1. Design a program to take user input for a matrix. Sort the matrix column-wise in ascending order. After sorting, perform an M-way merge to transform the sorted matrix into a linear array.”

2. Write a program to manage a library catalog using an array-based data structure. Implement functionalities to add books, display the catalog, and search for books by ISBN. Also includes a checkout system using a stack data structure such as implementing functionalities to check out books, display the stack of checked-out books, and handle returning books.

3. Implement tree traversal algorithms to traverse a file system and search for specific files or directories. The file system is represented as a tree structure, where each node represents a file or directory, and directories can contain subdirectories and files. Print the path for all files in the File System Tree.

   File System Tree:

   - Documents
     - Report.docx
     - Photos
       - SummerVacation.jpg
   - Programs
     - Python
       - script.py
   - Music
     - Playlist
       - Track1.mp3

4. Write a program to manage a student enrollment system using linked lists. Implement functionalities to add students, display the enrollment list, and search for students by their student ID. Additionally, incorporate a waitlist system using a queue data structure to manage overflow. Implement functionalities to add students to the waitlist, display the waitlist, and enroll students from the waitlist when spots become available.

5. Design a parcel delivery tracking system using a linked list. Implement functionalities to add parcels, display the parcel list, and search for parcels by

tracking number. Additionally, include a delivery status update system using a stack data structure to manage status changes. Implement functionalities to update delivery status, and display the stack of status changes.

6. Implement a hashing table by keeping a binary search tree for chaining while collision.

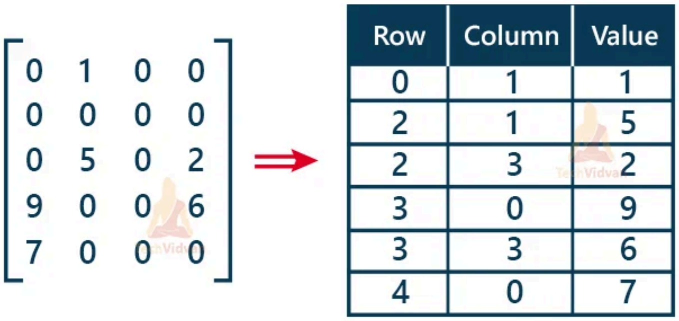
7. Develop a program to process a matrix provided by the user. The program should search for a specific element within the matrix, displaying its position if found. Also, print the matrix in a spiral way.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Result: 13 8 9 14 19 18 17 12 7 8 9 10 15 20 25 24 23 22 21 16 11 6 1 2 3 4 5

8. Recursively implement a merge sort so that the dividing process must stop at a point where the sub-array has five or fewer element elements. Use insertion sort to sort that sub-array and proceed with the merging.

9. Write a program to read and convert the matrix into a sparse matrix. Then implement matrix addition on two sparse matrices.



$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 2 \\ 9 & 0 & 0 & 6 \\ 7 & 0 & 0 & 0 \end{bmatrix} \Rightarrow$$

Row	Column	Value
0	1	1
2	1	5
2	3	2
3	0	9
3	3	6
4	0	7

10. Implement a priority queue using a linked list to keep patient_id along with its priority. The same idea of queue but while deque must happen according to the priority value. (Least value has more priority).

11. Implement a program to print the path between any two nodes in a binary tree.