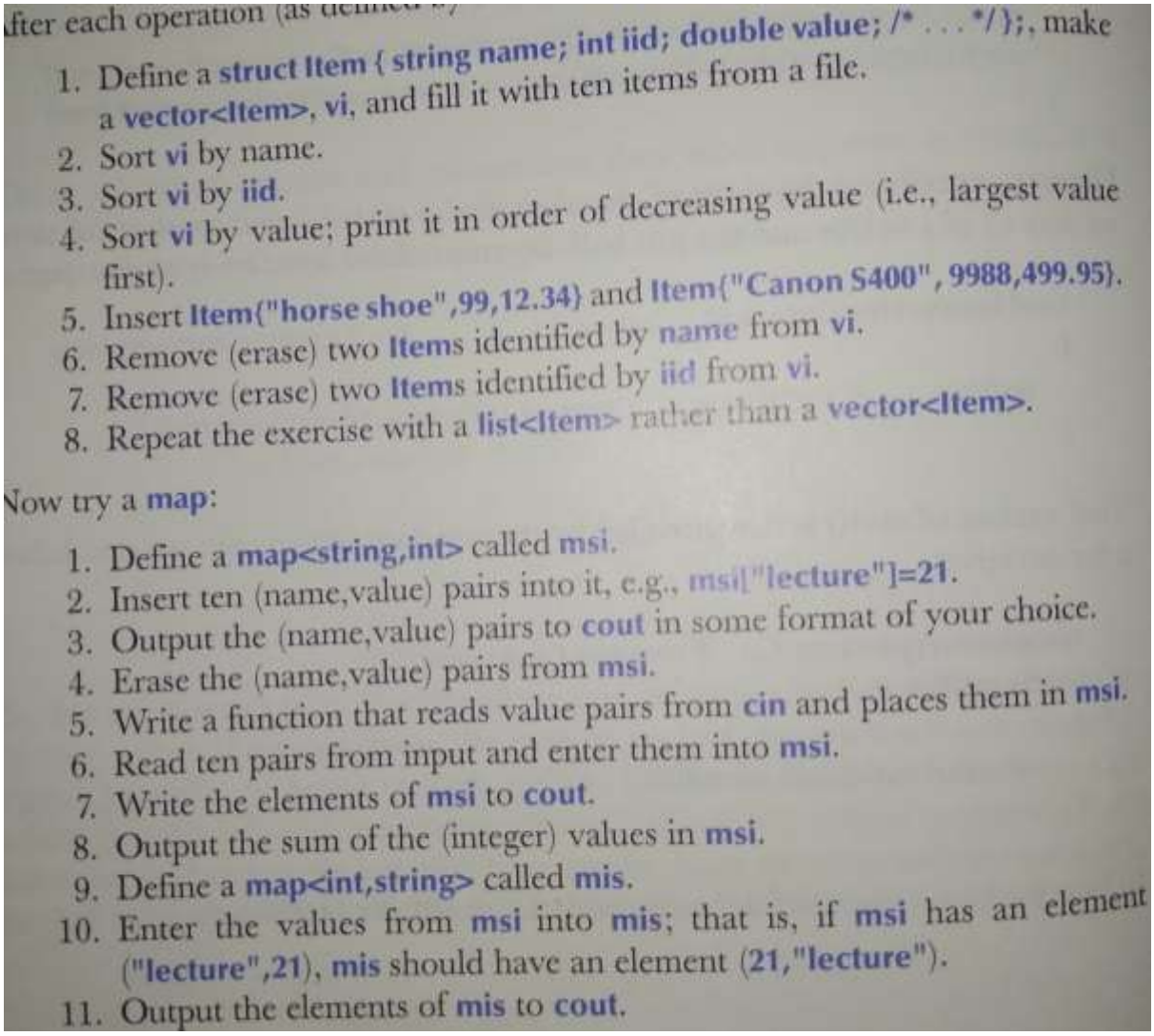
drill-1.cpp – first part of the drill exercise from Chapter 21

drill-2.cpp – second part of the drill exercise from Chapter 21



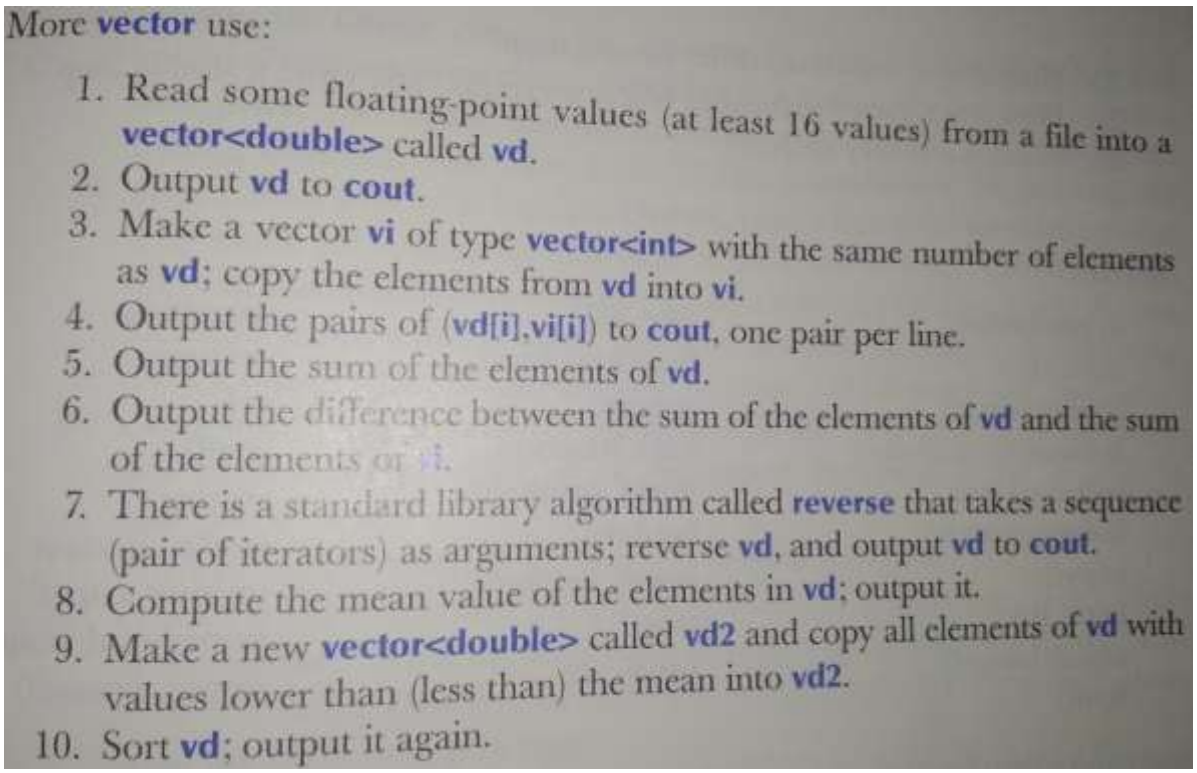
fter each operation (as defined *)

