**Program :**

Design, develop, code and run the program in any suitable language to implement the Next Date function. Analyze it from the perspective of boundary value testing, derive different test cases, execute these test cases and discuss the test results.

**CODE:**

function isLeapYear(year) {

return (year % 4 === 0 && year % 100 !== 0) || (year % 400 === 0);

}

function getNextDate(inputDate) {

const [year, month, day] = inputDate.split('-').map(Number);

if (month < 1 || month > 12 || day < 1 || day > 31) {

return "Invalid date";

}

if (month === 2) {

if (isLeapYear(year)) {

if (day < 29) {

return `${year}-${month.toString().padStart(2, '0')}-${(day + 1).toString().padStart(2, '0')}`;

} else if (day === 29) {

return `${year}-${(month + 1).toString().padStart(2, '0')}-01`;

} else {

return "Invalid date";

}

} else {

if (day < 28) {

return `${year}-${month.toString().padStart(2, '0')}-${(day + 1).toString().padStart(2, '0')}`;

} else if (day === 28) {

return `${year}-${(month + 1).toString().padStart(2, '0')}-01`;

} else {

return "Invalid date";

}

}

} else if ([4, 6, 9, 11].includes(month)) {

if (day < 30) {

return `${year}-${month.toString().padStart(2, '0')}-${(day + 1).toString().padStart(2, '0')}`;

} else if (day === 30) {

return `${year}-${(month + 1).toString().padStart(2, '0')}-01`;

} else {

return "Invalid date";

}

} else {

if (day < 31) {

return `${year}-${month.toString().padStart(2, '0')}-${(day + 1).toString().padStart(2, '0')}`;

} else if (day === 31) {

if (month === 12) {

return `${year + 1}-01-01`;

} else {

return `${year}-${(month + 1).toString().padStart(2, '0')}-01`;

}

} else {

return "Invalid date";

}

}

}

// Boundary Value Tests

const testCases = [

{ input: "0001-01-01", expectedOutput: "0001-01-02" },

{ input: "9999-12-31", expectedOutput: "Invalid date" },

{ input: "2020-02-28", expectedOutput: "2020-02-29" },

{ input: "2023-01-31", expectedOutput: "2023-02-01" },

{ input: "2023-02-28", expectedOutput: "2023-03-01" },

{ input: "2023-04-30", expectedOutput: "2023-05-01" },

{ input: "2023-13-01", expectedOutput: "Invalid date" },

{ input: "2023-02-29", expectedOutput: "Invalid date" }

];

for (const testCase of testCases) {

const result = getNextDate(testCase.input);

console.log(`Input Date: ${testCase.input}, Next Date: ${result}, Expected Output: ${testCase.expectedOutput}`);

}

**OUTPUT:**

**A white text with black text

Description automatically generated**

FAILURE CASES:

output:

Enter the size of a: 2 3

Enter the size of b: 2 3

Matrix addition is not possible