DAA [Day 1]

UID: 24MCI10204

Name: Rahul Saxena

Branch: 24MCA – AI & ML

Question 1:

a) Develop Sparse Matrix using Linked List.

b) Traverse Matrix in Spiral Manner.

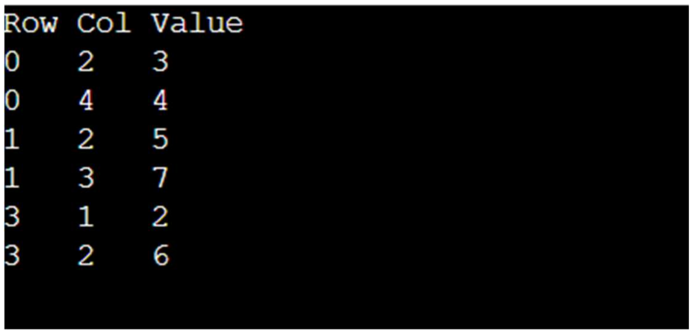
Answer:

Develop Sparse Matrix Using Linked list

```
class Node {
   int row, col, val;
   Node next;
   Node(int row, int col, int val) {
      this.row = row;
      this.col = col;
      this.val = val;
      this.next = null;
   }
}
public class SparseMatrixLinkedList {
   static Node head = null;
   static void insert(int row, int col, int val) {
      if (val == 0) return;
      Node newNode = new Node(row, col, val);
      if (head == null) {
         head = newNode;
      } else {
         Node temp = head;
         while (temp.next != null)
            temp = temp.next;
         temp.next = newNode;}}
   static void display() {
      System.out.println("Row Col Value");
      Node temp = head;
      while (temp != null) {
         System.out.println(temp.row + "   " + temp.col + "   " + temp.val);
         temp = temp.next;}}
   public static void main(String[] args) {
      int[][] matrix = {
         {0, 0, 3, 0, 4},
         {0, 0, 5, 7, 0},
         {0, 0, 0, 0, 0},
         {0, 2, 6, 0, 0}
      };
      for (int i = 0; i < matrix.length; i++)
         for (int j = 0; j < matrix[0].length; j++)
            insert(i, j, matrix[i][j]);
```

```
    display(); }}
```

Output:

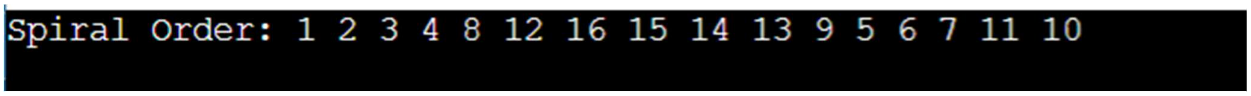


```
Row Col Value
0   2   3
0   4   4
1   2   5
1   3   7
3   1   2
3   2   6
```

Traverse Matrix in Spiral Manner

```
public class SpiralMatrix {
  static void spiralTraverse(int[][] mat) {
    int top = 0, bottom = mat.length - 1;
    int left = 0, right = mat[0].length - 1;
    while (top <= bottom && left <= right) {
      for (int i = left; i <= right; i++)
        System.out.print(mat[top][i] + " ");
      top++;
      for (int i = top; i <= bottom; i++)
        System.out.print(mat[i][right] + " ");
      right--;
      if (top <= bottom) {
        for (int i = right; i >= left; i--)
          System.out.print(mat[bottom][i] + " ");
        bottom--;
      }
      if (left <= right) {
        for (int i = bottom; i >= top; i--)
          System.out.print(mat[i][left] + " ");
        left++;
      }
    }
  }
  public static void main(String[] args) {
    int[][] matrix = {
      {1, 2, 3, 4},
      {5, 6, 7, 8},
      {9, 10, 11, 12},
      {13, 14, 15, 16}
    };
    System.out.print("Spiral Order: ");
    spiralTraverse(matrix);
  }
}
```

Output:



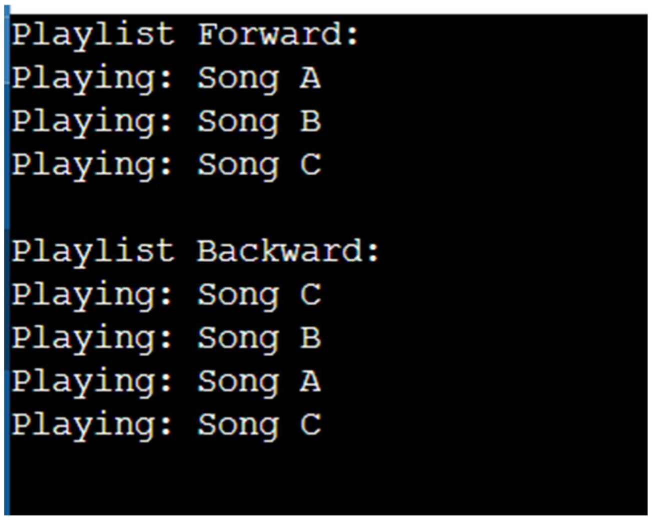
```
Spiral Order: 1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10
```

Question 2: Simulate Music Playlist using Circular Doubly Linked List

Answer:

```
class Song {
   String title;
   Song prev, next;
   Song(String title) {
      this.title = title;
      this.prev = this.next = null;
   }
}
public class Playlist {
   static Song tail = null;
   static void addSong(String title) {
      Song newSong = new Song(title);
      if (tail == null) {
         newSong.next = newSong.prev = newSong;
         tail = newSong;
      } else {
         newSong.next = tail.next;
         newSong.prev = tail;
         tail.next.prev = newSong;
         tail.next = newSong;
         tail = newSong;
      }
   }
   static void playForward() {
      if (tail == null) return;
      Song temp = tail.next;
      do {
         System.out.println("Playing: " + temp.title);
         temp = temp.next;
      } while (temp != tail.next);
   }
   static void playBackward() {
      if (tail == null) return;
      Song temp = tail;
      do {
         System.out.println("Playing: " + temp.title);
         temp = temp.prev;
      } while (temp != tail);
      System.out.println("Playing: " + temp.title);
   }
   public static void main(String[] args) {
      addSong("Song A");
      addSong("Song B");
      addSong("Song C");
      System.out.println("Playlist Forward:");
      playForward();
      System.out.println("\nPlaylist Backward:");
      playBackward();
   }
}
```

Output:



```
Playlist Forward:
Playing: Song A
Playing: Song B
Playing: Song C

Playlist Backward:
Playing: Song C
Playing: Song B
Playing: Song A
Playing: Song C
```