Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

**Symbol** **Value**

I 1

V 5

X 10

L 50

C 100

D 500

M 1000

For example, 2 is written as II in Roman numeral, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used:

* I can be placed before V (5) and X (10) to make 4 and 9.
* X can be placed before L (50) and C (100) to make 40 and 90.
* C can be placed before D (500) and M (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer.

**Input:** s = "MCMXCIV"

**Output:** 1994

Test on Dictionary, loop, and conditional statements

Note: Procedural Code or Process Flow or Flow Chart

1. Get the parameter as roman numeral
2. Create array/object of roman numeral vs value (1, 5, 10, 50, 100, 500, 1000, …..)
3. Create temporary variable for output
4. Split the roman numeral by to a single or 2 characters,
   1. To determine if 2 characters in sequence, check if:
      1. If V precedence of I, means IV or 4
      2. If L precedence of X, means XL or 40
      3. If C precedence of X, means XC or 90
      4. If D precedence of C, means CD or 400
      5. If M precedence of C, means CM or 900
5. Convert every split character or 2 character to specified number base on array/object of roman numeral
   1. Loop thru every character to convert them to corresponding numerical value
   2. Store the value to temporary output
6. Print or return the output variable

Given an array of intervals where intervals[i] = [starti, endi], merge all overlapping intervals, and return *an array of the non-overlapping intervals that cover all the intervals in the input*.

**Example 1:**

**Input:** intervals = [[1,3],[2,6],[8,10],[15,18]]

**Output:** [[1,6],[8,10],[15,18]]

**Explanation:** Since intervals [1,3] and [2,6] overlap, merge them into [1,6].

**Example 2:**

**Input:** intervals = [[1,4],[4,5]]

**Output:** [[1,5]]

**Explanation:** Intervals [1,4] and [4,5] are considered overlapping.

Test on List, sorting, and conditional statements

Note: Procedural Code or Process Flow or Flow Chart

1. Get the input value of array
2. Sort the array by ascending
3. Iterate/loop on every sub-array
4. Every iteration:
   1. Check if current array value (arr) is greater than the next array value (arr + 1).
   2. If current array value is greater than the next then move to the next array value (arr + 2) and so on (arr + n)…..
   3. If the (arr + n) is less than current array then use the current array and the (arr + n – 1) value
   4. Get the first value of current array and the last value of (arr + n – 1) value ex: [1, 5]
5. Return the array from the iteration.