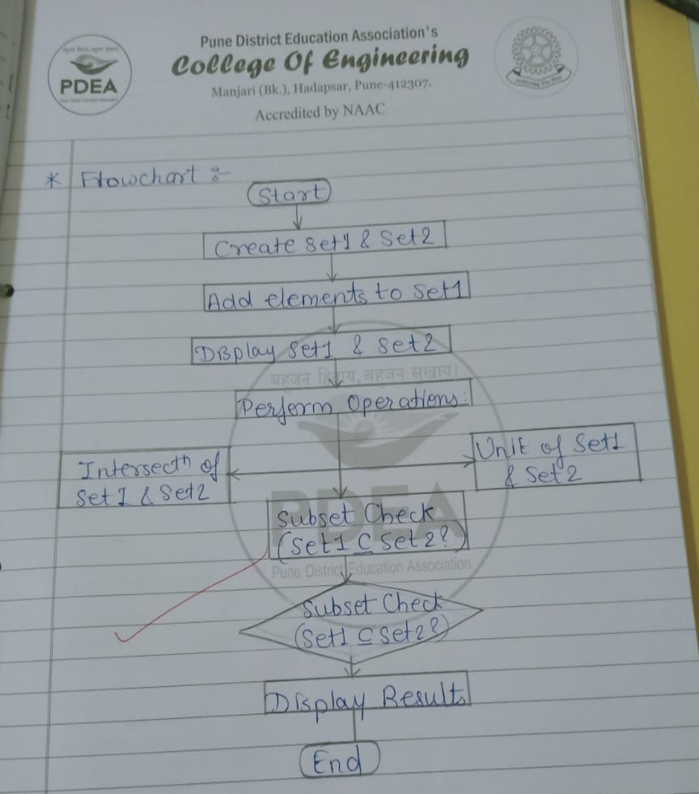
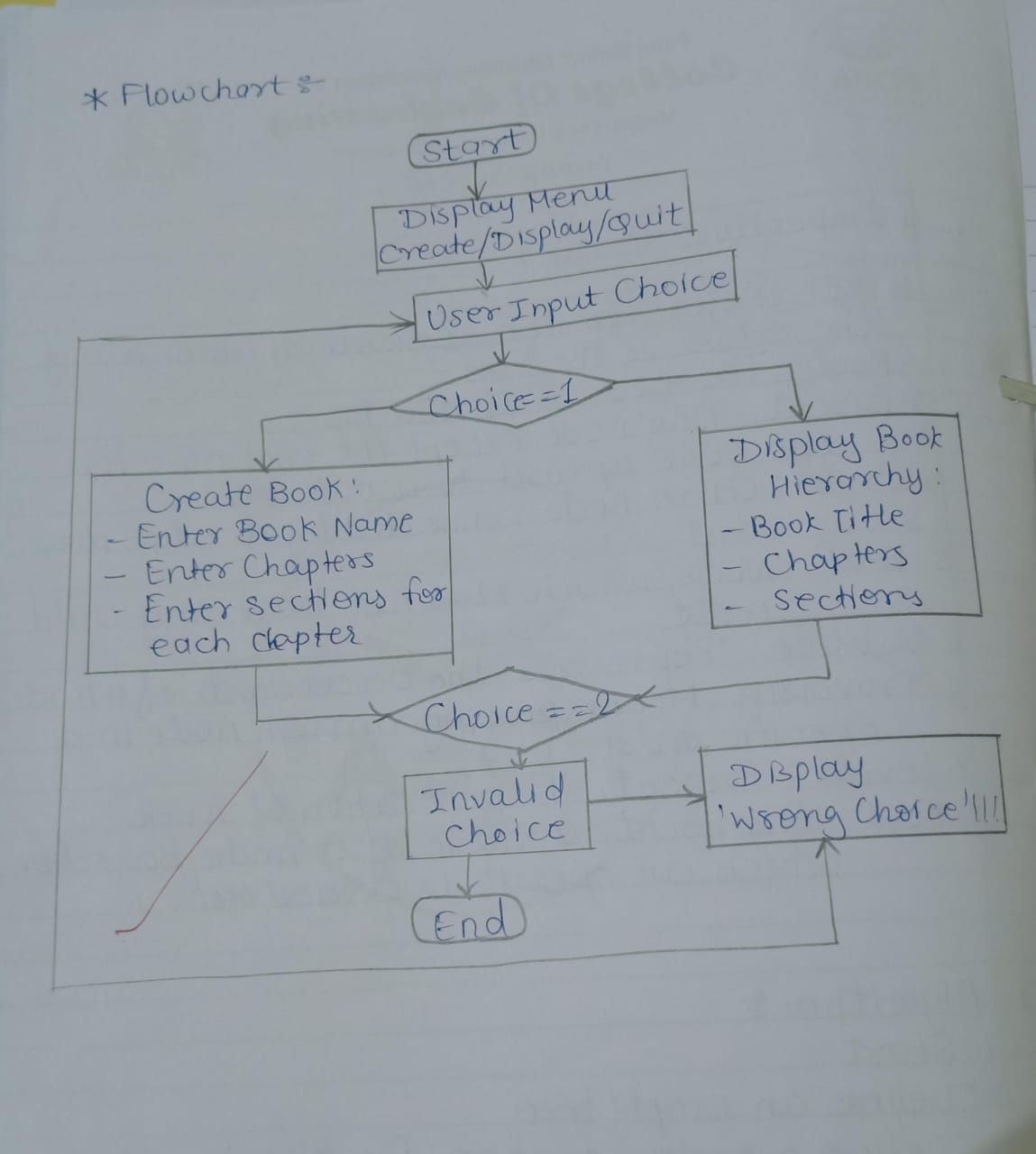
