**Question:**

A class named "BoundaryElementSort" with the following details:

**Data members:**

* "matrix": a 2D array to store the elements of the matrix.
* "m": an integer to store the number of rows.
* "n": an integer to store the number of columns.

**Member methods:**

* "BoundaryElementSort(int m, int n)": a parameterized constructor to initialize the 2D array and the dimensions of the matrix.
* "acceptMatrix()": a method to accept the elements into the matrix.
* "sortBoundaryElements()": a method to sort the boundary elements of the matrix in ascending order.
* "displayMatrix()": a method to display the original and sorted matrix.

In the main method, create an object of the "BoundaryElementSort" class, accept the matrix, sort the boundary elements, and display the original and sorted matrix.

**Algorithm :**

1. Start

2. Import the `Arrays` and `Scanner` classes from the `java.util` package.

3. Define a class `BoundaryElementSort` with the following instance variables:

- `matrix` of type `int[][]` to store the elements of the matrix.

- `m` of type `int` to store the number of rows.

- `n` of type `int` to store the number of columns.

4. Define a parameterized constructor for the class `BoundaryElementSort`:

- Accept two integers `m` and `n` as parameters representing the number of rows and columns.

- Initialize `this.m` with `m`.

- Initialize `this.n` with `n`.

- Initialize `this.matrix` with a new 2D array of size `m` x `n`.

5. Define a method `acceptMatrix()` to accept the elements into the matrix:

- Create a `Scanner` object to read input from the user.

- Print "Enter the elements of the matrix:".

- Loop through each row from `0` to `m-1`:

- Loop through each column from `0` to `n-1`:

- Read an integer from the user and store it in `matrix[i][j]`.

6. Define a method `sortBoundaryElements()` to sort the boundary elements of the matrix in ascending order:

- Create an array `boundaryElements` of size `2 \* (m + n - 2)` to store the boundary elements.

- Initialize an index variable `index` to `0`.

- Loop through each row from `0` to `m-1`:

- Loop through each column from `0` to `n-1`:

- If the element is on the boundary (i.e., `i == 0` or `i == m - 1` or `j == 0` or `j == n - 1`), add it to `boundaryElements` and increment `index`.

- Sort the `boundaryElements` array in ascending order using `Arrays.sort()`.

- Reset `index` to `0`.

- Place the sorted boundary elements back into the matrix:

- Loop through each column from `0` to `n-1` and assign `boundaryElements[index++]` to `matrix[0][j]`.

- Loop through each row from `1` to `m-1` and assign `boundaryElements[index++]` to `matrix[i][n - 1]`.

- Loop through each column from `n-2` to `0` and assign `boundaryElements[index++]` to `matrix[m - 1][j]`.

- Loop through each row from `m-2` to `1` and assign `boundaryElements[index++]` to `matrix[i][0]`.

7. Define a method `displayMatrix()` to display the original and sorted matrix:

- Print "Original Matrix:".

- Call the `printMatrix()` method to print the original matrix.

- Call the `sortBoundaryElements()` method to sort the boundary elements.

- Print "Sorted Boundary Elements Matrix:".

- Call the `printMatrix()` method to print the matrix with sorted boundary elements.

8. Define a helper method `printMatrix(int[][] matrix)` to print the matrix:

- Loop through each row from `0` to `m-1`:

- Loop through each column from `0` to `n-1`:

- Print the element `matrix[i][j]` followed by a space.

- Print a newline character.

9. Define the `main` method to test the `BoundaryElementSort` class:

- Create a `Scanner` object to read input from the user.

- Print "Enter the number of rows and columns:".

- Read two integers `m` and `n` from the user.

- Create an object of the class `BoundaryElementSort` with the parameters `m` and `n`.

- Call the `acceptMatrix()` method to accept the elements into the matrix.

- Call the `displayMatrix()` method to display the original and sorted matrix.

10. End

**Variable Description Table**

|  |  |  |
| --- | --- | --- |
| **Variable Name** | **Variable Type** | **Description** |
| matrix | int[][] | A 2D array to store the elements of the matrix. |
| m | int | An integer to store the number of rows in the matrix. |
| n | int | An integer to store the number of columns in the matrix. |
| boundaryElements | int[] | An array to store the boundary elements of the matrix. |
| index | int | An integer to keep track of the current index in the boundaryElements array. |
| i | int | A loop control variable for iterating through rows. |
| j | int | A loop control variable for iterating through rows.A loop control variable for iterating through columns. |
| opt | String | A temporary variable to store the last entered book name |
| nameOfBook | String | A variable to store the name of the book entered by the user . |