**Spiral Traversal of a Matrix:**

function TraverseSpiral(matrix) {

  if (matrix.length === 0) return [];

  let result = [];

  let top = 0, bottom = matrix.length - 1;

  let left = 0, right = matrix[0].length - 1;

 while (top <= bottom && left <= right) {

    for(let i = left; i <=right; i++) {

        result.push(matrix[top][i])

    }

    top ++;

    for(let i = top; i<=bottom; i++) {

        result.push(matrix[i][right])

    }

    right --;

    if (top <= bottom) {

        for (let i = right; i >= left; i--) {

            result.push(matrix[bottom][i]);

        }

        bottom--;

    }

    if (left <= right) {

        for (let i = bottom; i >= top; i--) {

            result.push(matrix[i][left]);

        }

        left++;

    }

 }

 return result

}

const matrix4x5 = [

  [17, 2, 33, 4],

  [6, 77, 8, 9],

  [11, 12, 123, 14],

  [16, 17, 88, 19],

];

console.log(TraverseSpiral(matrix4x5));

**QuickSort Array:**

class ReadlineConsole {

  constructor() {

    this.numbers = [];

    this.readline = require("readline").createInterface({

      input: process.stdin,

      output: process.stdout,

    });

  }

  async getNumbers() {

    const ask = async (question) => {

      return new Promise((resolve) => {

        this.readline.question(question, resolve);

      });

    };

    let input = await ask(

      "Enter required number of integers separated by spaces and then press enter: "

    );

    input = input.trim().replace(/\s+/g, " ");

    let numbersArray = input.split(" ");

    for (let i = 0; i < numbersArray.length; i++) {

      let number = parseInt(numbersArray[i]);

      if (!isNaN(number)) {

        this.numbers.push(number);

      }

    }

    this.readline.close();

  }

  async showNumbers() {

    for (let i = 0; i < this.numbers.length; i++) {

      console.log(this.numbers[i]);

    }

  }

  //Step 1 - Pick a pivot point any element

  //Step 2 - Take that element and place it in the correct place in the sorted arrray

  //Step 3 - Put everything that's smaller than the pivot into a 'left' array and everything that's greater than the pivot into a 'right' array

  //Step 4 - Repeat the process for the individual ‘left' and 'right' arrays till you have an array of length 1 which is sorted by definition

  //Step 5 - Repeatedly concatenate the left array, pivot and right array till one sorted array remains

  quickSort(arr) {

    if (arr.length < 2) {

      return arr;

    }

    let pivot = arr[arr.length - 1];

    let left = [];

    let right = [];

    for (let i = 0; i < arr.length - 1; i++) {

      if (arr[i] < pivot) {

        left.push(arr[i]);

      } else {

        right.push(arr[i]);

      }

    }

    return [...this.quickSort(left), pivot, ...this.quickSort(right)];

  }

}

(async () => {

  const readConsole = new ReadlineConsole();

  await readConsole.getNumbers();

  readConsole.showNumbers();

  const product = readConsole.quickSort(readConsole.numbers);

  console.log(product);

})();