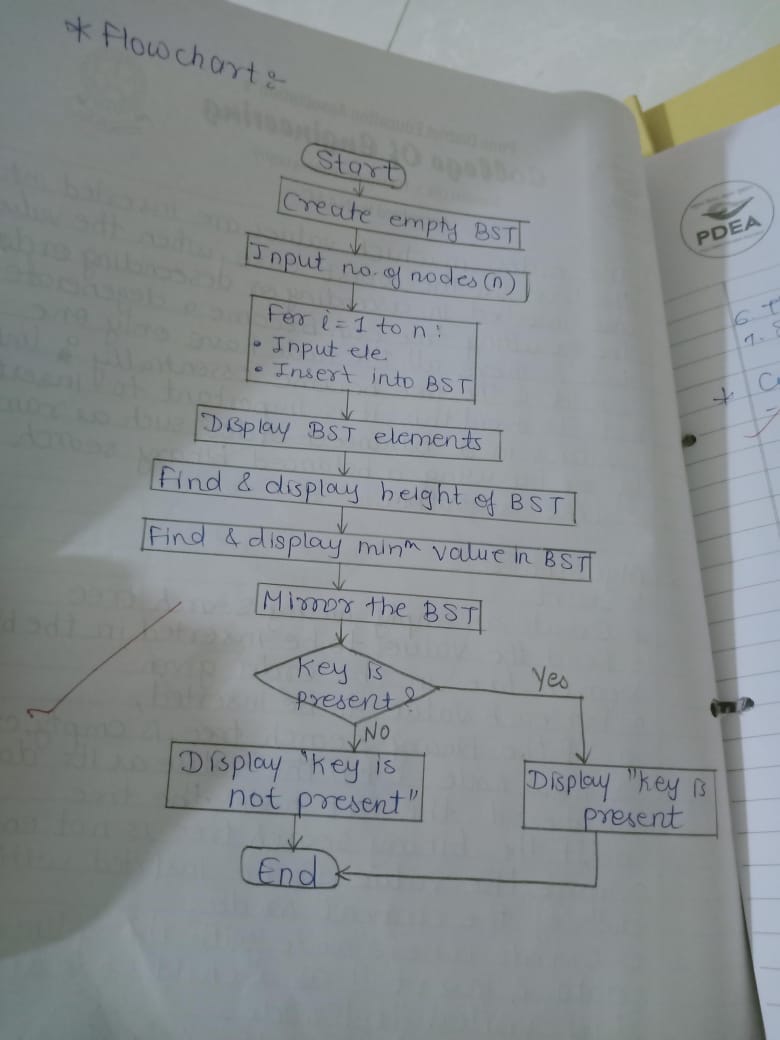
