Write an efficient function

```c
int rightmost (double x, const vector<double>& v);
```

that returns the rightmost position where \( x \) could be inserted in the sorted vector \( v \) (by adding one position at the end of \( v \) and moving the necessary elements one position to the right) so that \( v \) would remain sorted.

For instance, assume that \( x \) is 23. If \( v \) is \([15, 15, 20, 30, 40, 40]\), then we must insert \( x \) at the position 3 (between 20 and 30), and the resulting \( v \) would be \([15, 15, 20, 23, 30, 40, 40]\). If \( v \) is \([17, 23, 23, 35, 42, 42]\), then we could insert \( x \) at the positions 1, 2 or 3, so your function must return 3. If \( v \) is \([3, 5, 7, 9]\), \( x \) should be inserted at the position just to the right of the end of the vector, that is, 4. As a final example, if \( v \) is \([23, 23]\), \( x \) should be inserted at 2.

**Precondition**

The vector \( v \) is sorted in nondecreasing order.

**Observations**

- Your solution can only include the `vector` library.
- You can write and use additional functions if you need them.
- You only need to submit the required procedure; your main program will be ignored.

**Problem information**

Author: Salvador Roura
Translator: Salvador Roura
Generation: 2016-05-19 17:54:20

http://www.jutge.org