Occurrences of a double subvector X17935_en

Write the function

```
int subvector_from_k (const vector<int>& x, const vector<int>& y, int k)
```

that checks whether vector y contains the subvector x (doubled) at position k o higher. If it is found, it returns in which position, otherwise, it returns -1
For instance, if $x = [1, 2]$ e $y = [1, 2, 3, 4, \mathbf{2}, \mathbf{4}, 6, \mathbf{2}, \mathbf{4}, 8, 9]$, the function results would be:

- If $k = 0$, it would return 4 (first group $[\mathbf{2}, \mathbf{4}]$, at position 4)
- If $k = 2$, it would return 4 (first group $[\mathbf{2}, \mathbf{4}]$, at position 4)
- If $k = 5$, it would return 7 (second group $[\mathbf{2}, \mathbf{4}]$, at position 7)
- If $k = 7$, it would return 7 (second group $[\mathbf{2}, \mathbf{4}]$, at position 7)
- If $k = 8$, it would return -1

Next, write another function:

```
vector<int> starting_positions (const vector<int>& x, const vector<int>& y);
```

that given two vectors x and y returns a vector with all the positions in which the subvector $2x$ occurs in y.
The main program (already provided) uses the above functions to process a sequence of pairs of vectors (x_t, y_t), and for each pair creates and stores an `InputPos` struct, that contains both vectors (x, y), the vector of occurrence positions (nk), and the position of the pair in the input sequence (t).

```
struct InputPos {
  int t;             // number of pair (x_t,y_t)
  vector<int> x;     // input sequence x_t
  vector<int> y;     // input sequence y_t
  vector<int> nk;    // starting positions of 2*x_t in y_t
};
```

The program outputs each vector pair in the input, with the positions where the subvector was found. the pairs are decreasingly sorted by the number of occurrences of $2x$ in y, and by the pair position in the input in case of tie.
Thus, you will need to complete the comparison function that will appropriately sort the structs:

```
bool compareInputPos (const InputPos& inp1, const InputPos& inp2);
```

You MUST use the following code. Changing it outside the indicated blocks will render your program INVALID

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

// DO NOT MODIFY!: THIS STRUCTURE MUST BE USED TO STORE THE
// INPUT SEQUENCES AND THE OBTAINED STARTING POSITIONS FOR
// EACH PAIR OF SEQUENCES
struct InputPos {
  int t;             // number of pair (x_t,y_t)
  vector<int> x;     // input sequence x_t
  vector<int> y;     // input sequence y_t
  vector<int> nk;    // starting positions of 2*x_t in y_t
};

// DO NOT MODIFY!: THIS FUNCTION READS A VECTOR OF n INTEGERS
vector<int> read_vector(int n) {
  vector<int> x(n);
  for (int i = 0; i < n; ++i) cin >> x[i];
  return x;
}

// DO NOT MODIFY!: THIS FUNCTION ONLY OUTPUTS THE RESULTS
// (STORED IN A VECTOR OF InputPos)
void write_results(const vector<InputPos>& vInputPos) {
  int nn = vInputPos.size();
  for (int i = 0; i < nn; ++i) {
    // Outputs the number of positions
    cout << vInputPos[i].nk.size() << ": ";
    if (vInputPos[i].nk.size() == 0) cout << "-";
    // Outputs the positions
    for (int j = 0; j < vInputPos[i].nk.size(); ++j) {
      if (j != 0) cout << " ";
      cout << vInputPos[i].nk[j];
    }
    // Outputs the input sequences
    cout << " / ";
    for (int j = 0; j < vInputPos[i].x.size(); ++j) {
      if (j != 0) cout << " ";
      cout << vInputPos[i].x[j];
    }
    cout << " / ";
    for (int j = 0; j < vInputPos[i].y.size(); ++j) {
      if (j != 0) cout << " ";
      cout << vInputPos[i].y[j];
    }
    cout << endl;
  }
}
```

```
///------------------------------------------------------------------------------

// Pre: 0<=k<y.size()
// Post: The result is the first position i>=k where vector 2*x is found in y,
//       or -1 if no such position exists
int subvector_from_k (const vector<int>& x, const vector<int>& y, int k) {
  // ADD YOUR CODE HERE
}

// Pre: x.size()>0 and y.size()>0
// Post:The result is a vector containing all the positions in y where
//        subvector 2*x occurs.
vector<int> starting_positions (const vector<int>& x, const vector<int>& y) {
  // ADD YOUR CODE HERE
}

// Comparison function to sort the output as required
bool compareInputPos (const InputPos& inp1, const InputPos& inp2) {
  // ADD YOUR CODE HERE
}

// DO NOT MODIFY!: MAIN PROGRAM IS ALREADY COMPLETE
int main() {
  vector<InputPos> vInputPos;
  int t = 1;
  int n;
  while (cin >> n) {
    InputPos inp;
    inp.x = read_vector(n);
    int m;
    cin >> m;
    inp.y = read_vector(m);
    inp.t = t;
    inp.nk = starting_positions(inp.x,inp.y);
    vInputPos.push_back(inp);
    ++t;
  }

  sort(vInputPos.begin(), vInputPos.end(), compareInputPos);

  write_results(vInputPos);
}
```

Exam score: 3.500000 **Automatic part:** 30.000000%

Input

Input consists of a sequence of pairs of non-empty vectors of integers.

Output

The output consists of the input vector pairs, with the number of occurrences of $2x$ in y, and their position in the input sequence. The pairs must be sorted by the occurrence number, and by their position in the input in case of a tie, following the format of the examples.

Sample input

```
3
1 2 3
10
2 4 6 2 4 6 2 4 6 2

2
1 1
7
2 2 2 2 22 2 2

2
1 1
1
2

1
7
6
4 6 0 10 14 666

2
1 1
8
2 3 2 3 2 3 2 3

1
7
6
4 6 0 10 666 14

1
7
6
4 6 14 666 0 10
```

Sample output

```
4: 0 1 2 5 / 1 1 / 2 2 2 2 22 2 2
3: 0 3 6 / 1 2 3 / 2 4 6 2 4 6 2 4 6 2
1: 4 / 7 / 4 6 0 10 14 666
1: 5 / 7 / 4 6 0 10 666 14
1: 2 / 7 / 4 6 14 666 0 10
0: - / 1 1 / 2
0: - / 1 1 / 2 3 2 3 2 3 2 3
```

Problem information

Author : Enrique Romero Merino
Generation : 2018-12-21 11:50:43