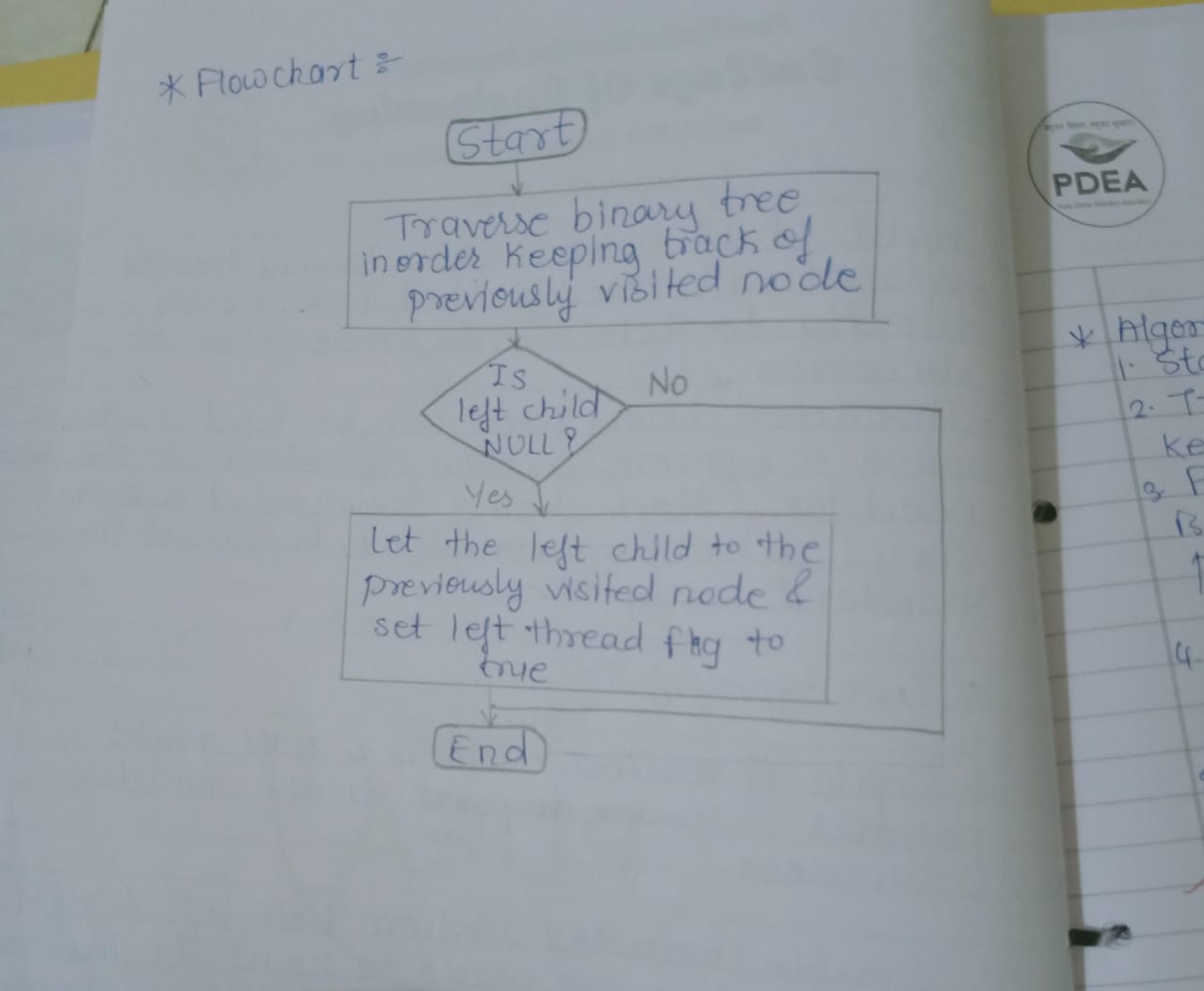
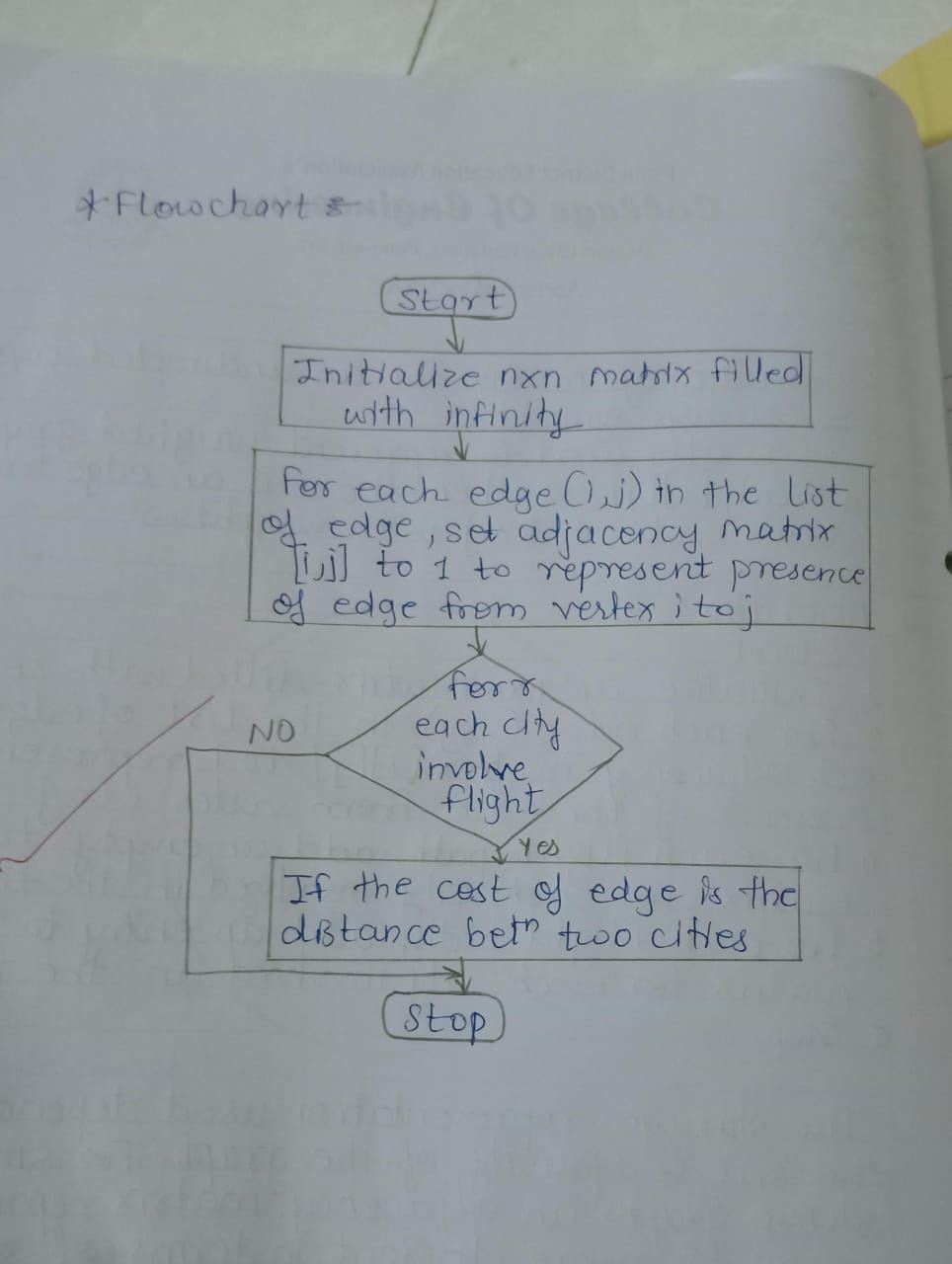
