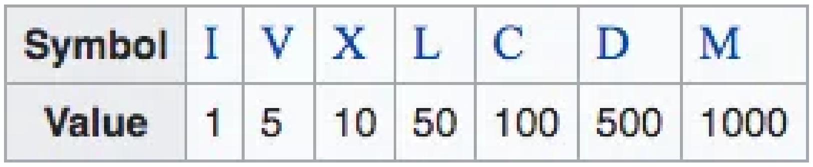
Lab: Roman Numeral to Integer: How to solve using a HashMap

Roman numerals are represented by seven different symbols. The following symbols and their values are as follows:



Symbol	I	V	X	L	C	D	M
Value	1	5	10	50	100	500	1000

Let’s start with some simple examples. The number “3” is written as III in Roman numeral, where we add the three one’s together. The number “7” is written as VII, where we add “V”, which is 5, to II, which is 2, resulting in 5+2 = 7. The number “11” is written as XI, where we add “X”, which is 10, to I, which is 1, resulting in 10+1 = 11.

With the above examples, we have them written in a way where the values are written from largest to smallest from left to right. However, there are instances where this is not the case.

For instance, “4” is not written as “IIII”. The roman numeral for 4 is written as IV. In this case, because the one is before the five, we use subtraction instead of the addition to get 4. The same principle applies to “9”, which in roman numerals is written as IX.

There are six different instances in which this principle applies:

1. I can be placed before V (4)
2. I can be placed before X (9)
3. X can be placed before L (40)
4. X can be placed before C (90)
5. C can be placed before D (400)
6. C can be placed before M (900)

So with these exceptions, we simply can’t just implement a solution where we add up all of the numbers — there will be instances where we have to subtract the numbers.

So what can we use to help us with this? A hash map!

Given a roman numeral, write a program use HashMap (map in C++) convert it to an integer (from 1 to 1000)