1. Define a **struct Item { string name; int iid; double value; /* . . . */};**, make a **vector<Item>**, **vi**, and fill it with ten items from a file.
2. Sort **vi** by name.
3. Sort **vi** by **iid**.
4. Sort **vi** by value; print it in order of decreasing value (i.e., largest value first).
5. Insert **Item{"horse shoe",99,12.34}** and **Item{"Canon S400",9988,499.95}**.
6. Remove (erase) two **Items** identified by **name** from **vi**.
7. Remove (erase) two **Items** identified by **iid** from **vi**.
8. Repeat the exercise with a **list<Item>** rather than a **vector<Item>**.

Now try a **map**:

1. Define a **map<string,int>** called **msi**.
2. Insert ten (name,value) pairs into it, e.g., **msi["lecture"]=21**.
3. Output the (name,value) pairs to **cout** in some format of your choice.
4. Erase the (name,value) pairs from **msi**.
5. Write a function that reads value pairs from **cin** and places them in **msi**.
6. Read ten pairs from input and enter them into **msi**.
7. Write the elements of **msi** to **cout**.
8. Output the sum of the (integer) values in **msi**.
9. Define a **map<int,string>** called **mis**.
10. Enter the values from **msi** into **mis**; that is, if **msi** has an element **("lecture",21)**, **mis** should have an element **(21,"lecture")**.
11. Output the elements of **mis** to **cout**.

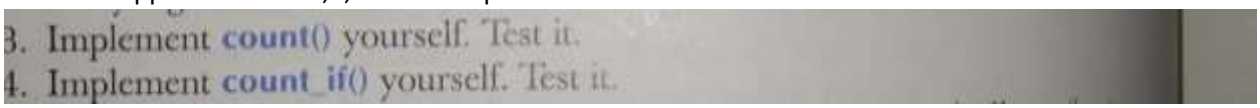
drill-3.cpp – third part of the drill exercise from Chapter 21



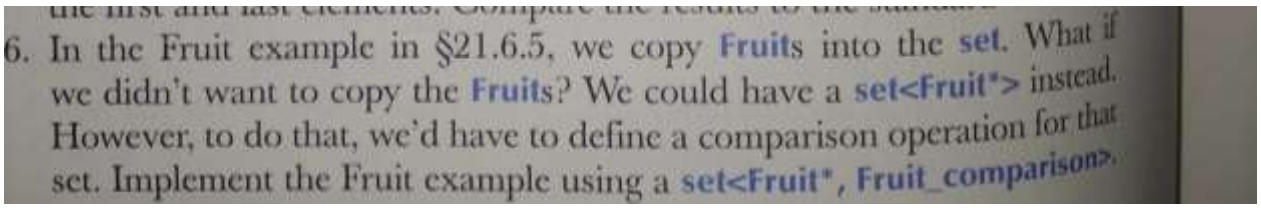
More **vector** use:

1. Read some floating-point values (at least 16 values) from a file into a **vector<double>** called **vd**.
2. Output **vd** to **cout**.
3. Make a vector **vi** of type **vector<int>** with the same number of elements as **vd**; copy the elements from **vd** into **vi**.
4. Output the pairs of (**vd[i]**,**vi[i]**) to **cout**, one pair per line.
5. Output the sum of the elements of **vd**.
6. Output the difference between the sum of the elements of **vd** and the sum of the elements of **vi**.
7. There is a standard library algorithm called **reverse** that takes a sequence (pair of iterators) as arguments; reverse **vd**, and output **vd** to **cout**.
8. Compute the mean value of the elements in **vd**; output it.
9. Make a new **vector<double>** called **vd2** and copy all elements of **vd** with values lower than (less than) the mean into **vd2**.
10. Sort **vd**; output it again.

exercises.cpp – exercises 3,4,6 from Chapter 21



3. Implement **count()** yourself. Test it.
4. Implement **count_if()** yourself. Test it.



6. In the Fruit example in §21.6.5, we copy **Fruit**s into the **set**. What if we didn't want to copy the **Fruit**s? We could have a **set<Fruit*>** instead. However, to do that, we'd have to define a comparison operation for that set. Implement the Fruit example using a **set<Fruit*, Fruit_comparison>**.