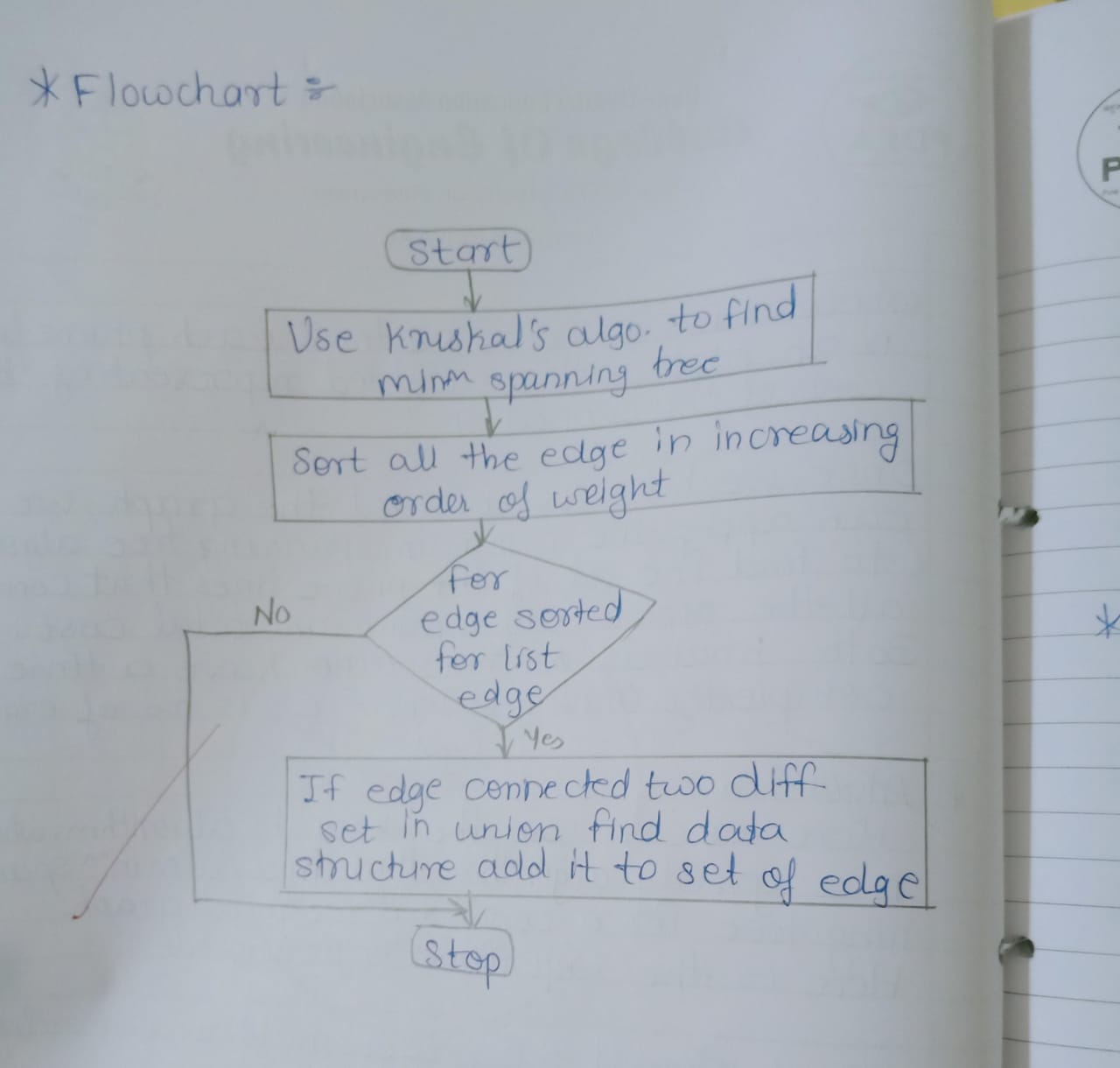
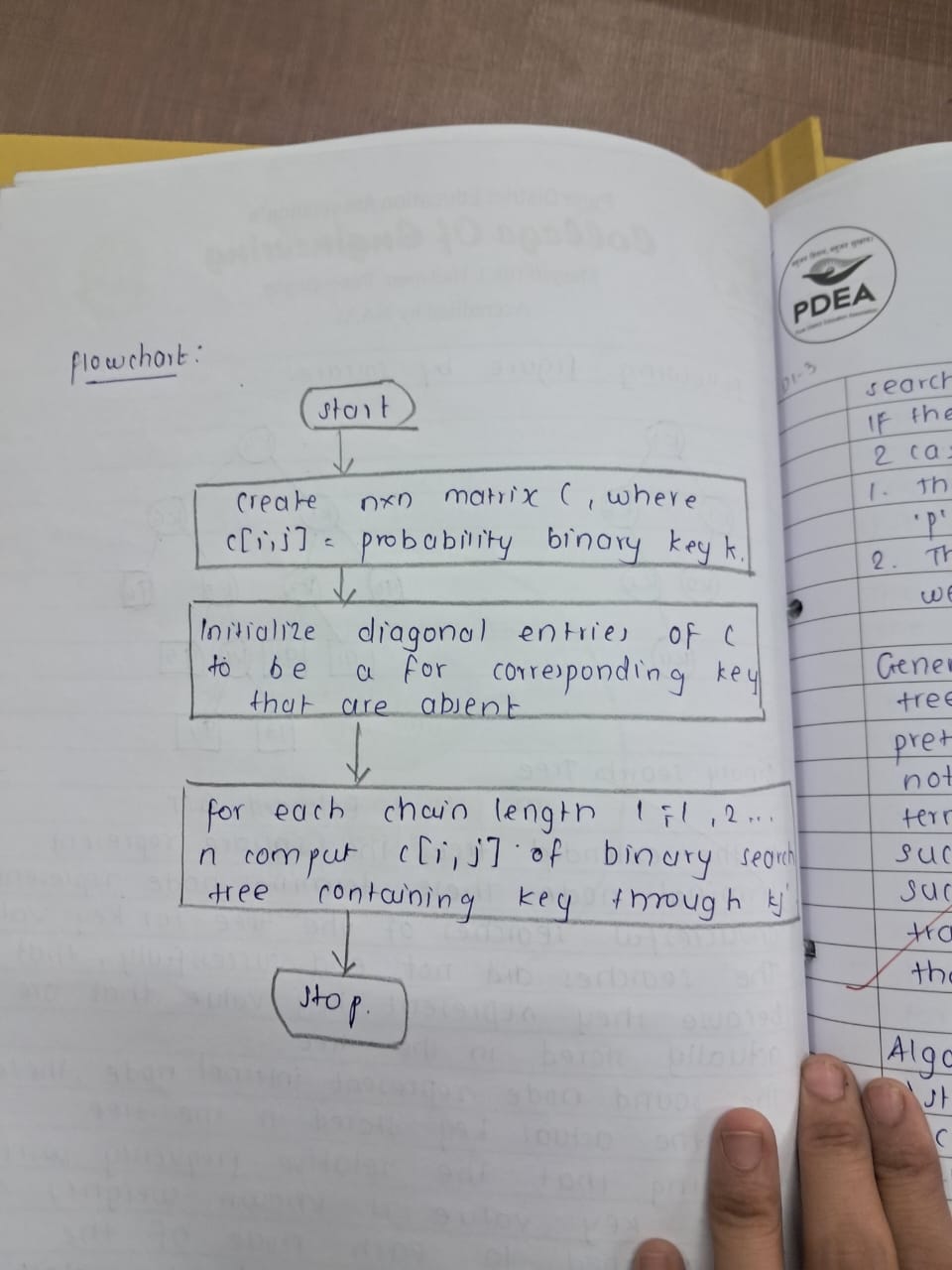
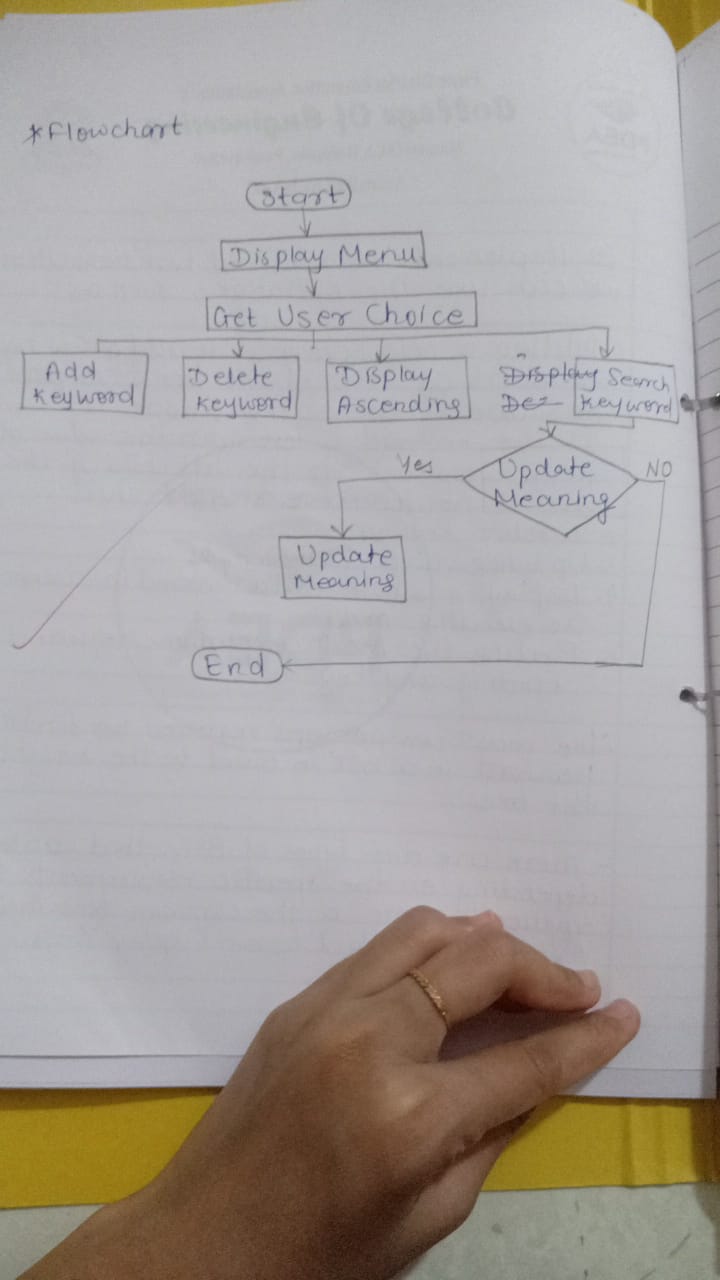
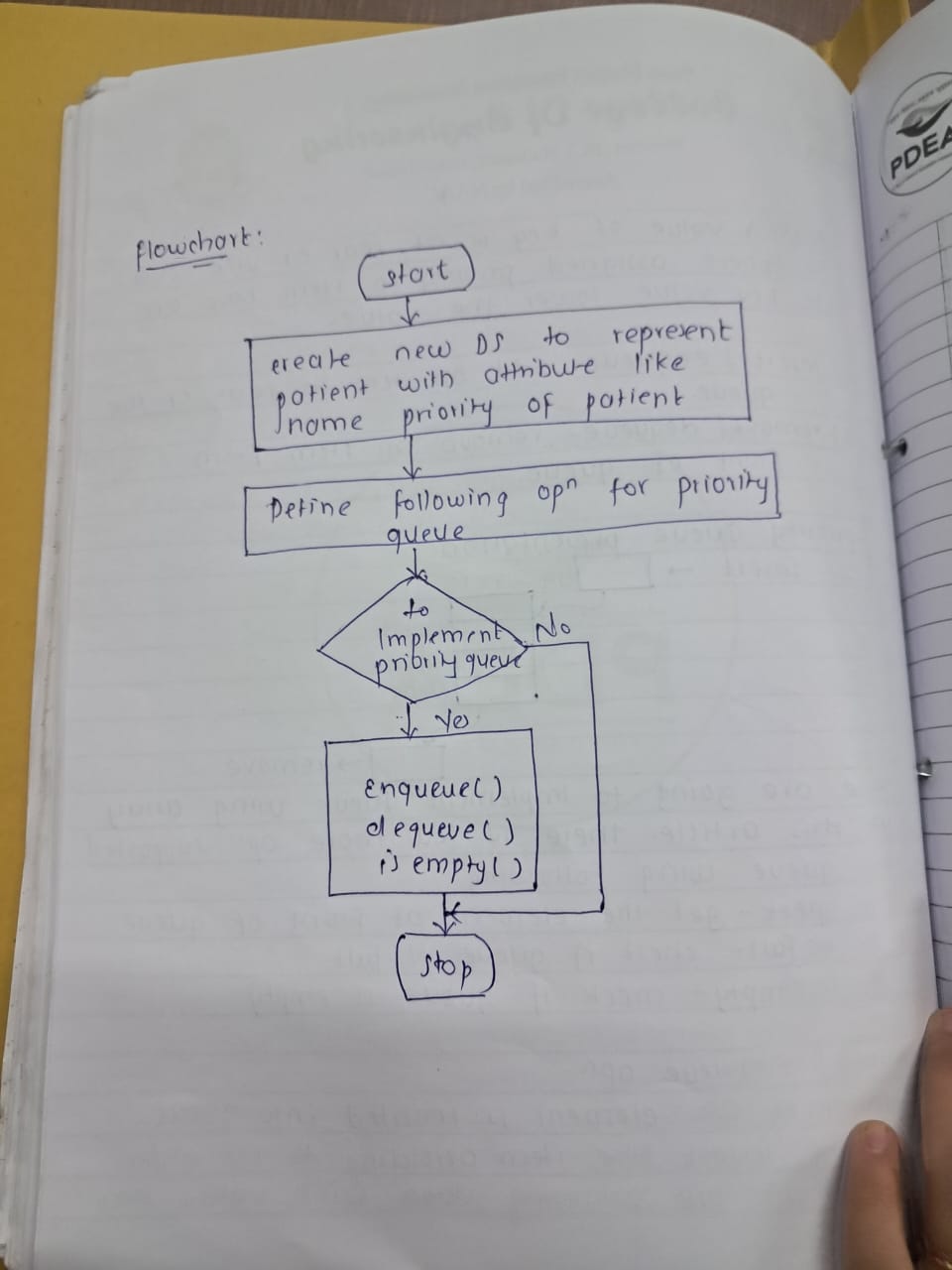
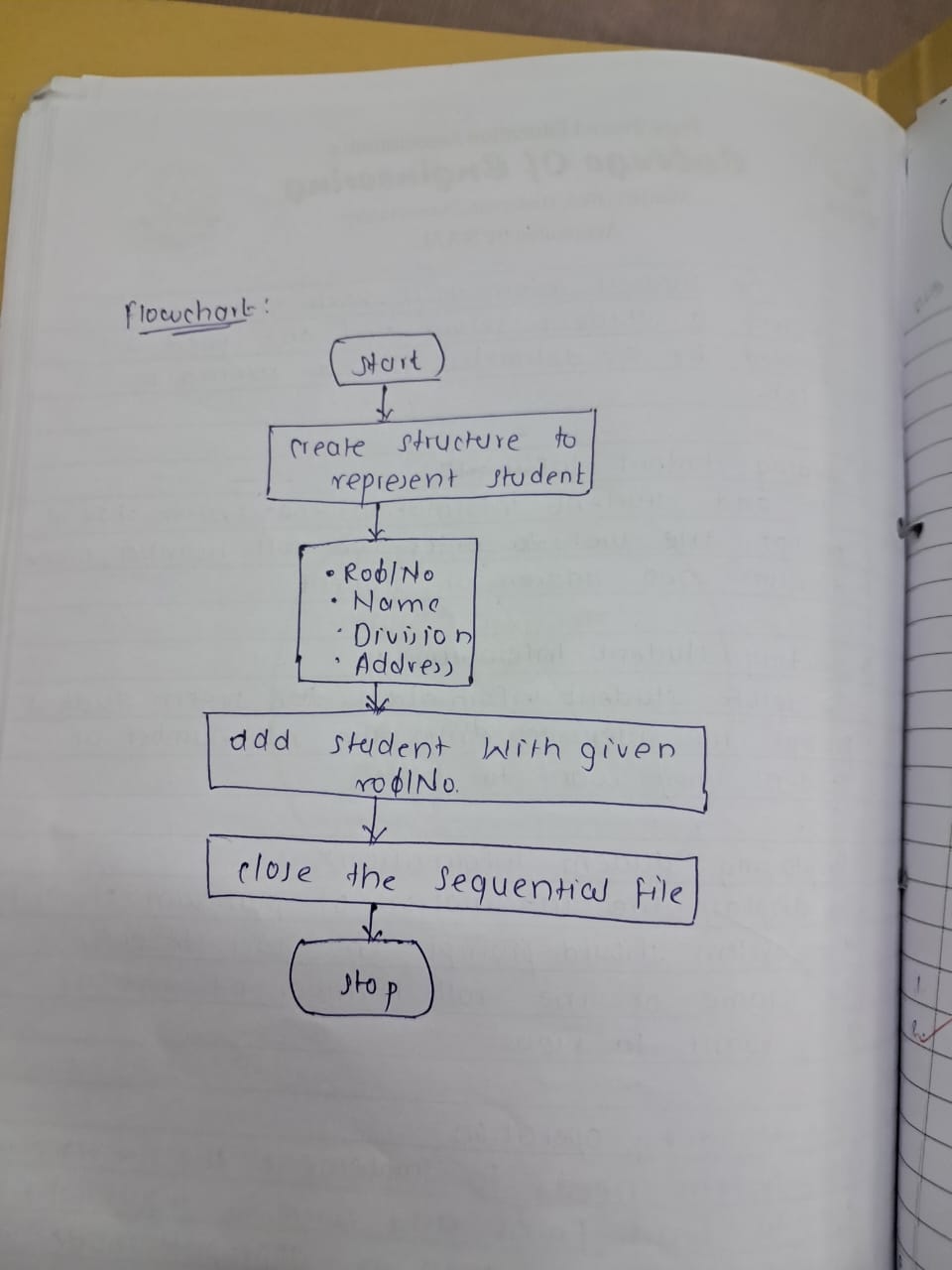
1. Algorithm:- **Telephone database**  
     
   1. Start  
   2. choose a Hash function.  
   3. Chouse empty hash table.  
   4. Insert new hash table.  
   5. 5. Look up a clients telephone no.  
   a. Calculate hash value.  
   b. Check if the slot of index n is empty  
   C. If Slot index n is occupied traverse the linked list until the client with matching hame is found.  
   d. Once matching client is found, return their telephone no.  
   6. Stop

****  
  
  
2) Algorithm:- **Implement set concept**  
  
1. Start  
2. Define set class with private data member.  
3. Define a constructor to intiliatize an array to fixed size.  
4. Define add () method to add new element  
5. Define remove() method.  
6. Define contain () method.  
7. Define size() method.  
8. Define iterator() method.  
9. Define the intersection () method.   
10. Define Union() method.  
11. Define difference () method  
12. Define subset() method.  
13. Stop  
  
  
  
3)Algorithm: **Construct tree & print the nodes**  
  
1. Start  
2. Define empty tree  
3. Define node structure.   
4. Read book & parse it into ch, sec & subsec  
5. Create Root hode for the book  
6.for each chapter, create child hode from root  
7. Create child hode for each section.   
8. Create child hode for each subsection.  
9. Traverse the tree & print the value of node.  
10. calculate time & space complexity.  
11. stop  
  
****  
4) Algorithm : **BST Opertions**  
  
1. Start  
2. Create empty Binary tree.  
3. Read the values To be inserted in the binary search tree in the order given  
4. Do the following   
a) If tree is empty create new node  
b) If tree is not empty , create , compare the values inserted  
C) Create new nodee with the value as its data and insert it is as a child  
5. Repeat step 4 for all values to be inserted  
6. The Binary search tree is now constructed  
7. Stop****  
  
  
  
5) Algorithm- **Convert BST into TBT**  
  
1. Start  
2. Traverse the binary tree in-order, keeping track  
of previously visited hode. 3. Check if left child is node.  
4. If previously visited nodes right child is null  
5. Repeat steps 3 and 4 until all nodes have been visited.  
6. Stop.   
  
  
6) Algorithm: **Flights paths bet cities as a graph**  
1. Start  
2. Initialize an n × n matrix filled with zero  
3. For each edge (I,j) in the list of edges  
4. For each flight path , add on empty to the matrix for each city involved in the flight  
5. If the cost of edge is the distance between two cities store the distance in entry in the matrix   
6. Stop  
  
  
  
7) Algorithm:-.. **MST algo for connecting multiple offices**  
  
1. Sort all the edges in increasing order of their weight  
2. Initialize an empty set of edges & an empty union.  
3. for each edge Connects two different sets in the union data structure   
4. Retun set of edges.  
  
****  
  
8)Algorithms :-  **OBST using dynamic programming**  
  
1.Start  
2. create an n × n matrix c, where C[i][j] represent the cost of binary search tree containing only the keys ki through ki-1.  
3. Initialize the diagonal entries of c to be the search probabilities for corresponding keys.  
4. for each chain length = 1,2,....n-1 each sharing index i=1,2,....n-1 compute the Cost of binary search tree.  
5. The minimum cost of binary search tree containing all keys is c [1] [n]  
6. Stop.  
  
  
9) Algorithm:- **Imple of dictionary using BST for searching & sorting**  
  
1. Start  
2. Define empty tree.  
3. Define function insert() to add new keyword.  
4. Define function delete () to delete key word.  
5. Define function update () to update value of key word.  
6. Define function as in order traversal () to displaying the data in ascending order.  
7. Define function as max comparisons () for finding the maximum comparisons required for finding any keyword  
8. Stop  
  
  
  
10) Algorithm :- **Priority Queue**  
  
1. Start  
2. Create a class or data structure to represent a patient.  
3. Create a priority queue data structure  
4. Define following operations  
a. Enqueue()  
b. Dequeue()  
c. is Empty()  
5. To implement priority queue.  
6. stop  
  
  
  
  
11) Algorithm: **File sequencing Management**  
  
1. Start  
2. Create a structure to represent a student with full attributes.  
a. Roll no.  
b. Name  
C. Division  
d. Address  
3. Define operations add student()  
4. Define function name delete student ()  
5. Define Display Student()  
6. close sequential file.  
7. Now, we can open the file  
8. stop.  
  
  
  
  
  
  
12) Algorithm:-  **Using sequential file handling to maintain data**  
  
To add the employee using indexed seql file org.   
  
1. Open the file for writing.  
2. write new emp employee's name  
3. update the index.  
4. close file  
  
  
To delete an employee   
  
1. Open file for reading.  
   2. find employee's record.  
3. Delete the employee.  
4. update the index.  
5. close the file.   
  
To display Info about particular employee using index file org.  
1. Open the file  
2. Use index to find employee  
3. Display employees info  
4. Close the file

